Security Analysis of Privately Verifiable Privacy Pass

Konrad Hanff
konrad@khanff.de
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany

Anja Lehmann
anja.lehmann@hpi.de
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany

Cavit Özbay
cavit.oezbay@hpi.de
Hasso Plattner Institute, University of Potsdam
Potsdam, Germany

Abstract

Privacy Pass is an anonymous authentication protocol which was initially designed by Davidson et al. (PETS'18) to reduce the number of CAPTCHAs that TOR users must solve. It issues single-use authentication tokens with anonymous and unlinkable redemption guarantees. The issuer and verifier of the protocol share a symmetric key, and tokens are *privately verifiable*. The protocol has sparked interest from both academia and industry, which led to an Internet Engineering Task Force (IETF) standard. While Davidson et al. formally analyzed the original protocol, the IETF standard introduces several changes to their protocol. Thus, the standardized version's formal security remains unexamined. We fill this gap by analyzing the IETF standard's privately verifiable Privacy Pass protocol.

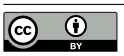
In particular, there are two main discrepancies between the analyzed and standardized version: First, the IETF version introduces a *redemption context*, that can be used for blindly embedding a validity period into the Privacy Pass tokens. We show that this variant has significant differences to public metadata extension that has been proposed for the same purpose in the literature. Redemption context offers better privacy and security than public metadata. We capture both stronger guarantees through game-based security definitions and show that the currently considered one-more unforgeability notion for Privacy Pass is insufficient when a redemption context is used. Thus, we propose a new property, *targeted context unforgeability*, and prove its incomparability to one-more unforgeability. Second, Davidson et al. focused on a concrete Diffie-Hellman based construction, whereas the IETF version is built generically from a *verifiable oblivious pseudorandom function* (VOPRF). Further, the analyzed protocol omitted the full redemption phase needed to prevent double-spending. We prove that the generic IETF construction satisfies the desired security and privacy guarantees covering the full life-cycle of tokens. Our analysis relies on natural security properties of VOPRFs, providing compatibility with any secure VOPRF instantiation. This enables crypto agility, e.g., allowing to switch to efficient quantum-safe VOPRFs when they become available.

CCS Concepts

• **Security and privacy → Privacy-preserving protocols**.

Keywords

Privacy Pass, Anonymous Token, IETF



This work is licensed under a Creative Commons Attribution 4.0 International License.
CCS '25, Taipei, Taiwan
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1525-9/2025/10
https://doi.org/10.1145/3719027.3765172

ACM Reference Format:
Konrad Hanff, Anja Lehmann, and Cavit Özbay. 2025. Security Analysis of Privately Verifiable Privacy Pass. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25), October 13–17, 2025, Taipei, Taiwan.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3719027.3765172

1 Introduction

Privacy Pass is a cryptographic protocol for anonymous authentication, initially invented to improve user experience in content delivery networks (CDNs) while preserving privacy [13]. CDNs such as Cloudflare support websites by hosting resources and enforcing security policies, e.g., blocking IPs or requiring users to complete proof-of-work challenges like CAPTCHAs. However, solving CAPTCHAs repeatedly across websites degrade usability. While cookies across websites could streamline access, they compromise privacy, especially for anonymity network users like those on Tor.

To address this, Davidson et al. [13] introduce a privacy-preserving mechanism where clients receive a batch of anonymous tokens after solving a CAPTCHA. The initial protocol was based on a Diffie-Hellman based verifiable oblivious pseudorandom function (VOPRF), where the issuer (and verifier) hold a symmetric VOPRF key. During issuance, the user can blindly obtain a number of VOPRF evaluations on random input values, and later redeem them by showing that it knows a valid input/output PRF pair for the issuer's secret key. The protocol aims at two core properties: *unlinkability*, ensuring that the user cannot be tracked between issuance and redemption, and *double-spend protection*, preventing a client from redeeming more tokens than she obtained.

After the original *privately-verifiable* version – where issuance and redemption require the same secret key – also a *publicly-verifiable* variant has been proposed [4, 5, 10]. Therein, redemption can be done using only a public key of the issuer, which can be build by using blind signatures instead of a VOPRF. The Privacy Pass protocol has seen notable adoption in the real-world already, it is implemented as a browser extension by Cloudflare [1], and also adopted in further applications: Meta's WhatsApp leverages it for anonymous logging called *De-identified telemetry* (DIT) [2], Google uses it in *Private State Tokens* (PST) and integrated into Chrome API [3], and Apple deployed *Private Access Tokens* (PAT) [29].

1.1 IETF Standard for Privacy Pass

Given its broad range of potential use cases and the involvement of major industry players, Privacy Pass became a key candidate for standardization. In 2020, the IETF formed a dedicated working group and standardized the protocol across three RFCs [8, 14, 26], which we refer to as the *IETF specification* throughout the paper for simplicity. The IETF specification contains a privately verifiable

and a publicly verifiable variant of the Privacy Pass protocol. Our work focuses solely on the privately verifiable protocol version.

The IETF specification diverges from the original protocol in both design and intended security goals. While the original work [13] particularly aimed to reduce the number of anonymous network requests that are challenged with CAPTCHAs by CDNs, the IETF specification's goal is to provide a generic protocol that could replace token-based authentication systems with privacy-preserving variants. Thus, it aims at more generic security and privacy properties, which must be restated for this broader scope. The IETF standard also differs from the original protocol design in two key aspects: 1) it introduces a *redemption context* into issuance and redemption and 2) it builds the privately-verifiable protocol generically from VOPRFs, while [13] used a concrete DH-based construction.

Redemption Context. The IETF specification defines a mechanism for embedding information into Privacy Pass tokens using a redemption context. This context is chosen by the verifier and made available to the client. The client can then obtain a token bound to this redemption context without revealing it to the issuer.

The main goal is to use the redemption context for ensuring token freshness. The verifier can assign a different, randomly chosen redemption context for a certain timeframe, e.g., day or week, effectively limiting token validity to specific periods. This prevents token hoarding, where malicious clients aim to collect a large number of tokens through a prolonged time frame and then use them for DDoS attacks. It also reduces complexity for the verifier: to detect and prevent double-spending of tokens, the verifier must store the redeemed tokens and check that a newly presented one has not been used before. When encoding a validity period in the token, this storage and checking can be reset with every new period.

Redemption Context ≠ Metadata. Interestingly, this redemption context has not been analyzed in any works so far. It was not present in the original protocol [13], and while several academic proposals explored ways to embed information into Privacy Pass tokens, they have done this for a different concept, namely *issuance metadata* [5, 9, 24, 28]. Issuance metadata serves the same purpose as the redemption context, but in contrast is revealed during issuance.

Therefore, using a redemption contexts offers much stronger privacy than metadata: In protocols with metadata, unlinkability is immediately reduced to an anyonymity set of users that have revealed the *same* metadata during issuance. As the redemption context is fully hidden towards the issuer, unlinkability there is maintained across all users. This aspect also allows to use much more fine-grained contexts and ensure stronger security, e.g., even allowing verifiers to require freshly issued tokens when needed. While redemption context still offers some level of privacy in this case, requesting fresh tokens with public metadata would make the issuance and redemption trivially linkable. We provide a detailed comparison of these approaches in Section 3 and Figure 2.

Generic VOPRF-based Design. The privately-verifiable IETF construction builds the Privacy Pass protocol generically from VOPRFs, a cryptographic primitive also standardized by the Internet Research Task Force (IRTF) [11]. Such a modular design is beneficial for ensuring cryptographic agility and supporting future adaptations of the specification. For example, if the VOPRF specification evolves to support a post-quantum secure construction such as [6, 16], the Privacy Pass IETF specification can adopt it without requiring further foundational security analysis. However, this requires a generic security analysis showing which properties of the VOPRF are necessary to securely realize the Privacy Pass protocol.

While the original privately verifiable Privacy Pass construction [13] was inspired by a concrete VOPRF instantiation, 2HashDH [23, 30], their security analysis is tied directly to this specific instantiation. Furthermore, their analysis relies on the non-standard *one-more ElGamal decryption* assumption which is not used in any analysis on 2HashDH VOPRF to our knowledge.

Lack of Security Analysis of IETF Privacy Pass. While Privacy Pass, or rather the general concept of *anonymous tokens*, has enjoyed significant attention from academia, none directly analyze the IETF specification. Instead, they extend the original protocol, e.g., by enabling public verifiability [4, 5, 10] or supporting additional features such as issuance metadata [4, 5, 9, 24, 28], or *non-transferablility* [15] restricting the redemption of a token to its assigned owner.

A notable exception is Chu et al. [10], who analyze an internet draft for *rate-limited Privacy Pass* [18]. Their privacy model assumes honest issuer (or attester) due to the underlying extension, whereas we are interested in the privacy properties in the presence of a malicious issuer – which is one of the main goals of Privacy Pass. Furthermore, although their formal syntax includes the redemption context, their security analysis omits the relevant unforgeability properties needed for the use cases of redemption context such as token expiry or freshness. Foremost, the focus of our work is different: we target the privately-verifiable version of the core IETF standard and aim at capturing and formalizing the impact of the redemption context on the security and privacy properties.

In fact, we are not alone in raising concerns on the formal security of the IETF specification. Notably, Wood – a co-author of both the original Privacy Pass paper [13] and the IETF specification – has highlighted the lack of formal security analysis for the standardized protocol. Wood argues that to fully understand the protocol's limitations, the working group and its users would benefit from refining the security and privacy goals of the protocol as described in the IETF specification [31]. This leads us to two main open questions that we address with our work:

(1) *What are the formal security and privacy goals of the IETF Privacy Pass specification?*
(2) *Can we show that the IETF construction satisfies the intended security goals by relying on natural properties of VOPRFs?*

1.2 Our Contributions

We present the first formal analysis of the IETF version of privately-verifiable Privacy Pass. Including the redemption context in our model is a core contribution as it revealed the need for an additional property. We formalize all desired security and privacy properties from privately-verifiable Privacy Pass through game-based definitions. Finally, we analyze the VOPRF-based construction from the IETF standard, and prove its security based on natural VOPRF-assumptions. Below, we give an overview of our contributions.

Formal Security Model. We present a formal security model capturing the desired security and privacy as expressed in the IETF

specification, in particular including the redemption context. We introduce the syntax of *privately verifiable Privacy Pass with redemption context* (PPass-RC), formalize its security properties, and assess how these properties align with the goals of the IETF specification.

Notably, we recognize that a new security notion is needed when redemption context is used. The commonly targeted property is **one-more-unforgeability**, ensuring that malicious client should not be able to generate more valid tokens than were issued by the issuer, meaning the number of redeemed tokens must not exceed the number of issued tokens [14, section 3.5]. We observe that one-more unforgeability alone is insufficient to support applications enabled by redemption contexts, such as enforcing token expiry or freshness: it does not guarantee that a malicious client is unable to forge a token for a *targeted* context. We therefore introduce a new property **targeted context unforgeability** ensuring that a client's previous interactions with the issuer should not help it forge a valid token for a randomly chosen redemption context. We also show that the one-more unforgeability and the targeted context unforgeability are incomparable, neither implies the other. This justifies our inclusion of targeted context unforgeability as a distinct and necessary security goal for the IETF specification.

For privacy, we define the desired **unlinkability** which requires that issuance of a token should not be linkable to its redemption [14, section 3.3]. This notion is stronger than unlinkability in the presence of public metadata, as it does not limit unlinkability to users that have revealed the same context during issuance.

Further, we also introduce the notion of **redemption robustness**, ensuring that if an honest client interacts with an honest issuer, the resulting tokens should be valid and successfully redeemable. Our notion extends the classic correctness property to also include malicious clients, guaranteeing that their behaviour cannot hinder honest clients from successful redemption.

