Anonymous Tokens with Private Metadata Bit

Ben Kreuter[1], Tancrède Lepoint[1], Michele Orrù[2,3,4], and Mariana P. Raykova[1]

[1] Google, New York, USA
[2] École Normale Supérieure, CNRS, PSL University, Paris, France,
[3] Inria, Paris, France
[4] Recurse Center, New York, USA

{benkreuter,tancrede,marianar}@google.com
michele.orru@ens.psl.eu

Abstract. We present a cryptographic construction for anonymous tokens with private metadata bit, called PMBTokens. This primitive enables an issuer to provide a user with a lightweight, single-use anonymous trust token that can embed a single private bit, which is accessible only to the party who holds the secret authority key and is private with respect to anyone else. Our construction generalizes and extends the functionality of Privacy Pass (PETS'18) with this private metadata bit capability. It provides unforgeability, unlinkability, and privacy for the metadata bit properties based on the DDH and CTDH assumptions in the random oracle model. Both Privacy Pass and PMBTokens rely on non-interactive zero-knowledge proofs (NIZKs). We present new techniques to remove the need for NIZKs, while still achieving unlinkability. We implement our constructions and we report their efficiency costs.

1 Introduction

The need to propagate trust signals while protecting anonymity has motivated cryptographic constructions for anonymous credentials [Cha82, CL01]. While we have constructions that support complex statements, this comes with computation and communication costs. On the other hand, some practical uses require very simple functionality from the anonymous credential, while having very strict efficiency requirements. One such example is the setting of Privacy Pass [DGS+18]. Privacy Pass was designed as a tool for content delivery networks (CDNs), which need a way to distinguish honest from malicious content requests, so as to block illegitimate traffic that could drain network resources causing a denial of service (DoS). Previous solutions leveraged IP reputation to assess the reputation of users. While helpful in many cases, IP reputation may also lead to a high rate of false positives because of shared IP use. In particular, this is the case for users of privacy tools, such as Tor, VPNs, and I2P. Privacy Pass [DGS+18] proposes a solution for this problem using anonymous tokens as a mechanism to prove trustworthiness of the requests, without compromising the privacy of the user. Since CDNs need to potentially handle millions of requests per second, efficiency of the cryptographic construction is of extreme importance.

In this paper, we consider anonymous tokens that can convey two trust signals, in such a way that the user cannot distinguish which of the two signals is embedded in her tokens. This extension is motivated by the fact that in a system relying on anonymous trust tokens, malicious users be identified as a threat if the issuer stops providing them with tokens. Since real-world attackers have means to corrupt honest users, finding out when they have been detected could serve as an incentive to corrupt more users or adversarial learning against spam detectors. Being able to pass on the information whether a user is on an allow or disallow list, and consume it in appropriate ways on the authentication side, mitigates such behavior. There has been recent interest in primitives that provide such functionality in standardization bodies such as the IETF and W3C. This includes a recent draft proposal for a *Trusted Token API* submitted by Google at the W3C, which calls for a secret metadata bit functionality.[5] Also an IETF working group[6] is discussing standardization of the core protocol of Privacy Pass used by Cloudflare[7] together with extensions including private metadata bit.

More concretely, we consider an anonymous token primitive that provides the following functionality: a user and an issuer interact and, as a result of this interaction, the user obtains a token with a private metadata bit (PMB) embedded in it. The private metadata bit can be read from a token using the secret key held by the issuer, at redemption time. Each token is one-time use, which enables the issuer to update the trust assigned to each user without requiring a complex revocation process, by just adjusting the number of tokens that can be issued at once and the frequency of serving new token requests. Anonymous token schemes offer the following security properties: unforgeability, unlinkability, and privacy of the metadata bit. *Unforgeability* guarantees that nobody but the issuer can generate new valid tokens. *Unlinkability* guarantees that the tokens that were issued with the same private metadata bit are indistinguishable to the issuer when redeemed. *Privacy of the metadata bit* states that no party that does not have the secret key can distinguish any two tokens, including tokens issued with different metadata bits.

Our goal is to construct a primitive which achieves the above properties, and has competitive efficiency introducing minimal overhead over Privacy Pass.

1.1 Our Contributions

Our work includes the following contributions. We formalize the security properties of the primitive of anonymous tokens with private metadata bit. We present a new construction for this primitive, called PMBToken, which extends Privacy Pass (PP) to support private metadata bit, while maintaining competitive efficiency. Further, we introduce new techniques that allow to remove the need for NIZK in the constructions of both Privacy Pass and PMBToken. This simplifies

[5] See https://github.com/WICG/trust-token-api#extension-metadata and https://web.dev/trust-tokens/.
[6] See https://datatracker.ietf.org/wg/privacypass/about/.
[7] See https://blog.cloudflare.com/cloudflare-supports-privacy-pass/.

and optimizes the constructions in which the NIZK proof computation is a major bottleneck. The resulting schemes satisfy a weaker unlinkability notion. Finally, we implement all the above candidate constructions in Rust, and we summarize the performance of our schemes.

A failed approach and its insight. The starting point of our study is Privacy Pass, which uses the verifiable oblivious PRF (VOPRF) primitive of Jarecki et al. [JKK14] $F_x(t) = x\mathsf{H}_t(t)$ (additively denoted). In the oblivious PRF evaluation mechanism, the user sends to the issuer $r\mathsf{H}_t(t)$ for a randomly selected value r, receives back $rx\mathsf{H}_t(t)$ from which she recovers the output $x\mathsf{H}_t(t)$.[8] Obliviousness is guaranteed by the blinding factor r, which makes the distribution of $r\mathsf{H}_t(t)$ uniform even when knowing t. The PRF output can be verified by the user providing her with a DLEQ proof, which guarantees that $\log_{\mathsf{H}_t(t)}(F_x(t)) = \log_G X$, where G is the base point and $X \coloneqq xG$ is published by the issuer as a public parameter for the scheme.

There is a natural idea to upgrade the above functionality to support a private metadata bit, which is to have two secret keys and use each of these keys for one of the bits. However, this idea does not work directly; the reason for this stems from the fact that the underlying VOPRF is a deterministic primitive. In particular, this means that if we are using two different keys for tokens issued with different private metadata bit values, the VOPRF evaluations on the same input t will be the same if they are issued with the same key and will be different with high probability if used with different keys. Thus, if the user obtains multiple tokens using the same input value t (the issuer, by blindness, has no way of telling), she will be able to distinguish which ones were issued with the same bit.

New randomized tokens and private metadata. To resolve the above issue we introduce a construction which makes the token issuance a randomized functionality where the randomness is shared between the user and the issuer. We use the following function $F_{(x,y)}(t; S) = x\mathsf{H}_t(t) + yS$, where t is the value that will be input of the user and S is the randomness of the evaluation, which will be determined by the two parties, more specifically $S = r^{-1}\mathsf{H}_s(r\mathsf{H}_t(t); s)$ where r is the blinding factor chosen by the user and s is a random value chosen by the issuer. This functionality suffices to construct a new anonymized token where during the oblivious evaluation the user sends $T' = r\mathsf{H}_t(t)$, receives back from the issuer $s, W' = xT' + y\mathsf{H}_s(T'; s)$, unblinds the values $S = r^{-1}\mathsf{H}_s(T'; s)$ and $W = r^{-1}W'$, and outputs (S, W).

The token verification checks that $W = x\mathsf{H}_t(t) + yS$. In order to provide verifiability, the issuer provides an element of the form $X = xG + yH$ and sends a proof that $X = xG + yH$ and $W = x\mathsf{H}_t(t) + yS$ are computed using the same secret key (x, y). This is similar to Okamoto–Schnorr blind signatures [Oka92],

[8] In Privacy Pass the resulting value $x\mathsf{H}_t(t)$ is used for the derivation of a HMAC key in order to avoid credential hijacking. We cover token hijacking in the full version. Throughout this work, we assume that the communication channel between user and issuer is encrypted and authenticated.

with the key difference that we redefine this as a secret key primitive which enables us to have a round-optimal blind evaluation algorithm.

We apply the idea of using two different keys for each private metadata bit value to the above randomized construction; the resulting construction is called PMBTokens. The public parameters are now a pair $(X_0 := x_0G + y_0H, X_1 := x_1G + y_1H)$, a token issued with a private metadata bit b is of the form $W' = x_b\mathsf{H}_t(t) + y_bS$ and the DLEQ proof is replaced with a DLEQOR proof stating that either W' and X_0, or W' and X_1, are computed using the same secret key (x_0, y_0) or (x_1, y_1).

Removing the NIZK. Both Privacy Pass and PMBTokens employ zero-knowledge arguments of knowledge to achieve unlinkability. This approach guarantees that the user can verify that she has obtained a token issued under the same secret key as in the issuer's public parameters. Unlinkability follows from the fact that tokens issued under the same secret key are indistinguishable. We consider a slightly weaker unlinkability guarantee for the user during token issuance, which is that either the token she has received is issued under the public key or the token is indistinguishable from a random value, however, the user cannot distinguish these two cases. Now, a user will not be able to know in advance whether she has a token that will be valid at redemption. Another difference is that, if the issuer misbehaves, incorrectly issued tokens will be indistinguishable for the issuer from incorrectly formed tokens that a malicious user may try to use. Note that in any of the above constructions the issuer also distinguishes valid from invalid, however, the difference is that the she can check if the received tokens are valid.

We present modifications of both Privacy Pass and PMBTokens that satisfy this version of unlinkability, while removing the need for DLEQ or DLEQOR proofs and improving the computational cost for the issuer, which is the bottleneck in systems that need to support large number of users who perform many transactions and hence need to obtain tokens regularly.

Our approach for removing the DLEQ proof from Privacy Pass borrows ideas from the construction of a verifiable partially oblivious PRF of Jarecki et al. [JKR18], but simplifies their construction which has additional complexity in order to achieve user verifiability. We use the idea to use not only multiplicative but also additive blinding of the user's input in the form $T' = r(\mathsf{H}_t(t) - \rho G)$. Now, an honest evaluation of the issuer $W' = xr(\mathsf{H}_t(t) - \rho G)$ can be unblinded by the user by computing $r^{-1}W + \rho X = x\mathsf{H}_t(t) - \rho(xG) + \rho X = x\mathsf{H}_t(t)$, where $X = xG$ is the issuer's public key. On the other hand, any dishonestly computed W' which is of the form $W' = r^{-1}T' + P$ for some $P \neq 0$ when unblinded will contain a random additive factor $r^{-1}P$, thus the resulting value will be random. Similarly to Jarecki et al. [JKR18], we can recover verifiability by doing another oblivious evaluation on the same value t and comparing the outputs, which will be equal only if the the issuer used the public key for both executions. We also observe that these checks can be batched for an arbitrary number of issued tokens by computing a random linear combination of the values $\mathsf{H}_t(t_i)$, obtaining a VOPRF evaluation on that value, and comparing with the same linear com-

bination of the other tokens. Thus a user can verify n tokens by running one additional token request only. We note further that removing the zero knowledge argument significantly simplifies the issuer work, which now consist only of one multiplication.

Applying the above idea to the anonymous token construction with private metadata bit is more challenging since the user does not know which of the two public keys the issuer will use. However, the user can unblind the response from the issuer using each of the public keys and thus obtain one valid and one random token. This property turns out to be true if the issuer behaves honestly but if the issuer is malicious, he can create public keys and a response W' such that the two values obtained from the unblinding with each of the public keys are correlated and this correlation can be used for fingerprinting the user. Thus, in our construction the user computes two values $T'_d = r_d(\mathsf{H}_t(t) - \rho_d G)$, for $d = 0, 1$, and the issuer uses one of them to compute his response $W' = x_b T'_b + y_b S'_b$ with a private bit b. The user unblinds W' using both public keys and the scalars r_d, ρ_d for $d \in \{0, 1\}$ to obtain S_0, W_0, S_1, W_1, which she uses for the final token. The resulting token verifies with only one of the issuer's keys: the key corresponding to the private metadata value.

Verification oracle. One last wrinkle in the security proof is whether the adversary for unforgeability and privacy of the metadata bit properties should have access to a verification oracle for tokens of his choice. This is not explicitly supported in the current Privacy Pass security proof [DGS+18]. We provide a new proof for unforgeability of Privacy Pass in the presence of a verification oracle based on a different hardness assumption, the *Chosen Target Gap Diffie–Hellman* assumption, which is a formalization of the Chosen Target Diffie–Hellman in a Gap DH group, which was defined by Boneh et al. [BLS01]. In the context of anonymous tokens with private metadata bit, we distinguish a VERIFY oracle which just simply checks validity of the token, and a READ oracle that returns the *value* of the private metadata bit (which could be 0, 1, or invalid, and in some applications, e.g. blocklisting, we can merge the states of value 0 and invalid bit). We present an anonymous token construction that provides unforgeability and privacy for the metadata bit even when the adversary has access to the VERIFY oracle, but we crucially require that the adversary *does not get an oracle access that reads the private metadata bit of a token.*

Efficiency of our constructions. We consider the most expensive computation operation in the above protocols (scalar multiplication) and the largest communication overhead (the number of group elements transferred). We report in Table 1 the efficiency of our constructions. Additionally, the variant of our constructions that supports a verification oracle in the PMB security game adds the overhead of Okamoto–Schnorr Privacy Pass to the overhead of PMBTokens. The modifications of the constructions that do not use DLEQ or DLEQOR proofs save work for the issuer with no or moderate increase in communication and increased user computation. This computation trade-off is beneficial for settings where the issuer handles orders of magnitude more token issuance requests than

Table 1. Computation and communication costs of our constructions.