Security Analysis. Finally, we formally analyze the security of the PPass-RC construction PP-VOPRF defined in the IETF specification. This construction is built generically from VOPRFs, which we first recast in our syntax. We then prove that PP-VOPRF satisfies all the security properties defined in Section 5 – redemption robustness, one-more and targeted context unforgeability, and unlinkability – by only relying on the natural security properties of VOPRFs which are correctness, obliviousness, and pseudorandomness.

Outline. In Section 2 we recap the IETF standard, and the expressed functionality and security properties therein. A particular focus is given to the redemption context, which we compare to the concept of issuance metadata in Section 3. In Section 4, we define the syntax of privately verifiable Privacy Pass with redemption context, and state the formal security model in Section 5. Finally, Section 6 proves the security of the IETF protocol in our model.

2 Overview of the IETF Privacy Pass Standard

This section describes the Privacy Pass protocol and its core procedures according to the IETF specification [14, 20, 26]. It then outlines the desired security properties of the protocol and our abstraction of the specification's entities and processes. For completeness, in Appendix A, we summarize the ongoing internet drafts of the Privacy Pass working group which are not standardized, yet.

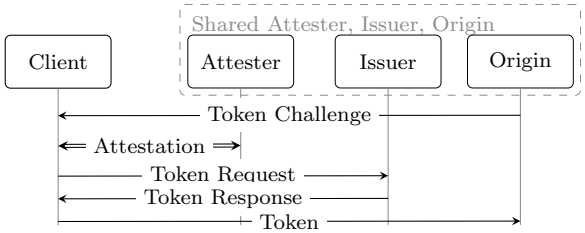


Figure 1: Interaction of parties in Privacy Pass.

2.1 High-Level Functionality

The IETF specification [14] defines four entities in a Privacy Pass protocol: *client*, *origin*, *attester* and *issuer*. The specification discusses a number of deployment models that differ in which of the roles issuer, origin, and attester are controlled by the same entity [14, section 4]. Thus, a "distributed" four-party system model proposed by the IETF specification is not enforced for all use cases, and in fact is mostly relevant for the *publicly*-verifiable setting.

For our analysis of the *privately- verifiable* variant, we consider the shared attester, issuer, origin deployment model, where all three belong to the same entity. Below, we outline the roles and interactions of these entities, followed by a description of the security goals of a Privacy Pass system and how this combined deployment model choice is suitable for the privately-verifiable Privacy Pass.

2.1.1 Entities in Privacy Pass. In a Privacy Pass system, a *client* seeks access to websites or online services. The *origin*, responsible for providing content or services, requires clients to authenticate. For instance, when a client sends an HTTP request to a content delivery network (CDN) such as Cloudflare to access a website, the CDN may require the client to prove that it is not a robot.

To facilitate this process, two additional entities come into play: the *attester* and the *issuer*. The attester is responsible for validating that the client has the necessary properties required by the origin to authenticate. In our example, the attester might be a CAPTCHA service that verifies whether the user can solve a simple puzzle, thereby confirming their identity as a non-robot. The issuer creates authentication tokens for clients, which are validated by the origin before granting access to the requested content. In this case, Cloudflare, serving as the origin, attester and issuer, generates authentication tokens for users accessing its website via HTTP request. Figure 1 gives an overview of the interactions of the Privacy Pass entities.

2.1.2 Token Issuance and Redemption. The Privacy Pass protocol has two main phases: token issuance and token redemption. Which tokens can be used in the redemption phase is steered via the *token challenge* which the origin sends upon the clients access request.

The token challenge determines how a client can authenticate to access the service. The token challenge value in the specification determines the accepted token type (e.g., privately or publicly verifiable), issuer and origin identifiers, and an extra field for redemption context. Focusing on privately verifiable tokens, the token type is fixed. Moreover, issuer and origin identifiers are redundant as these parties share a symmetric key. Our model therefore concentrates on the redemption context, which is explicitly defined by the standard as a way to restrict a token's valid context.

Redemption Context. The redemption context helps origins to require tokens bound to a specific context. However, one should keep in mind that the redemption contexts cannot be treated as attributes that are validated by the issuer. On the contrary, the protocol requires that the redemption context must not be revealed to the issuer during the token issuance to preserve the clients' privacy; which we discuss more in our security goals.

The specification provides several examples of redemption contexts. For instance, the tokens can be bound to a client's network by using autonomous system numbers (ASN), or be bound to timeframes. For the latter, the origin assigns a random redemption context to each timeframe and announces these values by time. Using random and unpredictable redemption contexts for each new timeframe ensures that malicious clients cannot obtain tokens for future timeframes. An origin can also use a random and fresh redemption context for a specific token challenge to enforce the client to obtain a freshly issued token from the issuer [14, section 2.1.1.2].

If the client already holds an unused token satisfying the token challenge, it can directly run the *token redemption* process and present this token to the origin. Otherwise, it must first obtain a token satisfying this token challenge via the *token issuance* process.

Token Issuance. To obtain a valid token for a token challenge, the client must first complete the attestation process, confirming the client's eligibility to get a token. How the client's attestation works or how the issuer obtains this information is outside the specification [14, section 3.5.1]. Once validated by an attester, the client obtains tokens by interacting with the issuer.

The issuer and the client can run multiple instances of the token issuance protocol at once if token caching for later use is allowed. However, the issuer must limit the maximum number of tokens issued at once. Otherwise, malicious clients can collaborate on a *hoarding attack* by caching many tokens and give them to a single user to redeem at once. Such attacks can be mitigated by employing expiry dates on tokens or rotating the issuer key so that the tokens issued with the old issuer key becomes invalid [14, section 7.1].

Token Redemption. In the redemption phase, the client simply sends an unused token that satisfies the token challenge to the origin. The origin validates that the token is valid for the previously chosen token challenge and was not redeemed before. Usually, the latter step requires origins to keep a list of redeemed tokens and perform a double-spending check.

The list of redeemed tokens grows by time, and using the redemption context to steer the token expiry reduces the bookkeeping burden. For instance, when using the redemption context to encode a timeframe, the origin need not store the tokens with expired timeframes. Similarly, by having client networks in the redemption context, each server of the origin can have smaller used tokens list dedicated to the client network it serves [26, section 2.2.2].

2.2 Security Goals & Deployment Model

The core goal of Privacy Pass is to provide this authentication process in a way that is not only secure, but also privacy-preserving. In essence, this requires the following intuitive security and privacy requirements that we later capture in our formal security model:

Correctness: Interaction of an honest client with an honest issuer must lead to a valid token and successful redemption.

One-More Unforgeability: Malicious clients cannot create valid tokens without the issuer's contribution, i.e., clients cannot redeem more tokens than issued [14, section 3.5].

Unlinkability: The issuance of a token cannot be linked to its redemption. Note that this requires the redemption context to be hidden during the issuance sessions [14, section 3.3].

The latter corresponds to *redemption context* unlinkability in the IETF standard. The standard lists three additional privacy requirements, that are not relevant for our study though: *attester-origin* unlinkability – which pertains to protocol parts outside the cryptographic core protocol. The other two – *origin-client* and *issuer-client* unlinkability – are implied by redemption context unlinkability in the shared issuer/attester/origin deployment model. We discuss these other requirements in more detail in Appendix B.

Deployment Model Choice. As mentioned, the IETF standard introduces the dedicated entities of origin, attested and issuer to be able to support different deployment settings, where each role could be run by a dedicated entity. For our work of studying the privately-verifiable Privacy Pass protocol, we focus solely on the setting where all are controlled by the same entity – which is reflected in our model by assuming that they all share the same corruption status, so they are either all honest or all corrupt. Let us explain why we believe this is a proper abstraction.

First, privately-verifiable Privacy Pass requires the issuer and the origin to share a secret key that is used to issue and redeem tokens. Thus, for the correctness and the one-more unforgeability properties of privately verifiable tokens, both the origin and the issuer have to remain honest. Furthermore, the attester does not contribute to the cryptographic protocol, so modeling the attester as a separate entity is not relevant to the cryptographic protocol.

Second, for the unlinkability property, allowing the attester, the issuer, and the origin to collude leads a stronger model than splitting them: they can share all their views and try to exploit their interactions with the clients to break their privacy. We show that the privately-verifiable Privacy Pass standard achieves the desirable privacy guarantees in this strong model. Thus, there is no need to treat weaker models that assume non-collision among the attester, the issuer, and the origin. For simplicity, we will refer to this combined entity as the issuer in the rest of the paper.

Deployment Model Beyond the Cryptographic Protocol. Note that the IETF specification's considerations is not limited to the cryptographic protocol and client-specific information can be revealed beyond what is covered by the cryptographic values. For example, the attester, the issuer, and the origin could also link their interactions with clients by using the IP addresses of clients. Our analysis focuses solely on the core cryptographic protocol of the privately-verifiable Privacy Pass and its suitable deployment models. Thus, considering different deployment models for the other layers of the system such as the network layer can still be meaningful.

3 Redemption Context vs. Issuance Metadata

The main contribution of our work is the formal treatment of the redemption context and study how this impacts the security and

	Visibility			Use Case		
	Client	Issuer/issuance	Issuer/redemption	Issuer-certified attr.	Token expiry	Token freshness
Redemption context	✓		✓		✓	✓
Public metadata	✓	✓	✓	✓	✓	
Private metadata		✓	✓	(✓)		

Figure 2: Comparisons of redemption context and public/private metadata.

privacy of Privacy Pass. Interestingly, while the redemption context is the only option explicitly chosen by the IETF standard, the academic literature so far exclusively focuses on another concept: private and public meta included during issuance data [5, 9, 24, 28]. The current IETF standard does not support private or public metadata by default, but only covers the necessary data structures to allow future constructions that may support them. Although there is a recent draft for Privacy Pass with public metadata support [20], this draft has yet to be incorporated into the Privacy Pass standard.

Roughly, all approaches aim to encode extra information within tokens: *private/public issuance metadata* [14, section 3.5]. The distinct characteristics among these approaches stem from the visibility of the metadata/context to different parties at various stages of the protocol. Figure 2 illustrates a comparison of the visibility of additional token data during the different phases of the protocol.

This section provides a comparison how these approaches differ and where the redemption context can provide stronger security and privacy than solutions with metadata. These additional properties are then formally defined in Section 5. We compare the three approaches of embedding extra information into Privacy Pass tokens by analyzing their use cases and impacts on the security goals of the Privacy Pass standard which was presented in Section 2.2.

3.1 Issuance Metadata

Issuance metadata is always visible to the issuer during the token issuance and the redemption. However, private and public metadata differ in their visibility to the client.

Private Metadata. It remains unknown to clients and is only visible to the issuer [14, section 3.5], enabling the issuer to link issuance and redemption without the client's knowledge. This extension is proposed for tracking abusive clients or marking specific tokens to monitor. However, it also poses a risk of violating Privacy Pass's privacy goal as a malicious issuer can exploit private metadata to track *any* client. Thus, the rest of our analysis focuses on public metadata which also has similar use cases to the redemption context.

Public Metadata. Unlike private metadata, *public metadata* is always visible to both the client and the issuer [14, section 3.5], allowing clients to reject tokens that compromise their privacy too much. Thus, protocols can support arbitrary values as public metadata [4, 9, 28] which gives rise to applications such as embedding *issuer-certified attributes* (e.g., free or paid membership) to tokens or implementing *token expiry* without key rotation [14, section 7.1].

By encoding a token expiry date into the public metadata, issuers can prevent clients from accumulating tokens and redeeming them all at once. This approach reduces the need for the issuer to store all redeemed tokens, decreasing their state size. However, the precision of the validity period determines the size of the clients' anonymity sets, so it should be coarsely grained to protect client privacy.

3.2 Redemption Context

The redemption context, visible to the client, differs from public or private metadata as the issuer learns it only upon redemption, but not during the issuance.