Construction	# Multiplications		Communication
	user	issuer	(# elements)
PP (Constr. 1), [DGS+18]	6	3	2
OSPP (Constr. 2)	9	6	2
PMBT (Constr. 3)	15	12	2
PPB (Constr. 4)	4	1	2
PMBTB (Constr. 5)	12	2	3

any particular user. We further implement our constructions in Rust, and report their practicals costs in Section 8. Using a Ristretto group on Curve25519, PMBTokens issuance runs in 845 µs and redemption takes 235 µs, while Privacy Pass issuance runs in 303 µs and redemption takes 95 µs. Without the issuance NIZK, Construction 5 introduces a small overhead over Privacy Pass.

Paper Organization. We overview the hardness assumptions and the building block primitives we use in Section 2. Section 3 defines our new anonymous tokens primitive and its security notions. We recall the Privacy Pass construction in Section 4, and present a (randomized) Okamoto–Schnorr anonymous tokens construction in Section 5. Next, Section 6 presents our construction for anonymous tokens with private metadata bit, called PMBTokens. Section 7 proposes modifications of Privacy Pass and PMBTokens that avoid the need of zero-knowledge proofs. Finally, Section 8 reports on the efficiency costs of our implementation.

Due to space constraints, the security proofs are provided in the full version of this paper [KLOR20].

1.2 Related work

Starting with the work of Chaum [Cha82], the concept of blind signatures has been widely used as a tool for building anonymous credentials. Blind Schnorr and Okamoto–Schnorr signatures, which have been studied and analyzed in the random oracle model [CP92, Oka92, PS00, Sch01, Sch06, FPS19], require three moves of interaction between the user and the issuer. Blind signatures constructions that achieve one round, which is the goal for our construction, rely on more expensive building blocks. Partially blind signatures, for which we also have round-optimal constructions [Fis06, SC12, BPV12], allow the issuer to embed some information in the signature, however, this information is *public*, unlike the private metadata bit that is the goal of our construction. The works of Boldyareva [Bol03] and Bellare et al. [BNPS03] achieve round optimal (one round) constructions in the random oracle model under interactive assumptions. These constructions use the same blinding idea as the VOPRF [JKK14] used by Privacy Pass, but are defined over groups where DDH is easy and CDH holds (or where the RSA assumptions hold), which enables public verifiability but requires larger group parameters. Other blind signature constructions have evolved from

constructions that need a CRS [Fis06, SC12, BFPV11] to constructions in the standard model [GG14, FHS15, FHKS16], but they rely on bilinear groups. This adds complexity to the group instantiations for schemes and computational cost, which we aim to minimize.

Group signatures [CvH91, Cam97, BMW03] present functionality which allows all member of the group to sign messages, with the property that signatures from different signers are indistinguishable. At the same time there is a master secret key that belongs to a group manager, which can be used to identify the signer of a message. We can view different signer keys as signing keys for the private metadata bits, and the master secret key as a way to read that bit value. Group blind signatures [LR98], which provide also the oblivious evaluation for the signing algorithm we aim at, provide a solution for the anonymous token functionality with a private metadata bit. Existing blind group signatures constructions [LR98, Ram13, Gha13] require multiple rounds of interaction for the oblivious signing and communication of many group elements.

Abdalla et al. [ANN06] introduced a notion of blind message authentication codes (MACs), a secret key analog to blind signatures. They showed that this notion can exist only assuming a commitment of the private key, and showed how to instantiate that primitive with Chaum's blind signatures [Cha82]. Davidson et al. [DGS+18] construct a similar private key functionality for anonymous tokens using a VOPRF [JKK14]; it is called Privacy Pass and is the basis of this work.

Everspaugh et al. [ECS+15] introduce the primitive of a partially oblivious PRF, which analogously to blind signatures allows the party with the secret key to determine part of the input for the PRF evaluation. However, this input needs to be public for verifiability. The presented partially blind PRF uses bilinear groups and pairings. Jarecki et al. [JKR18] show how to obtain a threshold variant of the partially oblivious PRF.

The work of Tsang et al. [TAKS07] presents a construction for blacklistable anonymous credentials using bilinear maps, which enables the issuer to create a blacklist of identities and the user can only generate an authentication token if she is not blacklisted; hence the user does find out whether she has been blacklisted in this process.

In keyed-verification anonymous credentials [CMZ14], the issuer and verifier are the same party. They use an *algebraic* MAC in place of a signature scheme, where the message space is a n-tuple of elements in $\mathbb{Z}_p$ (or in $\mathbb{G}$). They can be used to provide an anonymous token primitive (at a slightly higher cost) but they're overall meant for multi-use credentials. We are not aware of any extension that allows for the embedding of a private metadata bit.

2 Preliminaries

Notation. When sampling the value x uniformly at random from the set S, we write $x \leftarrow_{\$} S$. When sampling the value x from a probabilistic algorithm M, we write $x \leftarrow \mathsf{M}$. We use $:=$ to denote assignment. For an integer $n \in \mathbb{N}$, we denote

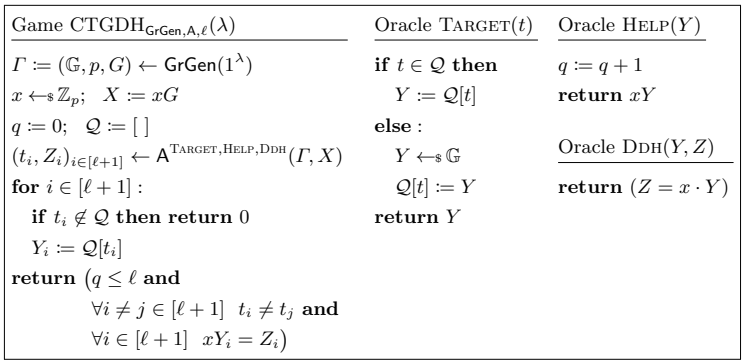

Fig. 1. The Chosen-target gap Diffie–Hellman security game.

with $[n]$ the interval $\{0, \ldots, n-1\}$. We denote vectors in bold. For a vector $\mathbf{a}$, we denote with a_i the i-th element of $\mathbf{a}$.

The output resulting form the interaction of two (interactive) PPT algorithms A, B is denoted as $[\![a, b]\!] \leftarrow \langle \mathsf{A}, \mathsf{B} \rangle$. If only the first party receives a value at the end of the interaction, we write $a \leftarrow \langle \mathsf{A}, \mathsf{B} \rangle$ instead of $[\![a, \perp]\!] \leftarrow \langle \mathsf{A}, \mathsf{B} \rangle$.

We assume the existence of a group generator algorithm $\mathsf{GrGen}(1^\lambda)$ that, given as input the security parameter in unary form outputs the description $\Gamma = (\mathbb{G}, p, G, H)$ of a group $\mathbb{G}$ of prime order p; G and H are two nothing-up-my-sleeve (NUMS) generators of $\mathbb{G}$. For simplicity, we will assume that the prime p is of length λ.

2.1 Security assumptions

We assume the reader to be familiar with the discrete logarithm (DLOG), decisional Diffie–Hellman (DDH), and computational Diffie–Hellman (CDH) assumptions. In this work, we will use the chosen-target gap Diffie–Hellman (CTGDH) assumption.

Chosen-target gap Diffie–Hellman. The chosen-target Diffie–Hellman (CTDH) assumption [Bol03, HL06] states that any PPT adversary A has negligible advantage in solving CDH on $\ell + 1$ *target* group elements, even when given access to a CDH helper oracle for ℓ instances. We formalize here its *gap* [OP01] flavor, in which the adversary has, in addition, access to a DDH oracle for arbitrary group elements. Note that the CTDH assumption was originally introduced by Boldyreva [Bol03] in gap DH groups [BLS01], that is, in groups where CDH is hard but DDH is assumed to be easy. In other words, the original definition of CTDH was *already* in groups where the adversary has access to a DDH oracle. Here, we introduce the *chosen-target gap Diffie–Hellman* assumption (CTGDH, that is, the gap version of CTDH) as a security experiment where the adversary is provided a challenge $X \in \mathbb{G}$, and has access to three oracles: the $\textsc{Target}$

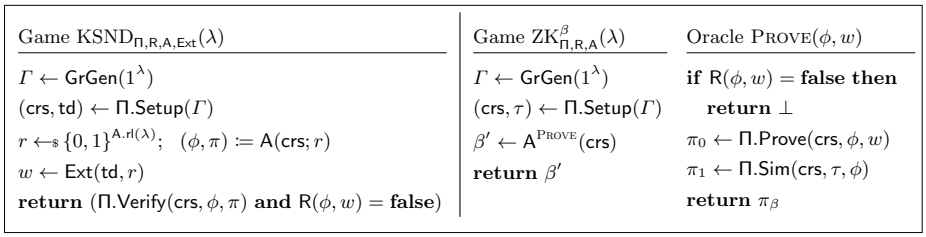

Fig. 2. Games for knowledge soundness (KSND), and zero knowledge (ZK).

oracle, that given as input a string $t \in \{0,1\}^*$, outputs a random group element; the HELP oracle, that outputs the CDH of X with an arbitrary group element $Y \in \mathbb{G}$, and the DDH oracle, that given as input two group elements $(Y,Z) \in \mathbb{G}^2$ returns 1 if and only if (X,Y,Z) is a Diffie–Hellman tuple. We describe the TARGET oracle in this cumbersome way to ease readability of the security proofs later.

Formally, we say that CTGDH holds for the group generator GrGen if for any PPT adversary A, and any $\ell \geq 0$:

$$\mathsf{Adv}^{\mathrm{ctgdh}}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda) := \Pr\left[\mathrm{CTGDH}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda) = 1\right] = \mathsf{negl}(\lambda)\,,$$

where $\mathrm{CTGDH}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda)$ is defined in Figure 1. Note that for $\ell = 0$, the game $\mathrm{CTGDH}_{\mathsf{GrGen},\mathsf{A},0}(\lambda)$ is equivalent to gap CDH.

2.2 Non-interactive arguments of knowledge

A non-interactive proof system Π for relation R consists of the following three algorithms:

- $(\mathsf{crs},\mathsf{td}) \leftarrow \Pi.\mathsf{Setup}(\Gamma)$, the setup algorithm that outputs a common reference string (CRS) crs together with some trapdoor information td.
- $\pi \leftarrow \Pi.\mathsf{Prove}(\mathsf{crs},\phi,w)$, a prover which takes as input some $(\phi,w) \in \mathsf{R}$ and a CRS crs, and outputs a proof π.
- $bool \leftarrow \Pi.\mathsf{Verify}(\mathsf{crs},\phi,\pi)$ a verifier that, given as input a statement ϕ together with a proof π outputs **true** or **false**, indicating acceptance of the proof.

The proof system Π is a non-interactive zero-knowledge (NIZK) argument of knowledge if it satisfies the following properties:

Completeness. Π is complete if every correctly generated proof verifies. More formally, a proof system Π is *complete* if for any $\Gamma \in [\mathsf{GrGen}(1^\lambda)]$, $\mathsf{crs} \in [\Pi.\mathsf{Setup}(\Gamma)]$ and $(\phi,w) \in \mathsf{R}$:

$$\Pr[\Pi.\mathsf{Verify}(\mathsf{crs},\phi,\Pi.\mathsf{Prove}(\mathsf{crs},\phi,w))] = 1 - \mathsf{negl}(\lambda)\,.$$

Knowledge soundness. A proof system Π for relation R is *knowledge-sound* if for any PPT adversary A there exists a PPT extractor Ext such that:

$$\mathsf{Adv}^{\mathrm{ksnd}}_{\Pi,\mathsf{R},\mathsf{A},\mathsf{Ext}}(\lambda) := \Pr\left[\mathrm{KSND}_{\Pi,\mathsf{R},\mathsf{A},\mathsf{Ext}}(\lambda)\right] = \mathsf{negl}(\lambda)\,,$$

where $\mathrm{KSND}_{\Pi,\mathsf{R},\mathsf{A},\mathsf{Ext}}(\lambda)$ is defined in Figure 2 and $\mathsf{A.rl}(\lambda)$ is the randomness length for the machine A. An *argument of knowledge* is a knowledge-sound proof system. In our proofs, for ease of notation, we will omit to specify explicitly that the extractor takes as input the coins of the adversary.

Zero Knowledge. A proof system Π for R is *zero-knowledge* if no information about the witness is leaked by the proof, besides membership in the relation. This is formalized by specifying an additional PPT algorithm $\Pi.\mathsf{Sim}$, that takes as input the trapdoor information td and a statement ϕ, and outputs a valid proof π indistinguishable from those generated via $\Pi.\mathsf{Prove}$. Formally, A proof system Π for relation R is zero-knowledge if for any PPT adversary A:

$$\mathsf{Adv}^{\mathsf{zk}}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda) \coloneqq \left|\Pr\left[\mathrm{ZK}^0_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)\right] - \Pr\left[\mathrm{ZK}^1_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)\right]\right| = \mathsf{negl}(\lambda)\,,$$

where $\mathrm{ZK}^{\beta}_{\Pi,\mathsf{R},\mathsf{A}}(\lambda)$ is defined in Figure 2.

Throughout this paper, we will assume the existence of the following proof systems, that we summarize here in Camenisch-Stadler notation [Cam97]:

$$\Pi_{\mathsf{DLOG}} \coloneqq \mathrm{NIZK}\{(x) : X = xG\} \tag{1}$$

$$\Pi_{\mathsf{DLEQ}} \coloneqq \mathrm{NIZK}\{(x) : X = xG \ \wedge \ W = xT\} \tag{2}$$

$$\Pi_{\mathsf{DLEQ2}} \coloneqq \mathrm{NIZK}\{(x,y) : X = xG + yH \ \wedge \ W = xT + yS\} \tag{3}$$

$$\Pi_{\mathsf{DLOGAND2}} \coloneqq \mathrm{NIZK}\{(\mathbf{x},\mathbf{y}) : \forall i \in \{0,1\} \ X_i = x_i G + y_i H\} \tag{4}$$

$$\Pi_{\mathsf{DLEQOR2}} \coloneqq \mathrm{NIZK}\{(x,y) : \exists i \in \{0,1\} \ X_i = xG + yH \wedge W = xT + yS\} \tag{5}$$

Eq. (1) and Eq. (4) are discrete logarithm proofs for one, respectively two generators. Eq. (2) and Eq. (3) prove discrete logarithm equality under one, respectively two generators. Finally, Eq. (5) proves discrete logarithm equality for one out of two group elements (in the witness, the index is denoted as $i \in \{0,1\}$). In the full version [KLOR20], we provide a more formal description of the above relations, and efficient instantiations with techniques for batching proofs at issuance time.