Better privacy through context hiding during issuance. The main advantage of the redemption context is that it can implement the main use case of public metadata – enforcing token expiry to keep the issuer's state small – but in a much more privacy-friendly way, as the issuance and redemption cannot be linked through the encoded information. With redemption context, the issuer publishes random context values for each timeframe, and the user blindly includes them in their issuance request. In contrast to public metadata, this allows to use fine-grained validity periods (which are better for efficiency) without harming the client's privacy, as the value is not visible during issuance. For example, issuers can support tokens without expiration dates for less critical operations like read requests, while requiring recently obtained tokens for more crucial operations like write requests.

Stronger Privacy: Issuance of tokens cannot be linked to their redemption even among tokens that contain *distinct* embedded information.

This is impossible to achieve when private/public metadata is used as they are known by the issuer both during the token issuance and redemption. Note that the IETF specification already targets such a strong unlinkability goal [14, section 3.3].

Stronger Security through fine-grained Token Freshness. The extreme case of the token expiry application with redemption context is requiring *fresh tokens* by choosing a dedicated random context for a client's authentication request [26, section 2.1.2]. This is not possible with public metadata without compromising client privacy, because a unique public metadata value that forces the client to run the issuance protocol immediately also allows the issuer to link the issuance and redemption protocol.

The privacy gain of redemption context comes with a caveat: as redemption context is not visible to the issuer, it cannot be used as an issued-certified attribute unlike public metadata.

Need for Stronger Unforgeability Notion. The IETF specification takes the redemption context into account when setting its privacy goals, but not when defining unforgeability. Specifically, it only considers one-more unforgeability, ensuring that an adversary cannot redeem more tokens than were issued by the issuer. Originally, this notion does not consider any embedded information in tokens.

While existing works have extended one-more unforgeability to include private and public metadata [5, 9, 24, 28], a similar extension for the redemption context has not been proposed yet.

We formally capture such one-more unforgeability but also notice that it is not sufficient for enforcing token freshness or token expiry through the redemption context: it does not guarantee that a client cannot use previous issuance interactions to forge a token for a redemption context that will be announced in the future. Therefore, enforcing token freshness or token expiry dates through the redemption context requires an additional security property beyond one-more unforgeability:

Targeted Context Unforgeability: The interactions of a client with the issuer in the past does not help the client to forge a token on a randomly chosen redemption context.

The different use cases and how they are covered by the various token types are summarized in Figure 2. Given its superior privacy properties for the client, we suggest that the redemption context should be used for token expiration over public metadata.

4 Definition of Privately-Verifiable Privacy Pass

Building upon our analysis of the IETF standard, we now present our generic syntax for privately-verifiable Privacy Pass with redemption context (PPass-RC) , which considers multiple clients that interact with one issuer – combining the roles of the attester, issuer and origin. We start by defining the syntax of the generic PPass-RC system, and then compare our model to previous works. A particular focus is the redemption context, that has not been covered in formal works yet, but brings advantages for the security and privacy of the Privacy Pass system. Beyond the redemption context, our model also makes the issuer's state and redemption phase explicit, which has often been kept outside of the definition but is crucial to formally capture the desired double-spending prevention.

4.1 Syntax of PPass-RC

The PPass-RC system consists of a setup and key generation phase, an issuance protocol and a redemption algorithm. The setup algorithm generates the public parameters of the system such as the underlying group description for the cryptographic operations.

4.1.1 Key Generation. After the setup is done, the issuer runs the key generation algorithm Kg to generate a key pair, consisting of a secret key sk and a verification key vk. The issuer uses the secret key to issue and verify tokens, while the clients use the verification key to ensure that the issuer behaves honestly. Thus, the clients must receive vk from the issuer before initiating the issuance protocol.

We assume that all clients receive the same verification key to prevent the issuer from partitioning clients by the secret key that verifies their tokens [14, section 6.2]. To ensure the verification key consistency, Davidson et al. [12] suggest using a trusted consistency mirror that distributes the same key material to all clients.

4.1.2 Issuance Protocol. The issuance protocol [8] defines how Privacy Pass tokens are created between a client and an issuer. This two-message protocol is initiated by the client and consists of three algorithms: Request, Issue, and Finalize.

Requesting a Token. Redemption context models the token challenge that the client must present a satisfying token for it. For a given redemption context ctx and verification key vk, the Request algorithm produces the blinded token request $\overline{x}$, sent to the issuer, and auxiliary data aux, stored by the client to finalize the issuer's response and compute the token.

To capture the issuer behavior in use cases like token freshness and expiry, our targeted context unforgeability definition requires modeling an unpredictable redemption context ctx^*. We achieve this by sampling ctx^* uniformly from a context space C, assuming that C is defined by the public parameters pp. For unpredictability of ctx^*, C must be large enough to prevent the adversary from guessing ctx^* before it is revealed. In practice, this can be done by including a random λ-bit field in the context or by defining contexts as PRF outputs under a secret key held by the issuer or origin, as suggested in the IETF specification [26, section 2.1.1.2].

Issuing and Finalizing a Token. The issuer runs the Issue algorithm with the secret key and issuance request $\overline{x}$ as input, producing an issuance response $\overline{y}$. Upon receiving this response, the client then runs the Finalize algorithm with inputs $\overline{y}$ and state aux, which produces the final token t. We note that the verification key vk could also be defined as input to the Finalize algorithm, but we omit it as the client can also store the verification key in aux.

4.1.3 Redemption protocol. This protocol allows a client to authenticate itself towards an origin to access a service which is provided by the origin. Note that the origin and the issuer are the identical parties in the shared attester, issuer, origin model.

First, the issuer chooses the redemption context value ctx which the client must present a satisfying token to authenticate itself for the service. The issuer sends this ctx to the client, which either picks a not yet redeemed token it has cached from a previous issuance for this context or initiates a new issuance to obtain a new token for this context. We assume that the client always fetches an unused token, either from a cache or by performing an issuance protocol with the issuer, and omit details on how cached tokens are stored.

The client sends a fetched token that satisfies the context ctx to the issuer. Finally, the redemption algorithm Redeem is run by the issuer with the client's token, the issuer key pair, the redemption context ctx value it chose, and its redemption state ist as inputs. We assume that the state is initialized as $ist := \perp$ when the issuer generate its keys. It is modeled to perform double-spending check, and it is updated with each run of the redemption algorithm. The output of Redeem is a bit b, which indicates whether the token was accepted, and an updated state ist'.

We now formally define the syntax we use for PPass-RC.

Definition 4.1 (PPass-RC). A privately verifiable Privacy Pass with redemption context scheme PPass-RC = (Kg, Request, Issue, Finalize, Redeem) is a tuple of algorithms such that:

$\mathsf{Setup}(\lambda) \to pp$**:** Generates public parameters pp from the security parameter λ. We assume that pp is an implicit input to all other algorithms and only make it explicit for the Kg.

$\mathsf{Kg}(pp) \to (sk, vk)$**:** Outputs an issuer key pair (sk, vk).

$\mathsf{Request}(vk, ctx) \to (\overline{x}, aux)$**:** For input verification key vk and a redemption context from the context space $ctx \in C$, it outputs an issuance request $\overline{x}$, and auxiliary state aux.

$\mathsf{Issue}(sk, \overline{x}) \to \overline{y}$**:** Given an issuer secret key sk and an issuance request $\overline{x}$, it outputs and issuance response $\overline{y}$.

$\mathsf{Finalize}(\overline{y}, aux) \to t/\perp$**:** Given the issuance response $\overline{y}$ and auxiliary data aux, it outputs the token t or outputs $\perp$ on failure.

$\mathsf{Redeem}(sk, ctx, t, ist) \to (b \in \{0,1\}, ist')$**:** Verifies the token t for the redemption context ctx using secret key sk, and outputs a boolean result and updated state ist'.

4.2 Comparison to Previous Models

We now discuss the differences between our syntax and previous anonymous token schemes [5, 9, 10, 13, 24, 28].

Redemption Context. Our syntax contains the redemption context ctx as input to both Request and Redeem algorithms. In most of the previous anonymous token schemes [5, 9, 13, 28], the redemption context ctx was not defined at all. This concept of a redemption context/challenge was introduced during the standardization process of the protocol [26] and is not present in the original paper [13], which may be the reason why many publications do not consider it explicitly. Although Chu et al. [10] models the redemption context/token challenge in their syntax, they still do not consider its use cases while setting their security goals.

Inputs of the Request *Algorithm.* Beyond ctx, our Request algorithm only takes the verification key vk as input unlike many prior schemes that include a random nonce as input [5, 13, 24, 28] or output [9]. Since the nonce is a construction-specific detail, we omit it at the definitional level to allow broader designs ideas – similar to Chu et al.'s syntax [10]. Another additional input to the Request algorithm in the IETF specification is called `token_key_id` [8, section 5]. Since this value is deterministically derived from the input verification key, we do not include it explicitly.

Issuer State. In our model, the issuer has a persistent state which it keeps between different redemption protocol runs. Except for the original work of Davidson et al. [13], previous Privacy Pass formalizations from academic literature do not model the issuer with state [5, 9, 10, 24, 28]. The motivation why we introduce state is to allow constructions to define how they prevent double-spending of tokens. Furthermore, modeling the issuer state will allow us to define a stronger correctness property than the previous works which guarantees the token correctness in the presence of malicious clients in Section 5.1. The impact of the issuer state is discussed in more detail in the context of our security model.

Client State. In the original work by Davidson et al. [13], the client has state and stores the tokens it obtained for later use. We do not model this process, as it heavily depends on how the client-side implementation chooses the token to be redeemed in real-world.

5 Security Model for PPass-RC

In this section, we formally define the security and privacy properties of PPass-RC. The desired privacy guarantee is *unlinkability*, ensuring that redeemed and issued tokens can not be correlated – the strengthened aspect here is that this even holds for tokens that have a different redemption context. For security, we define the properties of *one-more unforgeability* and *targeted context unforgeability*. The latter is an additional property needed to ensure that blind issuance cannot be exploited to forge a token for an unpredictable redemption context. We show that both unforgeability notions are incomparable, substantiating the need to consider both in the formal analysis. Finally, we also model a stronger correctness property of Privacy Pass: *redemption robustness*. This property guarantees the validity of a token created by an honest issuer and an honest client even in the presence of other malicious clients.

$\mathcal{O}\mathsf{Issue}(\overline{x})$	$\mathcal{O}\mathsf{Redeem}(ctx, t)$
return $\mathsf{Issue}(sk, \overline{x})$	**return** $(b, ist) \leftarrow \mathsf{Redeem}(sk, ctx, t, ist)$

$\mathsf{Exp}^{\text{red-rob}}_{\text{PPass-RC},\mathcal{A}}(\lambda)$

$pp \leftarrow \mathsf{Setup}(\lambda)$; $(sk, vk) \leftarrow \mathsf{Kg}(pp)$; $ist := \perp$

$ctx \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Issue}, \mathcal{O}\mathsf{Redeem}}(vk)$

$(\overline{x}, aux) \leftarrow \mathsf{Request}(vk, ctx)$; $\overline{y} \leftarrow \mathsf{Issue}(sk, \overline{x})$

$t \leftarrow \mathsf{Finalize}(\overline{x}, aux)$; $b \leftarrow \mathcal{O}\mathsf{Redeem}(ctx, t)$; **return** $b = 0$

Figure 3: The redemption robustness game.

5.1 Redemption Robustness

Redemption robustness defines the correct behavior of the scheme. Previous works on Privacy Pass have established a correctness property, where an honestly issued token should always be accepted during redemption [5, 13, 24, 28]. However, these studies overlooked the fact that the Privacy Pass verification algorithm is stateful to prevent double-spending, so each run of the token verification impacts the behavior of subsequent verification operations.

While a stateful verification does not affect the correctness when all clients are honest, malicious clients may exploit the issuer state to their advantage. By manipulating input to the verification algorithm, they can alter its state and prevent honest clients' tokens from being accepted. Notably, existing works either neglect to model issuer state altogether [5, 24, 28] or omit the presence of malicious clients, even when modeling the issuer's state [13].

Figure 3 depicts the redemption robustness game Expred-rob. In this game, the adversary acts as a malicious client, receiving and redeeming tokens with an honest issuer by using the token issuance and redemption oracles. Note that the token redemption oracle $\mathcal{O}$Redeem outputs the issuer state ist together with the boolean result of the token redemption. This captures that the issuer state ist is not secret (and in fact might be replicated), and revealing it to the adversary gives no advantage to the adversary. Upon completion of its token issuance and redemption queries, the adversary provides a redemption context for which it claims that a fresh, honestly issued token will not redeem successfully.

The game simulates an honest client by issuing a fresh token for the adversarially chosen context and redeems it with the issuer state that resulted from the adversary's redemption attempts. The adversary wins the game if the redemption rejects this token.

Definition 5.1 (Redemption robustness). A PPass-RC scheme is red-rob if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}^{\text{red-rob}}_{\text{PPass-RC},\mathcal{A}}(\lambda)\right] \leq \mathsf{negl}(\lambda)$.

5.2 Unlinkability

Our privacy notion formalizes the *redemption context unlinkability* requirement from the IETF specification [14, section 3.3-7.4]. It

$\mathcal{O}\mathsf{Request}(vk, ctx_0, ctx_1)$
$i := i + 1$
$\overline{x}_0, aux_{0,i} \leftarrow \mathsf{Request}(vk, ctx_0)$
$\overline{x}_1, aux_{1,i} \leftarrow \mathsf{Request}(vk, ctx_1)$
return $(\overline{x}_b, \overline{x}_{1-b})$

$\mathcal{O}\mathsf{Finalize}(i, vk, \overline{y}, \overline{y}')$
$t_{b,i} \leftarrow \mathsf{Finalize}(\overline{y}, aux_{b,i})$
$t_{1-b,i} \leftarrow \mathsf{Finalize}(\overline{y}', aux_{1-b,i})$
if $\perp \in \{t_{0,i}, t_{1,i}\}$: **return** $\perp$
return $(t_{0,i}, t_{1,i})$

$\mathsf{Exp}^{\mathsf{unlink}}_{\mathsf{PPass\text{-}RC},\mathcal{A}}(\lambda)$
$b \leftarrow\!\!\$\ \{0,1\};\ i := 0;\ pp \leftarrow \mathsf{Setup}(\lambda)$
$b' \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Request},\mathcal{O}\mathsf{Finalize}}(pp);$ **return** $b' = b$

Figure 4: The unlinkability game.

requires that tokens redeemed in the redemption protocol cannot be linked to an earlier issuance session. The most interesting aspect – which is unique to the setting with a redemption context – is that the unlinkability must also hold for tokens that encode different context values. This gives a stronger unlinkability than the one achievable by the tokens with public metadata which ensures privacy only among clients that received tokens with the same metadata.

We model unlinkability via a game where the adversary selects two redemption contexts, ctx_0 and ctx_1, and receives token requests $(ireq_b, ireq_{1-b})$ generated on these contexts in a random order determined by the challenge bit b. The adversary also sees the finalized tokens (t_0, t_1) in the original order of submitted contexts (ctx_0, ctx_1), simulating the token redemption behavior in which the contexts of the tokens are revealed. We require that the adversary cannot guess the bit b better than a random guess. Otherwise, it indicates that the adversary can link a token request to its corresponding token, breaking the redemption context unlinkability.

We define unlinkability through the game Expunlink in Figure 4. The adversary therein gets access to token requesting and token finalizing oracles. $\mathcal{O}$Request takes two redemption contexts and an issuer verification key. The adversary may call $\mathcal{O}$Request with different verification keys, but the tokens in a single $\mathcal{O}$Request call are still answered for a single key to prevent trivial wins. $\mathcal{O}$Redeem outputs the blinded token requests for given contexts in a random order using the challenge bit b. Each tuple of token requests are indexed with a counter i. Later on, the adversary is allowed to call $\mathcal{O}$Finalize with an index and the token responses corresponding to this index. $\mathcal{O}$Finalize finalizes both responses with the auxiliary information matching the index, and outputs the final tokens to the adversary if both tokens are finalized successfully. If any of the tokens fail to be finalized, then $\mathcal{O}$Finalize returns $\perp$. The adversary's task in the end is guessing the challenge bit b.

Definition 5.2 (Unlinkability). A PPass-RC scheme is unlink if for all PPT adversaries $\mathcal{A}$: $\left|\Pr\left[\mathsf{Exp}^{\mathsf{unlink}}_{\mathsf{PPass\text{-}RC},\mathcal{A}}(\lambda)\right] - 1/2\right| \leq \mathsf{negl}(\lambda)$.

Comparison to Previous Models. Our game aligns with the client privacy game by Chu et al. [10] except the assumptions specific to the rate-limited Privacy Pass. Like ours, their model supports unlinkability across different redemption contexts, but requires either the issuer or attester to be honest which is a constraint stemming from the rate-limited Privacy Pass draft [18]. In contrast, Davidson et al. [13] do not support redemption context or metadata, and define unconditional unlinkability between issuance and redemption. We achieve the same goal while supporting redemption context.

Prior works on anonymous tokens with issuance metadata [4, 5, 9, 24, 28] guarantee unlinkability only among the tokens with identical metadata values, which reduces the client's anonymity set. In contrast, redemption context maintains unlinkability even with varied values, and is already part of the standard. These facts make redemption context preferable to the metadata whenever the use case is suitable e.g. token expiry.

Limitations. Unlinkability prevents tokens from being linked to their issuance even across redemption contexts, but certain application scenarios can reduce the anonymity set. For instance, in token expiry, if users always request tokens in the latest epoch, timing analysis may narrow the set of possible issuance queries. However, even under ideally performed timing attacks, redemption context offers the same level of unlinkability as public issuance metadata – demonstrating the strength of redemption context. Furthermore, performing such timing attacks is non-trivial in scenarios involving tokens issued with different contexts in parallel.

5.3 Unforgeability

This section presents the formal definitions of the two unforgeability properties that we aim for the Privacy Pass protocol. The first property is *one-more unforgeability* ensuring that a client can redeem at most as many tokens as the number of issuances they have run with the issuer. This property is motivated by the standard [14, section 3.5] and modeled by previous works on anonymous tokens [5, 8–10, 13, 24, 28]. We adapt this property to our syntax by also considering the issuer state.

The second unforgeability property is *targeted context unforgeability* as motivated in Section 3 for the token expiry and freshness applications with the redemption context. This property ensures that previously issued tokens do not help clients to forge tokens on a randomly chosen fresh redemption context. To the best of our knowledge, a security property similar to our targeted context unforgeability was not yet studied for Privacy Pass.

We show that the two unforgeability properties are incomparable, i.e., they do not imply each other. Thus, the newly proposed targeted context unforgeability is crucial to securely use the redemption context for timestamping and ensuring token freshness.

5.3.1 One-More Unforgeability. Formal definitions of one-more unforgeability of anonymous tokens are already present in the literature for publicly [4, 5, 10] and privately [9, 13, 15, 24, 28] verifiable anonymous tokens. The general game structure for both variants allows the adversary to make i token issuance queries and requires from the adversary to return $i + 1$ valid tokens. Unlike publicly verifiable anonymous tokens, privately verifiable tokens require providing an extra oracle for the token redemption as the token redemption requires the issuer secret key as input. Notably, all privately verifiable token schemes, except Davidson et al. [13], provide such an oracle in their one-more unforgeability game.

Our definition's winning condition differs from the previous ones. All previous anonymous token schemes require the adversary to return $i+1$ valid and *distinct* tokens to enforce double-spending check in the one-more unforgeability game. In contrast, our syntax allows performing double-spending check in the Redeem algorithm as we model the issuer state. Thus, instead of requiring the adversary to

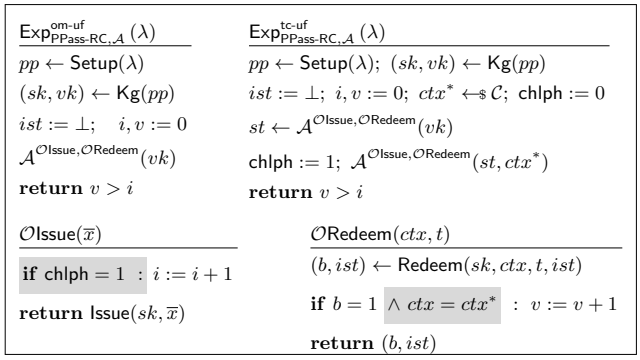


Figure 5: The one-more unforgeability (om-uf) and the targeted context unforgeability game (tc-uf). The shadowed text is only included in the tc-uf game.

output $i + 1$ distinct tokens, we employ the counter v which counts the number of the adversary's valid token redemption queries. The adversary wins if the number of valid token redemption queries exceeds the number of token issuance queries, $v > i$.

Definition 5.3 (One-more unforgeability). A PPass-RC scheme is om-uf if for all PPT adversaries $\mathcal{A}$: $\Pr\left[\mathsf{Exp}^{\text{om-uf}}_{\mathsf{PPass\text{-}RC},\mathcal{A}}(\lambda)\right] \leq \mathsf{negl}(\lambda)$.

5.3.2 Targeted Context Unforgeability. A core purpose of the redemption context is to ensure token freshness and token expiry, through assigning unpredictable context values for each epoch. By doing so, the issuer can enforce clients to initiate a fresh issuance protocol run for each redemption or for the current epoch rather than using previously issued and cached tokens.

Targeted context unforgeability captures the expected unforgeability of Privacy Pass tokens in such use cases. In particular it ensures that tokens that an attacker created before learning a new redemption context do not help it with creating a valid token for this new context. This property implicitly assumes that the context value is unpredictable, as otherwise the adversary can trivially win by guessing the value in advance.

Modeling Targeted Context Unforgeability. We model targeted context unforgeability through two phases: before and after the issuer announces a new redemption context ctx^*. The unpredictability of ctx^* is modeled by choosing it at random from the context space $\mathcal{C}$. In both phases, the adversary has access to issuance and redemption oracles, and can obtain tokens for redemption contexts of his choice (as they are fully hidden in the issuance phase anyway).

What differs in comparison to one-more unforgeability is how we count the number of non-trivial forgeries. Intuitively, the targeted context unforgeability must guarantee that the adversary cannot come up with tokens with the challenge context ctx^* more than the number of issuance queries made *after* announcing the challenge context. That is, the counter i of trivial forgeries is not increased in the first phase, but only in the second. Morevover, the game counts the number of valid redemptions with the challenge context as v. The adversary wins at the end of the game if $v > i$.

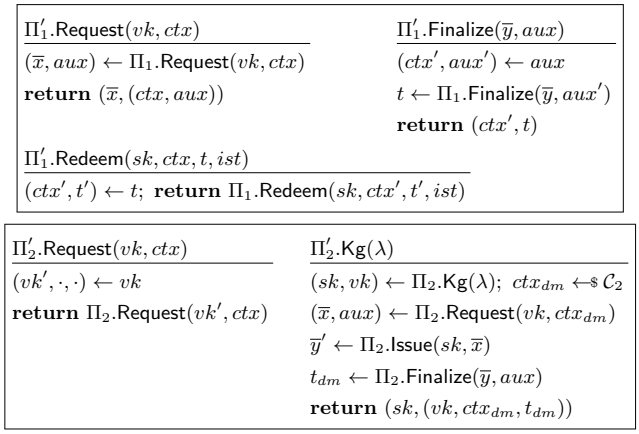


Figure 6: Top: A om-uf-secure but not tc-uf-secure scheme Π_1' (Theorem 5.5). Bottom: A tc-uf-secure but not om-uf-secure scheme Π_2' (Theorem 5.6). Missing algorithms of Π_1' and Π_2' are identical to the ones of Π_1 and Π_2.