3 Anonymous tokens

We describe two flavors of anonymous tokens. The first flavor enables a *user* to obtain a *token* from an *issuer*; the user can later use this token as a trust signal, for one-time authentication. In the second flavor, the issuer has an additional input during the token issuance, a *private metadata bit*, that is hidden within the token. The private metadata bit can later be recovered by the issuer at redemption time. The following definition captures both functionalities; in shaded text, we refer only to the anonymous token with private metadata bit.

Anonymous token. An *anonymous token scheme with private metadata bit* AT consists of the following algorithms:

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{AT.Setup}(1^\lambda)$, the setup algorithm that takes as input the security parameter λ in unary form, and returns a CRS crs and a trapdoor td.
  All the remaining algorithms take crs as their first input, but for notational clarity, we usually omit it from their lists of arguments.

– $(\mathsf{pp}, \mathsf{sk}) \leftarrow \mathsf{AT.KeyGen}(\mathsf{crs})$, the key generation algorithm that generates a private key sk along with a set of public parameters pp;
– $\sigma \leftarrow \langle \mathsf{AT.User}(\mathsf{pp}, t), \mathsf{AT.Sign}(\mathsf{sk}, b) \rangle$, the token issuance protocol, that involves interactive algorithms $\mathsf{AT.User}$ (run by the user) with input a value $t \in \{0,1\}^\lambda$, and $\mathsf{AT.Sign}$ (run by the issuer) with input the private key sk and a bit b. At the end of the interaction, the issuer outputs nothing, while the user outputs σ, or $\perp$.
– $bool \leftarrow \mathsf{AT.Verify}(\mathsf{sk}, t, \sigma)$, the verification algorithm that takes as input the private key sk and a token (t, σ). It returns a boolean indicating if the token was valid or not.
– $\mathsf{ind} \leftarrow \mathsf{AT.ReadBit}(\mathsf{sk}, t, \sigma)$, the metadata extraction algorithm that takes as input the private key sk, and a token (t, σ). It returns an indicator $\mathsf{ind} \in \{\perp, 0, 1\}$, which is either the private metadata bit, or $\perp$.

Throughout the rest of this paper, we assume that AT has a one-round signing protocol initiated by the user. Thus, for simplicity, we split the signing algorithms ($\mathsf{AT.Sign}$ and $\mathsf{AT.User}$) into non-interactive algorithms that take as input a message, and the partial state (if any). They will return the next message together with the updated state st_i. Concretely, the signing protocol will be composed of the following (non-interactive) algorithms:
– $(\mathsf{usr\_msg}_0, \mathsf{st}_0) \leftarrow \mathsf{AT.User}_0(\mathsf{pp}, t)$;
– $\mathsf{srv\_msg}_1 \leftarrow \mathsf{AT.Sign}_0(\mathsf{sk}, \mathrm{b}, \mathsf{usr\_msg}_0)$;
– $\sigma \leftarrow \mathsf{AT.User}_1(\mathsf{st}_0, \mathsf{srv\_msg}_1)$

We demand that anonymous token schemes satisfies correctness, unforgeability, unlinkability, and privacy of the metadata bit.

Correctness. An anonymous token scheme AT is *correct* if any honestly-generated token verifies and the correct private metadata bit is retrieved successfully. That is, for any $\mathsf{crs} \in [\mathsf{AT.Setup}(1^\lambda)]$, any $(\mathsf{pp}, \mathsf{sk}) \in [\mathsf{AT.KeyGen}(\mathsf{crs})]$, any $t \in \{0,1\}^\lambda$, and $b \in \{0,1\}$:

$$\Pr[\mathsf{AT.Verify}(\mathsf{sk}, t, \langle \mathsf{AT.User}(\mathsf{pp}, t), \mathsf{AT.Sign}(\mathsf{sk}, \mathrm{b}) \rangle) = 1] = 1 - \mathsf{negl}(\lambda), \quad (6)$$

$$\Pr[\mathsf{AT.ReadBit}(\mathsf{sk}, t, \langle \mathsf{AT.User}(\mathsf{pp}, t), \mathsf{AT.Sign}(\mathsf{sk}, b) \rangle) = b] = 1 - \mathsf{negl}(\lambda) \quad (7)$$

Unforgeability. The first security property that we require from an anonymous token is unforgeability, which guarantees that no adversary can redeem more tokens than it is allowed. This is formalized with a standard one-more security game where the adversary can interact with the issuer at most ℓ times, and at the end must output $\ell + 1$ valid tokens. The adversary has also access to a verification oracle for tokens of its choice. In the private metadata bit variant, the adversary can interact with the issuer ℓ times for each private metadata bit, but should not be able to generate $\ell + 1$ valid tokens with the same private metadata.

Definition 1 (One-more unforgeability). *An anonymous token scheme* AT *is* one-more unforgeable *if for any* PPT *adversary* A*, and any* $\ell \geq 0$*:*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{AT},\mathsf{A},\ell}(\lambda) \coloneqq \Pr\left[\mathrm{OMUF}_{\mathsf{AT},\mathsf{A},\ell}(\lambda) = 1\right] = \mathsf{negl}(\lambda),$$

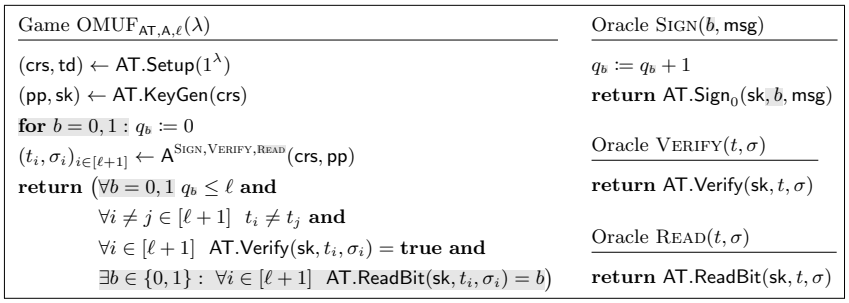

Fig. 3. One-more unforgeability game for the anonymous token scheme AT.

where $\mathrm{OMUF}_{\mathsf{AT},\mathsf{A},\ell}(\lambda)$ *is defined in Figure 3.*

Unlinkability. This security property is concerned with user anonymity, and guarantees that a malicious issuer cannot link the redemption of a token with a particular execution of the token issuance protocol. More precisely, in κ-unlinkability, if m tokens were issued but not yet redeemed, the adversary cannot link the relative issuance session of a token with probability better than κ/m, even after seeing the remaining $m-1$ tokens in a random order.