Definition 5.4 (Targeted context unforgeability). PPass-RC is tc-uf if for all PPT adversaries $\mathcal{A}$, $\Pr\left[\mathsf{Exp}^{\text{tc-uf}}_{\mathsf{PPass\text{-}RC},\mathcal{A}}(\lambda)\right] \leq \mathsf{negl}(\lambda)$.

Note that the targeted context unforgeability can be extended to model multiple epochs with each being assigned a fresh random context by the challenger. Then the adversary can choose the targeted epoch to attack at the end of the game. Our definition can be seen as a selective version of the proposed property, with the challenge epoch fixed by the game. We opt for a simpler definition that models the essence of the needed property.

5.3.3 One-More and Targeted Context Unforgeability are Incomparable. Now we show that both unforgeability definitions, one-more unforgeability and targeted context unforgeability, are different. This justifies the need for the new targeted context unforgeability notion, and including it in the security analysis of Privacy Pass. To show this, we give two examples of Privacy Pass schemes that fulfill one property but not the other. These schemes are used in the proofs of Theorems 5.5 and 5.6 below. Note that the full proofs of the theorems are in Appendix C.

Theorem 5.5. *om-uf-security of a Privacy Pass scheme does not imply tc-uf-security of that scheme.*

To prove Theorem 5.5, we give an example of a one-more unforgeable scheme that is not targeted context unforgeable. We start with a scheme Π_1 that has one-more and targeted context unforgeability and construct a scheme Π_1' that is not targeted context unforgeable, but still satisfies one-more unforgeability. The idea is that the redemption context is ignored during redemption so that each token is accepted for all values of the context. Obviously, this scheme is not targeted context unforgeable, since an adversary can redeem any token for any redemption context. We can rely on the one-more unforgeability of Π_1 to show the one-more unforgeability of Π_1', as Π_1' only changes how redemption context is treated in Π_1, but it does not leak more Π_1 tokens than the number of issuance queries.

We now look at the other direction and show that targeted context unforgeability does not imply one-more unforgeability.

Theorem 5.6. *tc-uf-security of a Privacy Pass scheme does not imply om-uf-security of that scheme.*

Again, to prove Theorem 5.6, we give a counterexample to the implication. We start with a scheme Π_2 with targeted context and one-more unforgeability and construct a scheme Π_2' that is not one-more unforgeable. The idea here is that the Π_2' key generation extends the issuer public key with a valid token on a random context. The adversary can win the one-more unforgeability game by redeeming this token which comes for free without querying token issuance oracle. Simultaneously, the extra token on a random redemption context does not help the adversary to break the targeted context unforgeability of Π_2'. This is because the targeted context unforgeability game already allows learning tokens on arbitrary redemption contexts before the challenge context is announced.

Theorem 5.5 and Theorem 5.6 clearly show that the new property targeted context unforgeability is different from one-more unforgeability of Privacy Pass and deserves separate attention.

6 Privacy Pass from VOPRFs

We now analyze the generic VOPRF-based construction of privately verifiable Privacy Pass from the IETF specification. We first state the IETF protocol as PP-VOPRF in our syntax, and then prove that it satisfies the security properties from Section 5, assuming that the underlying VOPRF satisfies a number of natural properties, such as correctness, oblivousness, and pseudorandomness. This yields the first formal security analysis of the IETF scheme that includes the redemption context and is build generically from a VOPRF.

Before presenting PP-VOPRF and our analysis, we first recap the definition and security properties of VOPRFs.

Definition of VOPRFs. A verifiable oblivious PRF (VOPRF) is a protocol between two parties that computes a pseudorandom function (PRF). One party (the server) provides the secret PRF key and the other party (the client) provides the input and receives the output. We follow the definitions of Tyagi et al. [30] adapted to our notation. We denote the input and output spaces of the underlying PRF function, VOPRF.Eval, of a VOPRF as $\mathcal{X}$ and $\mathcal{Y}$, respectively. We first give the syntax of VOPRFs formally.

Definition 6.1 (VOPRF). A VOPRF is a tuple of algorithms s.t.:

$\mathsf{Setup}(\lambda) \to pp$**:** Outputs the public parameters for an input security parameter.

$\mathsf{Kg}\,(pp) \to (sk, pk)$**:** Outputs a key pair (sk, pk).

$\mathsf{Eval}(sk, x) \to y$**:** A deterministic algorithm which evaluates the PRF with the secret key sk on $x \in \mathcal{X}$ and output is as $y \in \mathcal{Y}$.

$\mathsf{Blind}(pk, x) \to (\overline{x}, aux)$**:** A blinding algorithm that blinds the value x and also outputs auxiliary state information aux.

$\mathsf{BlindEval}(sk, \overline{x}) \to \overline{y}$**:** Given a blinded input $\overline{x}$ and a secret key sk, outputs the blinded evaluation $\overline{y}$.

$\mathsf{Unblind}(\overline{y}, aux) \to y/\perp$**:** Unblinds the blinded evaluation $\overline{y}$, outputs $y \in \mathcal{Y}$ on success and $\perp$ on failure.

Next we define the correctness of a VOPRF.

Definition 6.2 (VOPRF Correctness). A VOPRF VOPRF is correct if for all λ, $pp \leftarrow \mathsf{Setup}(\lambda)$, $(sk, pk) \leftarrow \mathsf{VOPRF.Kg}(pp)$, $x \in \mathcal{X}$, $\overline{x}, aux \leftarrow \mathsf{Blind}(pk, x)$, and $y \leftarrow \mathsf{Unblind}(\mathsf{BlindEval}(sk, \overline{x}), aux)$: $\Pr[\mathsf{Eval}(sk, x) = y] = 1$.

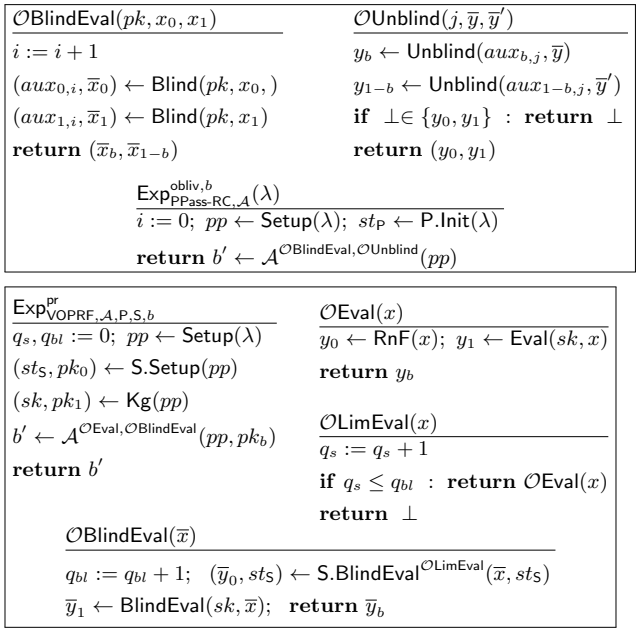


Figure 7: Top: VOPRF obliviousness game. Bottom: VOPRF pseudorandomness game.

We define VOPRF obliviousness, ensuring that neither the client's input nor the output is leaked to the server.

Definition 6.3 (VOPRF Obliviousness). A VOPRF is oblivious if for all PPT adversaries $\mathcal{A}$, for the experiment given in Figure 7, $\left|\Pr\left[\mathsf{Exp}^{\mathsf{obliv},0}_{\mathcal{A},\mathsf{VOPRF}}(\lambda)\right] - \Pr\left[\mathsf{Exp}^{\mathsf{obliv},1}_{\mathcal{A},\mathsf{VOPRF}}(\lambda)\right]\right| \le \mathsf{negl}(\lambda)$.

Finally, we define pseudorandomness of VOPRFs. Following Tyagi et al., our definition uses a random function generator $\mathsf{RnF} \leftarrow \$\mathsf{FnGen}(\mathcal{X}, \mathcal{Y})$ that samples a random function $\mathsf{RnF} : \mathcal{X} \to \mathcal{Y}$. We omit the ideal primitive that Tyagi et al. defined to model random oracles explicitly, for simplicity.

Definition 6.4 (VOPRF Pseudorandomness). A VOPRF is pseudorandom if for given a simulator S and the experiment in Figure 7, and for all PPT adversaries $\mathcal{A}$,

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{pr},0,\mathsf{S}}_{\mathcal{A},\mathsf{VOPRF}}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{pr},1,\mathsf{S}}_{\mathcal{A},\mathsf{VOPRF}}(\lambda) = 1\right]\right| \le \mathsf{negl}(\lambda).$$

6.1 Construction PP-VOPRF

We now state the privately verifiable Privacy Pass construction PP-VOPRF from the IETF specification [8, 26]. The construction builds upon a VOPRF as defined above. PP-VOPRF runs the VOPRF evaluation to let the client blindly obtain PRF values evaluated under the issuer's secret key. The client embeds both the redemption context and a random nonce in the requested value. Redemption essentially checks that the client has sent a correct in/output pair, which requires access to the issuer's secret key.

The detailed construction is given in Fig. 8 and described below. Note that we do not fix a specific context space C for PP-VOPRF, but C must have super-polynomial size to the security parameter to satisfy targeted context unforgeability.

Privately Verifiable Privacy Pass from VOPRF. The setup and issuer key generation algorithms of PP-VOPRF are identical to the VOPRF setup and key generation algorithms.

Token Issuance. This algorithm aims to create a token on the redemption context ctx and a client-chosen random nonce noc. Thus, the client blinds $\mathsf{H}_{dg}(ctx)\|noc$ using the VOPRF.Blind algorithm.

The token issuance simply runs the blind VOPRF evaluation algorithm VOPRF.BlindEval on the client's blinded token request and outputs the blinded VOPRF evaluation $\overline{y}$. To finalize the token creation, the client unblinds $\overline{y}$ as y. If the VOPRF unblinding algorithm outputs $\perp$, then the client also outputs $\perp$. Otherwise, the client sets the final token as (dg, noc, y).

The token redemption checks whether dg corresponds to the input context ctx, the token contains the valid VOPRF evaluation y for $dg\|noc$, and noc is not in the issuer state ist. Note that the redemption relies on the non-blind PRF evaluation algorithm VOPRF.Eval, as it already knows the inputs to the PRF. If they all hold, the issuer adds noc to its state for double-spending protection and accepts the token. Otherwise, it rejects the token.

The finalized token explicitly contains dg which is the hash of the targeted redemption context. We note that this is not necessary, as dg can be re-computed inside Redeem by using the input ctx – but we keep it for consistency with the standard.

Differences From the IETF Standard. The construction presented here is slightly different from the one in the IETF specification.

First, the Redeem algorithm employs a hardcoded double-spending protection through the if-else statement. The specification only describes informally [14, section 7.1-2] that there should be a double-spending check against replay attacks. For PP-VOPRF, storing the nonces from the tokens is sufficient. Second, the specification fixes the hash function H_{dg} to SHA256, but our proof of targeted context unforgeability models H_{dg} as a random oracle.

6.2 Security of PP-VOPRF

We show that PP-VOPRF from Figure 8 satisfies all desired security and privacy properties. In a nutshell, the unlinkability and unforgeability of PP-VOPRF are ensured by the obliviousness and pseudorandomness of the underlying VOPRF, respectively. We only give proof sketches here and defer the full proofs to Appendix D.

6.2.1 Redemption Robustness. We show the redemption robustness of PP-VOPRF, assuming that the underlying VOPRF is correct.

Theorem 6.5. PP-VOPRF *is red-rob if* VOPRF *is correct.*

Proof sketch. A token is valid when it contains a correct VOPRF evaluation y on $dg\|noc$ and noc is not blacklisted by ist. The former holds by relying on the VOPRF correctness and the latter holds because the randomly chosen noc by the honest client colludes with the ones in ist only with a negligible probability.

6.2.2 Unlinkability. PP-VOPRF's unlinkability easily follows from the obliviousness of the underlying VOPRF. We prove Theorem 6.6 using a reduction from VOPRF obliviousness which ensures that the blinded input $\overline{x}$ leaks no information about its input x, implying that the token request reveals no information about the $dg\|noc$.

Theorem 6.6. PP-VOPRF *is unlink if* VOPRF *is oblivious.*

Proof sketch. The obliviousness game for VOPRFs and the unlinkability game for Privacy Pass are similar, making the reduction straightforward. Intuitively, we build an adversary $\mathcal{B}$ which can distinguish between blinded VOPRF inputs on $dg_0\|noc_0$ and $dg_1\|noc_1$ by using an adversary $\mathcal{A}$ that can distinguish between the PP-VOPRF token requests for ctx_0 and ctx_1. The only extra step that $\mathcal{B}$ takes is computing the appropriate VOPRF input values $x := noc_i\|\mathsf{H}_{dg}(ctx_i)$ for random nonces noc_i and given ctx_i values where $i \in \{0, 1\}$. Otherwise, $\mathcal{B}$ simply echoes messages between $\mathcal{A}$ and the VOPRF obliviousness challenger.

6.2.3 One-More Unforgeability. We show the one-more unforgeability of PP-VOPRF by relying on the pseudorandomness of the underlying VOPRF.

Theorem 6.7. PP-VOPRF *is om-uf if* VOPRF *is pseudorandom, and the output space size of* VOPRF $|\mathcal{Y}|$ *is super-polynomial to* λ.

Proof sketch. Our proof first switches to a game that all VOPRF evaluations are simulated with a truly random function and the VOPRF simulator which is indistinguishable if the pseudorandomness of the VOPRF holds. Then we argue that the adversary cannot learn more evaluations of the random function than it queries the token issuance oracle by relying on the behavior of the VOPRF simulator. This leads us to our final argument that the adversary's advantage to forge a token is not better than guessing the output of the random function on a fresh point correctly. As long as the output space $\mathcal{Y}$ is *large* enough, this is negligible.

A weaker VOPRF property, one-more unpredictability [7, 17, 25], allows the adversary to interact with a VOPRF evaluation algorithm i times and requires the adversary to return correct evaluations on $i + 1$ distinct messages. While this property may seem sufficient for showing one-more unforgeability of PP-VOPRF, it is not. Specifically, PPass-RC unforgeability games permit an adversary to use a redemption oracle, allowing unlimited non-blind PRF evaluations, which one-more unpredictability games do not support.

6.2.4 Targeted Context Unforgeability. Now we show the targeted context unforgeability of PP-VOPRF, again by relying on the pseudorandomness of the underlying VOPRF.

Theorem 6.8. PP-VOPRF *is tc-uf in ROM for* H_{dg} *is a random oracle if* VOPRF *is pseudorandom, and the sizes of the context space and the* VOPRF *output space* $|\mathcal{C}|$ *and* $|\mathcal{Y}|$ *are super polynomial to* λ.

Proof sketch. The proof strategy for Theorem 6.8 follows the approach of Theorem 6.7, with additional steps. We first transition to a truly random function via a game hop Game_1, relying on the pseudorandomness of the VOPRF. To show that the adversary must guess the function's output for some input, we ensure that the issuance queries prior to the challenge context's announcement do not reveal VOPRF evaluations on inputs involving the challenge context. This is done through further game hops that guarantee:

(1) The adversary does not call $\mathcal{O}$Redeem on the challenge context ctx^* before ctx^* is revealed, and
(2) No collision occurs in outputs of the random oracle H_{dg}.

After ensuring these two conditions, the only way for an adversary to learn a VOPRF evaluation on a random context is luckily guessing ctx^* and making a blind issuance query for it. We eliminate this case by relying on the properties of the VOPRF simulator. Notice that we

$\mathsf{Request}(vk, ctx)$	$\mathsf{Finalize}(\overline{y}, aux)$	$\mathsf{Redeem}(sk, ctx, t, ist)$
$noc \leftarrow\!\!\$\ \{0,1\}^\lambda;\ dg \leftarrow \mathsf{H}_{dg}(ctx)$	$(dg, noc, aux') \leftarrow aux$	$(dg, noc, y) \leftarrow t;\ x \leftarrow dg \Vert noc;$ **if** $ist = \bot$: $ist \leftarrow \emptyset$
$\overline{x}, aux \leftarrow \mathsf{VOPRF.Blind}(dg \Vert noc)$	$y \leftarrow \mathsf{VOPRF.Unblind}(\overline{y}, aux')$	**if** $dg = \mathsf{H}_{dg}(ctx) \wedge noc \notin ist \wedge \mathsf{VOPRF.Eval}(sk, x) = y$:
return $(\overline{x}, (dg, noc, aux))$	**if** $y = \bot$: **return** $\bot$	$ist \leftarrow ist \cup \{noc\};$ **return** $(1, ist)$
	return (dg, noc, y)	**else** : **return** $(0, ist)$

Figure 8: PPass-RC construction PP-VOPRF **from the IETF specification [8, 26].** Setup**,** Kg**, and** Issue **algorithms of** PP-VOPRF **are identical to the** VOPRF.Setup**,** VOPRF.Kg**, and** VOPRF.BlindEval**, respectively.** H_{dg} **is a hash function over** $\mathsf{H}_{dg}: \{0,1\}^* \rightarrow \{0,1\}^\lambda$**.**

can have an exact list of leaked VOPRF evaluations to the adversary through book-keeping the $\mathcal{O}$LimEval queries of the simulator S. This allows us to argue that each of these leaked evaluations can contain $\mathsf{H}_{dg}(ctx^*)$ only with a negligible probability.

These steps allow us to defer sampling ctx^* to the start of the second game phase, and no $\mathcal{O}$Issue queries relate to ctx^* beforehand. The rest of the proof mirrors the one-more unforgeability argument: since the adversary must produce a forgery beyond the number of issuance queries made after ctx^* is known, their success probability is bounded by the chance of correctly guessing the output of a random function on an unseen input. This concludes our proof.

Necessity of ROM. One might ask if the ROM is necessary, or if collision resistance of H_{dg} would suffice. Due to a subtle issue, it does not. We require that $\mathsf{H}_{dg}(ctx^*)$ does not collide with $\mathsf{H}_{dg}(ctx')$ for any ctx' used in $\mathcal{O}$Issue queries before ctx^* is known. We leverage this statement while ensuring that the adversary learns no VOPRF output related to ctx^*. However, even if the adversary finds such a collision, it does not have to use ctx' in a $\mathcal{O}$Redeem query, which prevents a reduction to standard collision resistance by finding the tuple (ctx^*, ctx'). In contrast, ROM allows us to statistically ensure no such collisions occur, which is why we use it in our proof.

6.2.5 Instantiating VOPRF*.* The IETF specification builds on RFC 9497, the IRTF's VOPRF standard [11]. While both our analysis and the IETF specification are compatible with alternative instantiations, RFC 9497 currently supports the 2HashDH construction with flexible output space, originally proposed by Jarecki et al. [23]. The IETF specification uses this construction with output space fixed as $\mathcal{Y} := \{0,1\}^{384}$. Tyagi et al. [30] further prove that 2HashDH satisfies the necessary game-based security properties for PP-VOPRF. We refer to [23, 30] for the corresponding proofs.

Corollary 6.9. 2HashDH *[23, 30] is a perfectly correct, pseudorandom, and oblivious* VOPRF *scheme.*

7 Conclusion

Our work has formalized the security and privacy properties of the IETF Privacy Pass protocol, and has proven that they can be securely realized from VOPRFs. One of the core observations was the impact the redemption context has on these properties, as this feature has not been captured in a formal analysis yet.

Redemption Context for Better Privacy. We show that the redemption context that is already included in the IETF specification, offers a more privacy-preserving alternative for encoding expiry information than public metadata. The latter has been the focus of academic works [4, 9, 28] and is currently proposed as an extension for enforcing token expiry [20–22]. Encoding expiry dates as public metadata immediately reduces the anonymity set size at issuance, as issuance and redemption can be linked through that added information.

Redemption context avoids this linkage and can provide the same functionality with better privacy. Still, one should consider the redemption context's impact on the other protocol layers before using it in real-world. For instance, while redemption context ensures unlinkability for token freshness in the cryptographic layer, timing attacks on the freshly issued tokens can still reduce the anonymity sets. Thus, applications must carefully design and restrict allowed redemption contexts, e.g. using a *verifiable random function* (VRF) to set redemption contexts instead of a PRF as suggested in the specification [26, section 2.1.1.2]. This would allow clients verify that a redemption context satisfies a pre-defined restriction, such as a token expiry, by checking it corresponds to certain timestamp. As a result, evaluating the redemption context's implications beyond the cryptographic layer is crucial.

Anonymous Tokens with Redemption Context. Embedding blind information in the issuance phase of anonymous tokens that is enforced to have some context and meaning for the application is an interesting concept beyond the privately-verifiable Privacy Pass protocol. This concept is applicable to all types of anonymous tokens, ranging from VOPRFs [13], blind MACs [9, 15], to blind signatures [4, 10], and enables a more privacy-preserving way to encode context information in these primitives too. They can all be adapted to support redemption contexts by appending them to the signed nonce, as in PP-VOPRF. In particular, this can also be used in the publicly verifiable variant of the IETF specification which relies on RSA blind signatures.

Publicly Verifiable Privacy Pass with Redemption Context. When using such a redemption context, our introduced property of targeted context unforgeability must be adapted and analysed for the public verifiable setting too. However, targeted context unforgeability of these schemes will be more challenging to prove. Existing publicly-verifiable Privacy Pass schemes aim solely at one-more unforgeability, which is a well-established property that RSA blind signatures achieve. However, to prove targeted context unforgeability in PP-VOPRF we required a stronger notion: pseudorandomness. Intuitively, pseudorandomness allows a reduction to check leaked evaluations of the random function at any point of the game, allowing us to detect whether a ctx^* evaluation was leaked before ctx^* was revealed. It is uncertain how one-more unforgeability can ensure this. Blind signatures lack a default pseudorandomness notion.

Establishing similar guarantees for constructions based on other primitives is an open and interesting research direction. Establishing similar guarantees for constructions based on other primitives is an open and interesting research direction.

Acknowledgments

This research was partially funded by the HPI Research School on Systems Design. It was also supported by the German Federal Ministry of Education and Research (BMBF) through funding of the ATLAS project under reference number 16KISA037.