Definition 2 (Unlinkability). *An anonymous token scheme* AT *is* κ-unlinkable *if for any* PPT *adversary* A*, and any* $m > 0$*:*

$$\mathsf{Adv}^{\mathrm{unlink}}_{\mathsf{AT},\mathsf{A},m}(\lambda) := \Pr\left[\mathrm{UNLINK}_{\mathsf{AT},\mathsf{A},m}(\lambda) = 1\right] \le \frac{\kappa}{m} + \mathsf{negl}(\lambda)\,,$$

where $\mathrm{UNLINK}_{\mathsf{AT},\mathsf{A},m}(\lambda)$ *is defined in Figure 4.*

Private metadata bit. The last security property protects the private metadata bit in the issued tokens.[9] It guarantees that the private metadata embedded in a token is entirely hidden from anyone who does not possess the private key including the user. More precisely, we require that, even if the adversary corrupts a large number of users, it cannot guess with probability non-negligibly bigger than $1/2$ if newly issued tokens have private metadata 0 or 1.

Formally, this is modeled as an indistinguishability game where the adversary has access to two signing oracles: one where it can provide both the message to be signed and the private metadata bit to be used, and one where the adversary chooses only the message (the metadata bit is fixed), and a verification oracle for the validity of the tokens. The adversary's goal is to guess the challenge private metadata bit used.

[9] Consider, e.g., the following practical scenario: the issuer is suspecting that it is targeted by a DoS attack, and decides to tag users that it believes are controlled by a bot using the private metadata bit. The private metadata bit property ensures it is difficult for anyone, but the issuer, to learn how malicious traffic is classified.

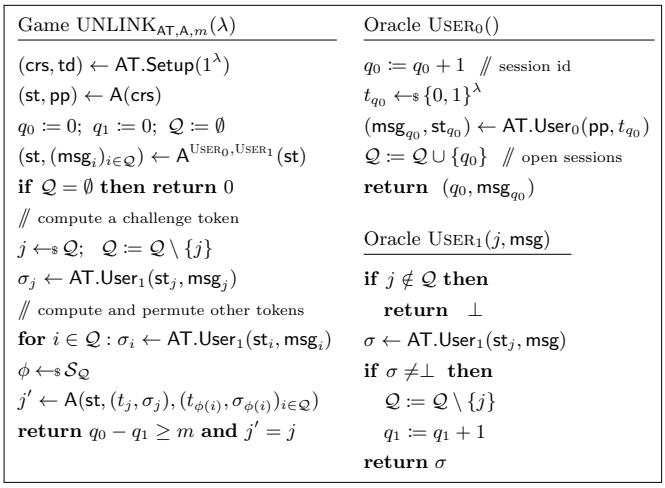

Fig. 4. Unlinkability game for the anonymous token scheme AT. For a set X, $\mathcal{S}_X$ denotes the symmetric group of X.

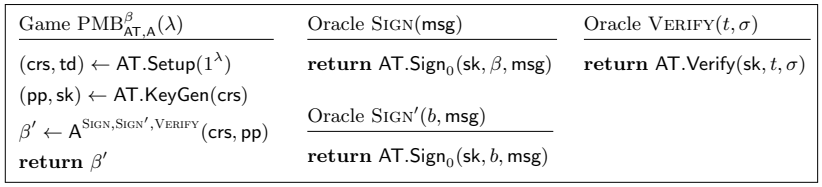

Fig. 5. Private metadata bit game for the anonymous token scheme AT.

Definition 3 (Private metadata bit). *An anonymous token scheme* AT *provides* private metadata bit *if for any* PPT *adversary* A*:*

$$\mathsf{Adv}^{\text{pmb}}_{\mathsf{AT},\mathsf{A}}(\lambda) := \left|\Pr\left[\text{PMB}^0_{\mathsf{AT},\mathsf{A}}(\lambda)\right] - \Pr\left[\text{PMB}^1_{\mathsf{AT},\mathsf{A}}(\lambda)\right]\right| = \mathsf{negl}(\lambda)\,,$$

where $\text{PMB}^\beta_{\mathsf{AT},\mathsf{A}}(\lambda)$ *is defined in Figure 5.*

Token hijacking. In our formalization, we do not consider man-in-the-middle adversaries that can steal tokens from honest users. This attack vector, called *token hijacking*, can be mitigated with the use of message authentication codes (MACs). Roughly speaking, instead of sending the entire token (t,σ) over the wire, the user can derive a symmetric key $\mathsf{k} := \mathsf{H}(\sigma)$ to MAC a shared message (e.g., the resource or the URL she's trying to access). The resulting message authentication code is sent together with t to the issuer (and any supplementary randomness that the user needs to recompute σ). We discuss in detail such concerns in the full version of this paper [KLOR20].

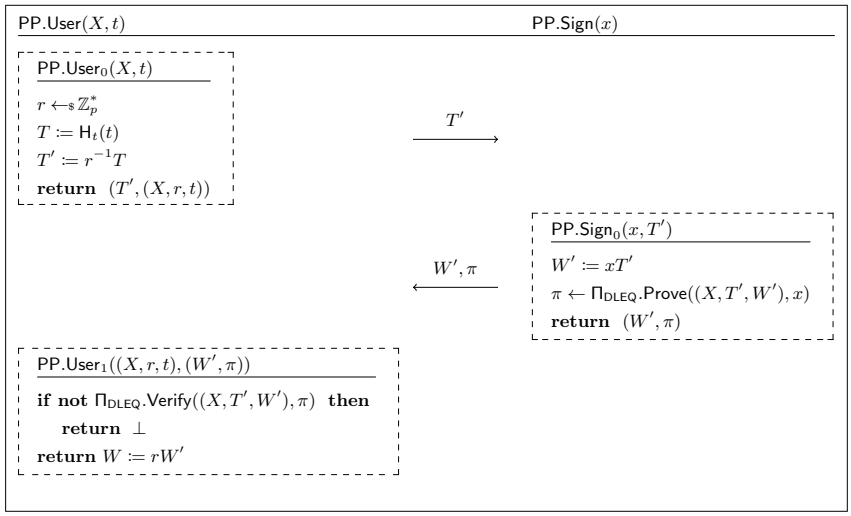

Fig. 6. Token issuance for PP (Construction 1).

4 Review: Privacy Pass

We start by recalling, using the notation from Section 3, the anonymous token scheme proposed in [DGS+18] (under the name Privacy Pass) and built on top of the verifiable oblivious PRF (VOPRF) "2HashDH-NIZK" [JKK14]. Privacy Pass uses a Schnorr proof in the issuance phase, that we generalize here to any NIZK. Differently from the initial proof of Goldberg et al. [DGS+18], our proof takes into account the presence of a verification oracle, and the knowledge error of the proof system.