References

[1] 2020. A privacy-enhancing protocol and browser extension. https://privacypass.github.io/. [Accessed 15-04-2025].
[2] 2021. DIT - enabling de-identified data collection on WhatsApp. https://engineering.fb.com/2021/04/16/security/dit/. [Accessed 15-04-2025].
[3] 2025. Private State Tokens. https://developers.google.com/privacy-sandbox/protections/private-state-tokens/. [Accessed 15-04-2025].
[4] Ghous Amjad, Kevin Yeo, and Moti Yung. 2025. RSA Blind Signatures with Public Metadata. *Proceedings on Privacy Enhancing Technologies* (2025).
[5] Fabrice Benhamouda, Tancrede Lepoint, Michele Orru, and Mariana Raykova. 2022. Publicly verifiable anonymous tokens with private metadata bit. *Cryptology ePrint Archive* (2022).
[6] Ward Beullens, Lucas Dodgson, Sebastian Faller, and Julia Hesse. 2025. The 2Hash OPRF Framework and Efficient Post-quantum Instantiations. In *Advances in Cryptology – EUROCRYPT 2025: 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4–8, 2025, Proceedings, Part VIII* (Madrid, Spain). Springer-Verlag, Berlin, Heidelberg, 332–362.
[7] Silvia Casacuberta, Julia Hesse, and Anja Lehmann. 2022. SoK: Oblivious Pseudorandom Functions. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*. 625–646.
[8] Sofia Celi, Alex Davidson, Steven Valdez, and Christopher A. Wood. 2024. Privacy Pass Issuance Protocols. RFC 9578. doi:10.17487/RFC9578
[9] Melissa Chase, F. Betül Durak, and Serge Vaudenay. 2023. Anonymous Tokens with Stronger Metadata Bit Hiding from Algebraic MACs. In *Advances in Cryptology – CRYPTO 2023*, Helena Handschuh and Anna Lysyanskaya (Eds.). Springer Nature Switzerland, Cham, 418–449.
[10] Hien Chu, Khue Do, and Lucjan Hanzlik. 2023. On the Security of Rate-limited Privacy Pass. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 2871–2885.
[11] Alex Davidson, Armando Faz-Hernandez, Nick Sullivan, and Christopher A. Wood. 2023. Oblivious Pseudorandom Functions (OPRFs) Using Prime-Order Groups. RFC 9497. doi:10.17487/RFC9497
[12] Alex Davidson, Matthew Finkel, Martin Thomson, and Christopher A. Wood. 2023. *Key Consistency and Discovery.* Internet-Draft draft-ietf-privacypass-key-consistency-01. Internet Engineering Task Force. Work in Progress.
[13] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. 2018. Privacy Pass: Bypassing Internet Challenges Anonymously. *Proceedings on Privacy Enhancing Technologies* 2018, 3 (2018), 164–180.
[14] Alex Davidson, Jana Iyengar, and Christopher A. Wood. 2024. The Privacy Pass Architecture. RFC 9576. doi:10.17487/RFC9576
[15] F. Betül Durak, Laurane Marco, Abdullah Talayhan, and Serge Vaudenay. 2024. Non-Transferable Anonymous Tokens by Secret Binding. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security (CCS '24)*. Association for Computing Machinery, New York, NY, USA, 2460–2474.
[16] Muhammed F. Esgin, Ron Steinfeld, Erkan Tairi, and Jie Xu. 2024. LeOPaRd: Towards Practical Post-Quantum Oblivious PRFs via Interactive Lattice Problems. https://eprint.iacr.org/2024/1615 Publication info: Preprint..
[17] Adam Everspaugh, Rahul Chaterjee, Samuel Scott, Ari Juels, and Thomas Ristenpart. 2015. The pythia {PRF} service. In *24th USENIX Security Symposium (USENIX Security 15)*. 547–562.
[18] Scott Hendrickson, Jana Iyengar, Tommy Pauly, Steven Valdez, and Christopher A. Wood. 2024. *Rate-Limited Token Issuance Protocol.* Internet-Draft draft-ietf-privacypass-rate-limit-tokens-06. Internet Engineering Task Force. Work in Progress.
[19] Scott Hendrickson and Thibault Meunier. 2024. *Attester Issuer Protocol.* Internet-Draft draft-hendrickson-pp-attesterissuer-00. Internet Engineering Task Force. Work in Progress.
[20] Scott Hendrickson and Christopher A. Wood. 2024. *Privacy Pass Issuance Protocols with Public Metadata.* Internet-Draft draft-ietf-privacypass-public-metadata-issuance-01. Internet Engineering Task Force. Work in Progress.
[21] Scott Hendrickson and Christopher A. Wood. 2024. *The PrivateToken HTTP Authentication Scheme Extensions Parameter.* Internet-Draft draft-ietf-privacypass-auth-scheme-extensions-01. Internet Engineering Task Force. Work in Progress.
[22] Scott Hendrickson and Christopher A. Wood. 2025. *Privacy Pass Token Expiration Extension.* Internet-Draft draft-hendrickson-privacypass-expiration-extension-03. Internet Engineering Task Force. Work in Progress.
[23] Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. 2014. Round-Optimal Password-Protected Secret Sharing and T-PAKE in the Password-Only Model. In *Advances in Cryptology – ASIACRYPT 2014*, Palash Sarkar and Tetsu Iwata (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 233–253.
[24] Ben Kreuter, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. 2020. Anonymous tokens with private metadata bit. In *Advances in Cryptology–CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part I 40*. Springer, 308–336.
[25] Anja Lehmann. 2019. ScrambleDB: Oblivious (Chameleon) Pseudonymization-as-a-Service. *Proceedings on Privacy Enhancing Technologies* (2019).
[26] Tommy Pauly, Steven Valdez, and Christopher A. Wood. 2024. The Privacy Pass HTTP Authentication Scheme. RFC 9577. doi:10.17487/RFC9577
[27] Raphael Robert, Christopher A. Wood, and Thibault Meunier. 2025. *Batched Token Issuance Protocol.* Internet-Draft draft-ietf-privacypass-batched-tokens-04. Internet Engineering Task Force. Work in Progress.
[28] Tjerand Silde and Martin Strand. 2022. Anonymous Tokens with Public Metadata and Applications to Private Contact Tracing. In *Financial Cryptography and Data Security*. Springer International Publishing, Cham, 179–199.
[29] Reid Tatoris and Maxime Guerreiro. 2022. Private Access Tokens: eliminating CAPTCHAs on iPhones and Macs with open standards. https://blog.cloudflare.com/eliminating-captchas-on-iphones-and-macs-using-new-standard/. [Accessed 15-04-2025].
[30] Nirvan Tyagi, Sofia Celi, Thomas Ristenpart, Nick Sullivan, Stefano Tessaro, and Christopher A Wood. 2022. A fast and simple partially oblivious PRF, with applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 674–705.
[31] Christopher A. Wood. 2023. Not-So-Low Hanging Fruit: Security and Privacy Research Opportunities for IETF Protocols. In *Proceedings of the Applied Networking Research Workshop*. ACM, San Francisco CA USA, 41–43.

A Ongoing Internet Drafts on Privacy Pass

Currently, there are further internet drafts which could become part of the Privacy Pass IETF specification. We point at these drafts to show the position and the future direction of the IETF specification.

Although the initial use case of Privacy Pass assumes that tokens will be issued in a batched way to be redeemed later, the current IETF specification only considers a single token issuance per each interaction with the issuer and the attester. Batched token issuance protocol [27] extends the protocol accordingly to fill this gap. Public issuance metadata was studied extensively by the literature, and two ongoing internet drafts [20, 21] aim to extend the IETF specification for public metadata support. Another draft considers a specific use case of public metadata to token expiry [22]. Rate-limited Privacy Pass, which was formally analyzed by Chu et al. [10], is another draft [18] that extends the publicly verifiable Privacy Pass. The internet draft on the attester issuer protocol [19] specifies a privacy preserving method for the attestation process so that the token issuance and attestation cannot be linked to each other.

B Unlinkability Requirements of the Standard

We explain the other privacy requirements that were noted by the standard and evaluate their relavance to the cryptographic protocol of privately verifable Privacy Pass.

Attester-Origin Unlinkability. It requires that from the interactions during the attestation procedure with two clients, the attester cannot determine if both interactions correspond to the same origin or two different origins [14, section 3.3-7.3]. In the cryptographic protocol level, this requirement is not meaningful for the IETF standard as attesters do not join the cryptographic protocol.

Attester-origin unlinkability is instead important in rate-limited Privacy Pass. This is an extension to publicly verifiable Privacy Pass, where the attester rate-limits issuance requests from specific clients while not learning the origin for whom the tokens are meant [18]. Chu et al. [10] analyze the security of this extension. They formalize attester-origin unlinkability under the name origin privacy.

Origin-Client Unlinkability. This property means that the origin (also the issuer in our case) cannot distinguish between clients based on the redemption of tokens with the same context [14, section 3.3]. The IETF standard's cryptographic protocol binds neither client-specific information to tokens nor long-term identifiers to clients. In practice, however, clients may have state, e.g., caching tokens and redeeming them later. Privacy can be lost if clients use unusual redemption orders such as lexicographic order. Davidson et al. [13] model client state by assuming tokens are redeemed in issuance order, for which origin-client privacy follows directly from their unlinkability property or from our redemption-context unlinkability, as both hide the issuance order. We omit this property, as it is already ensured by the redemption-context unlinkability.

C Full Proofs of Theorems 5.5 and 5.6

C.1 Proof of Theorem 5.5

Proof. Let $\Pi_1 := (\mathsf{Setup}, \mathsf{Kg}, \mathsf{Request}, \mathsf{Issue}, \mathsf{Finalize}, \mathsf{Redeem})$ be a one-more unforgeable Privacy Pass scheme with redemption robustness. We define $\Pi_1' := (\Pi_1.\mathsf{Setup}, \Pi_1.\mathsf{Kg}, \Pi_1'.\mathsf{Request}, \Pi_1.\mathsf{Issue}, \Pi_1'.\mathsf{Finalize}, \Pi_1'.\mathsf{Redeem})$ with $\Pi_1'.\mathsf{Request}$, $\Pi_1'.\mathsf{Finalize}$, and $\Pi_1'.\mathsf{Redeem}$ as shown in Figure 6.

Π_1' is Redemption Robust. The new scheme Π_1' has redemption robustness. This can be shown by relying on the redemption robustness of Π_1 with a reduction as follows. The reduction forwards the token issuance queries for the Π_1' to the Π_1 challenger. For the token redemption queries, the reduction must parse t as $t := (ctx', t')$ and query the Π_1 challenger's redemption oracle for $O\mathsf{Redeem}(ctx', t')$. Finally, when the Π_1' adversary outputs a context ctx^*, the reduction forwards ctx^* to the Π_1 challenger. For an honestly computed token, the token redemption of Π_1 and Π_1' are identical. Furthermore, the states of the two schemes are identical. Thus, the honest token issuance and redemption of the Π_1 challenger perfectly simulates the honest token issuance and redemption of Π_1'. Hence, if the adversary breaks the redemption robustness of Π_1', then the reduction breaks the redemption robustness of Π_1.

Π_1' is One-More Unforgeable. Furthermore, Π_1' is one-more unforgeable. Similar to the redemptions robustness proof, we can build a reduction to the one-more unforgeability of Π_1 that forwards the issuance queries of Π_1' adversary as they are, and forwarding the redemption queries as $O\mathsf{Redeem}(ctx', t')$ after parsing the input redemption token as $t := (ctx', t')$. The responses of the Π_1 one-more unforgeability challenger perfectly simulates the responses of the Π_1' challenger. Thus, given an efficient Π_1' om-uf adversary, the reduction breaks the one-more unforgeability of Π_1.

Π_1' is not Targeted Context Unforgeable. Finally, we show an attack against the targeted context unforgeability of Π_1'. After obtaining a token t for an arbitrary context ctx from honest issuance, an adversary asks for the challenge context ctx^*. Upon learning the challenge context, the adversary queries the redemption oracle for (t, ctx^*). It is accepted as a valid token since the $\Pi_1'.\mathsf{Redeem}$ token redemption ignores the input redemption context. Then the adversary finalizes and breaks the targeted context unforgeability of Π_1' as it forged a valid token for ctx^* without making any queries after the challenge context announcement.

This concludes the proof and shows that om-uf-security of a Privacy Pass scheme does not imply its tc-uf-security. □

C.2 Proof of Theorem 5.6

Proof. Let $\Pi_2 := (\mathsf{Setup}, \mathsf{Kg}, \mathsf{Request}, \mathsf{Issue}, \mathsf{Finalize}, \mathsf{Redeem})$ be a targeted context unforgeable Privacy Pass scheme with redemption robustness. We define $\Pi_2' := (\Pi_2.\mathsf{Setup}, \Pi_2'.\mathsf{Kg}, \Pi_2'.\mathsf{Request}, \Pi_2.\mathsf{Issue}, \Pi_2.\mathsf{Finalize}, \Pi_2.\mathsf{Redeem})$ with $\Pi_2'.\mathsf{Kg}$, $\Pi_2'.\mathsf{Request}$, and $\Pi_2'.\mathsf{Finalize}$ as defined in in Figure 6. Π_2' is identical to Π_2, except that the issuance public key also contains an honestly issued dummy token t_{dm} for a dummy redemption context ctx_{dm}. These values in the extended public key are discarded in the $\Pi_2'.\mathsf{Request}$ algorithm.