Construction 1 (Privacy Pass). Let Π_{DLEQ} be a proof system for relation $\mathsf{R}_{\mathsf{DLEQ}}$; let H_t be a random oracle $\{0,1\}^* \to \mathbb{G}$. The anonymous token scheme PP [DGS+18] is composed of the following algorithms:

- $(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{PP.Setup}(1^\lambda)$: invoke the group generator $\Gamma \leftarrow \mathsf{GrGen}(1^\lambda)$ and the CRS generation algorithm of the underlying proof system $(\mathsf{pcrs},\mathsf{ptd}) \leftarrow \Pi_{\mathsf{DLEQ}}.\mathsf{Setup}(\Gamma)$. Return $\mathsf{crs} \coloneqq (\Gamma,\mathsf{pcrs})$ and $\mathsf{td} \coloneqq \mathsf{ptd}$.
- $(X,x) \leftarrow \mathsf{PP.KeyGen}(\mathsf{crs})$: sample a uniformly random element $x \leftarrow_\$ \mathbb{Z}_p^*$, that will constitute the secret key. Let $X \coloneqq xG$ be the public parameter. Return (X,x).
- $W \leftarrow \langle \mathsf{PP.User}(X,t), \mathsf{PP.Sign}(x)\rangle$: illustrated in Figure 6.
- $bool \leftarrow \mathsf{PP.Verify}(x,t,W)$: return **true** if $W = x\mathsf{H}_t(t)$; else, return **false**.

Note that this anonymous token protocol is deterministic, i.e., there will exist a unique value $W \in \mathbb{G}$ corresponding to a string $t \in \{0,1\}^\lambda$ that verifies. This property will make difficult to directly extend the construction to support private metadata bit.

Correctness. By correctness of the underlying proof system, at the end of the protocol the user returns $\perp$ only with negligible probability. If the user returns $W \in \mathbb{G}$, then W always satisfies the verification equation, since: $W = rW' = r(xT') = xT = x\mathsf{H}_t(t)$.

Security. PP satisfies both unforgeability and 1-unlinkability.

Theorem 4. *If CTGDH holds for* GrGen *and* Π_{DLEQ} *is a zero-knowledge proof system for relation* R_{DLEQ}*, then* $\mathsf{PP}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQ}}]$ *is one-more unforgeable with advantage:*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{PP},\ell}(\lambda) \leq \mathsf{Adv}^{\mathrm{ctgdh}}_{\mathsf{GrGen},\ell}(\lambda) + \mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLEQ}},\mathsf{R}_{DLEQ}}(\lambda).$$

The proof can be found in the full version, and follows directly from chosen-target gap Diffie–Hellman for GrGen, and zero-knowledge of Π_{DLEQ}. Consider an adversary A in the game $\mathrm{OMUF}_{\mathsf{PP},\mathsf{A}}(\lambda)$. To win the game, A must return $\ell + 1$ tokens $(t_i, W_i)_{i \in [\ell+1]}$ such that, for all $i \in [\ell + 1]$:

$$\text{(a) } x\mathsf{H}_t(t_i) = W_i, \qquad \text{(b) } \forall j \neq i : \quad t_j \neq t_i \tag{8}$$

During its execution, the adversary A can query at most ℓ times the signing oracle, which given as input $T^* \in \mathbb{G}$ computes the Diffie–Hellman $W = xT^*$, and sends it together with a proof π that W was correctly computed; additionally, A can query the oracle $\textsc{Verify}(t^*, W^*)$ that returns 1 if $W^* = x\mathsf{H}_t(t^*)$, that is, if $(X, \mathsf{H}_t(t^*), W^*)$ is a DH tuple. Since Π_{DLEQ} is zero-knowledge, it is possible to simulate the proof π. We are thus left with the game CTGDH (Fig. 1), where $\textsc{Verify}$ can be replaced by the oracle $\textsc{Ddh}$, H_t by $\textsc{Target}$, and $\textsc{Sign}$ by the oracle $\textsc{Help}$ (together with $\Pi_{\mathsf{DLEQ}}.\mathsf{Sim}$).

Theorem 5. *Let* GrGen *be a group generator algorithm. If* Π_{DLEQ} *is a knowledge-sound proof system for relation* R_{DLEQ}*, then* $\mathsf{PP}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQ}}]$ *is 1-unlinkable.*

We remark that the i-th message sent by $\mathsf{PP}.\mathsf{User}_0$ is $T'_i = r^{-1}T_i$, for a uniformly random $r_i \in \mathbb{Z}_p^*$. Therefore, T'_i contains no information about T_i or t_i. Additionally, by knowledge soundness of Π_{DLEQ}, it is possible to extract the witness $x \in \mathbb{Z}_p$ used to compute the signatures. With it, the user can compute W_i herself, without ever using the responses from the issuer. It follows that the view of the adversary is limited to random group elements and PP is 1-unlinkable, as long as the proof system is knowledge-sound.

5 Okamoto–Schnorr Privacy Pass

In this section, we describe a novel anonymous token scheme that generalizes PP (Section 4) and allows for *randomized* tokens, which will be an important property when we extend the construction to support private metadata bit (Section 6). Roughly speaking, while in PP we issue tokens (and DLEQ proofs) using one generator G of $\mathbb{G}$, in this construction we will issue tokens under two generators (G, H), in a similar way to Okamoto–Schnorr [Oka92] signatures. Similarly to Okamoto–Schnorr, it is important here that the discrete logarithm of H base G is unknown. Fixing $y = 0$ in the protocol below, we obtain PP (cf. Section 4).

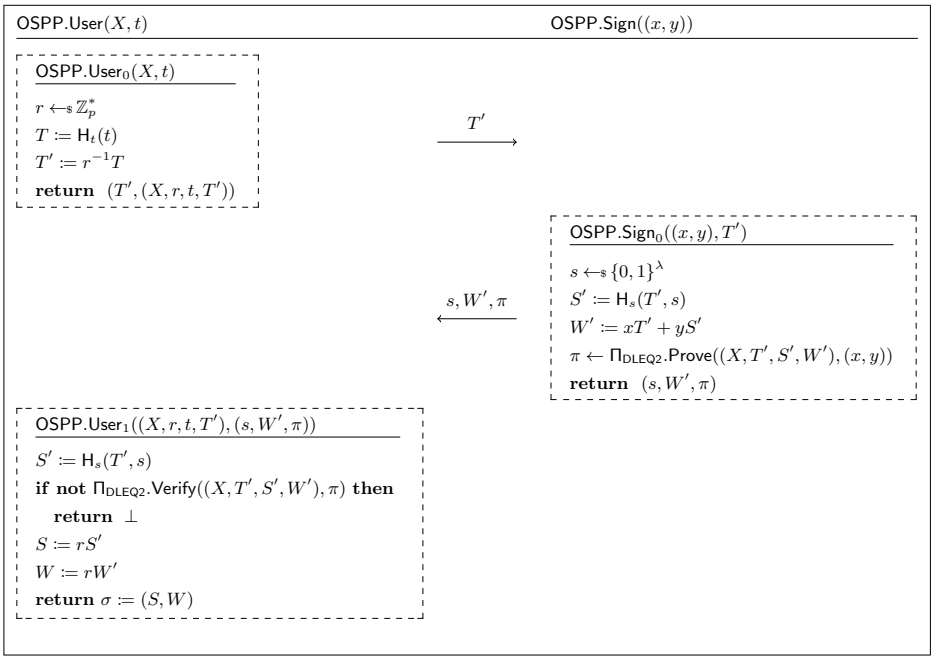

Fig. 7. Token issuance for OSPP (Construction 2).