Π_2' is Redemption Robust. We show the redemption robustness of Π_2' by relying on the redemption robustness of Π_2. The reduction between two can get the Π_2 public key, sample the dummy context ctx_{dm}, and get a valid token on it using the token issuance oracle. Then these value can be forwarded to the Π_2' adversary as the Π_2' public key. Observe that the issuer-side algorithms of Π_2' other than the key generation is identical to Π_2 algorithms. Thus, the adversary's queries can be forwarded to the Π_2 challenger by the reduction, and an honestly issued Π_2' token can be rejected only if the same token is rejected by the Π_2 redemption on the same state. Thus, we conclude that Π_2' satisfies redemption robustness.

Π_2' is Targeted Context Unforgeable. Similarly, Π_2' is targeted context unforgeable because an additional honest token does not help to break targeted context unforgeability with overwhelming probability. A simple reduction to the targeted context unforgeability of Π_2 can make an internal issuance query to the Π_2 challenger to simulate the dummy context and token in the Π_2' public key, ctx_{dm} and t_{dm}. The token issuance and redemption queries of the adversary can be forwarded to the Π_2 challenger. Notice that the initial issuance query of the reduction does not increment the counter i as it is made before the announcement of the challenge context. Thus, whenever the adversary breaks Π_2' targeted context unforgeability, then the reduction breaks Π_2 targeted context unforgeability.

Π_2' is not One-More Unforgeable. Π_2' is not one-more unforgeable because an adversary can redeem the dummy token in the public key without making any issuance oracle queries. If Π_2 is redemption robust, this query must be accepted by the redemption oracle and the adversary breaks the one-more unforgeability of Π_2'.

As a result, we conclude that the tc-uf-security of a Privacy Pass scheme does not imply its om-uf-security. □

D Full Proofs of Section 6

This section gives the full proofs of Section 6. In the proofs with game hops, W_i denotes the event that the adversary wins Game_i. Game_0 is identical to the original game that is analyzed in the proof.

D.1 Proof of Theorem 6.5

Proof. Let VOPRF be a correct VOPRF and let $\mathcal{A}$ be a PPT adversary in the Expred-rob$[\mathcal{A}][\text{PP-VOPRF}](\lambda)$ game. Let q be the number of queries to the redemption oracle $\mathcal{O}$Redeem made by $\mathcal{A}$ during the game. After $\mathcal{A}$ terminates with output ctx, for q_{red} being the number of redemption oracle queries, $|ist| \leq q_{red}$ as each $\mathcal{O}$Redeem query adds at most one element to ist.

Now, consider the token $t = (dg, noc, y)$ that is the result of an honest issuance. We argue that $\mathcal{O}\text{Redeem}(ctx, t)$ returns 0 with only negligible probability. By construction, $dg = \mathsf{H}_{dg}(ctx)$, so condition (1) in PP-VOPRF.Redeem is true. Because VOPRF is correct, $\text{VOPRF.Eval}(sk, dg \| noc) = y$, so condition (2) is true. Finally, since noc is chosen randomly from $\{0,1\}^\lambda$ and ist contains at most q_{red} distinct values, the probability that $noc \in state$ is at most $q_{red}/2^\lambda$ which is negligible. Thus, for a perfectly correct VOPRF, we conclude that, $\mathsf{Adv}^{\text{red-rob}}_{\mathcal{A},\text{PP-VOPRF}}(\lambda) \leq q_{red}/2^\lambda$. □

D.2 Proof of Theorem 6.6

Proof. Let $\mathcal{A}$ be a PPT adversary in the $\mathsf{Exp}^{\text{unlink}}_{\mathcal{A},\text{PP-VOPRF}}(\lambda)$ game. We define adversary $\mathcal{B}$ as follows. $\mathcal{B}$ receives the public parameters from the obliviousness challenger and forwards them to $\mathcal{A}$ as the public parameters of PP-VOPRF.

For the ith request oracle query $\mathcal{O}\text{Request}(vk, ctx_0, ctx_1)$, $\mathcal{B}$ samples the random nonces $noc_{0,i}, noc_{1,i} \leftarrow\!\$ \{0,1\}^\lambda$, and computes the VOPRF inputs $(x_0, x_1) := (noc_{0,i} \| \mathsf{H}_{dg}(ctx_0), noc_{1,i} \| \mathsf{H}_{dg}(ctx_1))$. Then $\mathcal{B}$ calls $(\overline{y}_b, \overline{y}_{1-b}) \leftarrow \mathcal{O}\text{Blind}(vk, x_0, x_1)$ and it returns the blinded requests $(\overline{x}_b, \overline{x}_{1-b})$ to $\mathcal{A}$.

For jth finalize oracle query $\mathcal{O}\text{Finalize}(j, \overline{y}, \overline{y}')$, $\mathcal{B}$ immediately calls $res \leftarrow \mathcal{O}\text{Unblind}(j, \overline{y}, \overline{y}')$ and return res to $\mathcal{A}$. Observe that $\mathcal{O}$Blind identically simulates $\mathcal{O}$Finalize.

Finally, $\mathcal{B}$ gets $\mathcal{A}$'s guess for the challenge bit b' and forwards it to the obliviousness challenger. $\mathcal{B}$ wins whenever $\mathcal{A}$ wins. □

D.3 Proof of Theorem 6.7

Proof. We present a sequence of games to show the one-more unforgeability of PPass-RC.

$\underline{\text{Game}_1}$: This game changes the VOPRF evaluations with truly random values by using the pseudorandomness property of VOPRF. In particular, instead of running VOPRF.Kg, Game_1 runs $(st_S, pk) \leftarrow \text{S.Setup}(pp)$ to set the challenge public key pk and samples a truly random function RnF. In the blind issuance oracle, instead of VOPRF.BlindEval, we run the simulator $(\overline{y}, st_S) \leftarrow \text{S.BlindEval}(\overline{x}, st_S)$ and simulate the $\mathcal{O}$LimEval queries of the simulator using the random function RnF. Finally, in the token redemption oracle, instead of computing $\text{VOPRF.Eval}(sk, dg \| noc)$, we run $\mathsf{RnF}(dg \| noc)$.

It is straightforward to observe that Game_0 simulates the pseudorandomness game for $b = 1$ and Game_1 simulates it for $b = 0$. Thus, using a distinguisher $\mathcal{D}$ between games Game_0 and Game_1, we can build a pseudorandomness adversary $\mathcal{B}$ against VOPRF. Namely, $\mathcal{B}$ uses the pseudorandomness challenge public key as the issuer public key against $\mathcal{D}$. Then $\mathcal{B}$ simulates token issuance queries by using $\mathcal{O}$.BlindEval oracle and the token redemption queries by using the $\mathcal{O}$.Eval oracle of the pseudorandomness game. If $b = 1$, then $\mathcal{B}$ plays Game_0 aginst $\mathcal{D}$, and if $b = 0$, $\mathcal{B}$ plays Game_1 against $\mathcal{D}$. Thus, $|\Pr[W_0] - \Pr[W_1]| \leq \mathsf{Adv}^{\text{pr,S}}_{\mathcal{B},\text{VOPRF}}(\lambda)$.

Now we argue that the adversary can win Game_1 only with a negligible probability. In Game_1, VOPRF evaluations are simulated using the truly random function RnF. Furthermore, $\mathcal{O}$LimEval ensures that S learns at most q_{iss} evaluations of RnF where q_{iss} is the number of issuance oracle queries that $\mathcal{A}$ makes, so S can leak at most q_{iss} evaluations of RnF through its simulation to the adversary. Thus, the adversary has to guess the output of RnF at an unrevealed position. For q_{red} being the number of token redemption oracle queries, the adversary's probability to guess the output of RnF at such an input is $q_{red}/|\mathcal{Y}|$. Finally, we conclude that $\mathsf{Adv}^{\text{om-uf}}_{\mathcal{A},\text{PP-VOPRF}}(\lambda) \leq \mathsf{Adv}^{\text{pr,S}}_{\mathcal{B},\text{VOPRF}}(\lambda) + q_{red}/|\mathcal{Y}|$. □

D.4 Proof of Theorem 6.8

Proof. We present a sequence of games to show the one-more unforgeability of PPass-RC.

$\underline{\text{Game}_1}$: Just as in the proof of Game_1 of Theorem 6.7, this game changes the VOPRF evaluations with truly random values by using the pseudorandomness property of VOPRF.

$\underline{\text{Game}_2}$: This game adds an abort condition to Game_1. In particular it aborts if there is a valid redemption oracle query for the challenge context before the challenge context is revealed to the adversary. Game_2 aborts only with a negligible probability as ctx^* is randomly chosen and the oracles do not reveal information about ctx^*. Thus, for q_{red} being the number of redemption oracle queries, $|\Pr[W_2] - \Pr[W_1]| \leq q_{red}/|C|$.

$\underline{\text{Game}_3}$: This game ensures that no H_{dg} collision occurs. If a collision occurs, it aborts. As H_{dg} is a random oracle, for q_H being the number of random oracle queries, $|\Pr[W_2] - \Pr[W_1]| \leq q_H^2/2^\lambda$.

$\underline{\text{Game}_4}$: This game just introduces a change in the order of code on when it samples the challenge redemption context ctx^*. It samples ctx^* after the adversary finishes its first-round queries instead of sampling ctx^* at the beginning of the game. ctx^* is used in $\mathcal{O}$Redeem for the if condition that increments the counter v, but we know by Game_2 that there is no such query before the challenge context is revealed. Thus, Game_4 starts checking the if condition in the redemption oracle after the challenge context is revealed. As the adversary's view does not change in this game, $\Pr[W_4] = \Pr[W_3]$.

$\underline{\text{Game}_5}$: This game adds another abort condition which limits the $\mathcal{O}$LimEval queries of the simulator S. When the challenge context is sampled, if there is a previous $\mathcal{O}\text{LimEval}(x)$ query by S for $str \in \{0,1\}^*$ and $x = \mathsf{H}_{dg}(ctx^*) \| str$, then Game_5 aborts. As the challenge context is sampled randomly after all queries, for q_{iss} being the number of token issuance queries, $|\Pr[W_5] - \Pr[W_4]| \leq q_{iss}/|C|$.

Now we finalize our proof by arguing that the probability that the adversary wins Game_6 is negligible. Observe that in Game_5 both i and v counters are equal to 0 before the challenge context is revealed. Furthermore, we also know that the VOPRF simulator S did not make any $\mathcal{O}$LimEval queries with the ctx^*, so S did not reveal any information about the evaluations of RnF on a point related to ctx^*. Thus, after learning ctx^* and making i issuance queries, the adversary can learn at most i evaluations of RnF on points related to ctx^*. In order to win, the adversary must guess the output of RnF for at least one extra point related to ctx^*. As RnF is truly random, this only happens with the probability $q_{red}/|\mathcal{Y}|$, so $\Pr[W_6] \leq q_{red}/|\mathcal{Y}|$. As a result, we conclude that $\mathsf{Adv}^{\text{tc-uf}}_{\mathcal{A},\text{PP-VOPRF}}(\lambda) \leq \mathsf{Adv}^{\text{pr,S}}_{\mathcal{B},\text{VOPRF}}(\lambda) + 2q_{red}/|C| + q_H^2/2^\lambda + q_{iss}/|\mathcal{Y}|$. □