Construction 2 (Okamoto–Schnorr Privacy Pass). Let GrGen be a group generator algorithm; let Π_{DLEQ2} be a proof system for relation $\mathsf{R}_{\mathsf{DLEQ2}}$; let $\mathsf{H}_t, \mathsf{H}_s$ be two random oracles $\{0,1\}^* \to \mathbb{G}$. We construct an anonymous token scheme OSPP defined by the following algorithms:

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{OSPP.Setup}(1^\lambda)$: invoke the group generator $\Gamma \leftarrow \mathsf{GrGen}(1^\lambda)$ and the CRS generation algorithm of the underlying proof system $(\mathsf{pcrs}, \mathsf{ptd}) \leftarrow \Pi_{\mathsf{DLEQ2}}.\mathsf{Setup}(\Gamma)$. Return $\mathsf{crs} := (\Gamma, \mathsf{pcrs})$ and $\mathsf{td} := \mathsf{ptd}$.
- $(X, (x, y)) \leftarrow \mathsf{OSPP.KeyGen}(\mathsf{crs})$: sample the secret key $(x, y) \leftarrow\!\!\$\ (\mathbb{Z}_p^*)^2$. Let $X := xG + yH$ be the public parameter. Return $(X, (x, y))$.
- $\sigma \leftarrow \langle \mathsf{OSPP.User}(X, t), \mathsf{OSPP.Sign}((x, y)) \rangle$: illustrated in Figure 7.
- $bool \leftarrow \mathsf{OSPP.Verify}((x, y), t, \sigma)$: read $(S, W) := \sigma$. Return **true** if $W = x\mathsf{H}_t(t) + yS$; else, return **false**.

Correctness. By correctness of the underlying proof system, the protocol aborts only with negligible probability. If the user returns $\sigma := (S, W) \in \mathbb{G}^2$, then σ always satisfies the verification equation, since $W = rW' = r(xT' + yS') = xT + yS = x\mathsf{H}_t(t) + yS$.

Security. OSPP satisfies both unforgeability and 1-unlinkability.

Theorem 6. *If CTGDH holds for* GrGen*, and* Π_{DLEQ2} *is a zero-knowledge proof system for relation* R_{DLEQ2}*, then* $\mathsf{OSPP}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQ2}}]$ *is one-more unforgeable*

with advantage:

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{OSPP},\ell}(\lambda) \leq \mathsf{Adv}^{\mathrm{ctgdh}}_{\mathsf{GrGen},\ell}(\lambda) + \mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLEQ2}},\mathsf{R}_{DLEQ2}}(\lambda).$$

The proof of unforgeability is essentially the same argument of the previous section, with a slightly more careful analysis to deal with the additional element $s \in \{0,1\}^\lambda$. The reduction for CTGDH receives a challenge $A \in \mathbb{G}$, that is now embedded in the public parameters as $X = A + yH$ for some $y \leftarrow\!\!\$\, \mathbb{Z}_p$; the signing oracle computes $W' = \textsc{Help}(T') + yS'$, and the read oracle $\textsc{Ddh}(W - yS)$. The tokens returned by the adversary can be converted in CDH solutions computing $W_i - yS_i$, for all $i \in [\ell + 1]$.

Theorem 7. *If DDH holds for* GrGen *and* Π_{DLEQ2} *is an argument of knowledge for relation* R_{DLEQ2}*, then* $\mathsf{OSPP}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQ2}}]$ *is* 1*-unlinkable.*

Similarly to the previous proof, we first notice that in the i-th message, T'_i contains no information about t; additionally, by knowledge soundness of the proof system, W_i must be computed with the same "witness" (x, y) satisfying $X = xG + yH$ (if there exists $(x', y') \neq (x, y)$, then it is possible to construct an adversary for discrete log for GrGen). The core difference now is that we use the same blinding factor r both on S' and W'. The proof hence proceeds in two steps: first, $W \coloneqq rW'$ is computed as $W \coloneqq xT + yS$ with the extracted witness, and next $S \coloneqq rS'$ is replaced by $S \leftarrow\!\!\$\, \mathbb{G}$. This last step can be reduced to DDH: if the adversary manages to distinguish $(T', T = rT', S', S = rS')$ from $(T', T = rT', S', U)$ for $U \leftarrow\!\!\$\, \mathbb{G}$, then it is possible to construct an adversary B for game $\mathrm{DDH}^{\beta}_{\mathsf{GrGen},\mathsf{B}}(\lambda)$.

6 Private metadata bit tokens

In this section, we present PMBTokens, an extension of the anonymous token construction from Section 5 that supports a private metadata bit. The high-level idea is that we use two different secret keys, one for each private metadata bit. In order to hide which bit is associated with the token, we will use OR proofs (i.e., Π_{DLEQOR2} of Eq. (5)).

Construction 3 (PMBTokens). Let GrGen be a group generator algorithm; let Π_{DLEQOR2} be a proof system for relation $\mathsf{R}_{\mathsf{DLEQOR2}}$; let $\mathsf{H}_t, \mathsf{H}_s$ be two random oracles $\{0,1\}^* \to \mathbb{G}$. We construct an anonymous token scheme PMBT defined by the following algorithms:

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{PMBT.Setup}(1^\lambda)$: invoke the group generator $\Gamma \leftarrow \mathsf{GrGen}(1^\lambda)$ and the CRS generation algorithm of the underlying proof system $(\mathsf{pcrs}, \mathsf{ptd}) \leftarrow \Pi_{\mathsf{DLEQOR2}}.\mathsf{Setup}(\Gamma)$. Return $\mathsf{crs} \coloneqq (\Gamma, \mathsf{pcrs})$ and $\mathsf{td} \coloneqq \mathsf{ptd}$.
- $(\mathbf{X}, (\mathbf{x}, \mathbf{y})) \leftarrow \mathsf{PMBT.KeyGen}(\mathsf{crs})$: let $(\mathbf{x}, \mathbf{y}) \leftarrow\!\!\$\, (\mathbb{Z}_p^*)^2 \times (\mathbb{Z}_p^*)^2$ be the secret key. Define the public parameters as:

$$\mathbf{X} \coloneqq \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} \coloneqq \begin{bmatrix} x_0 G + y_0 H \\ x_1 G + y_1 H \end{bmatrix};$$

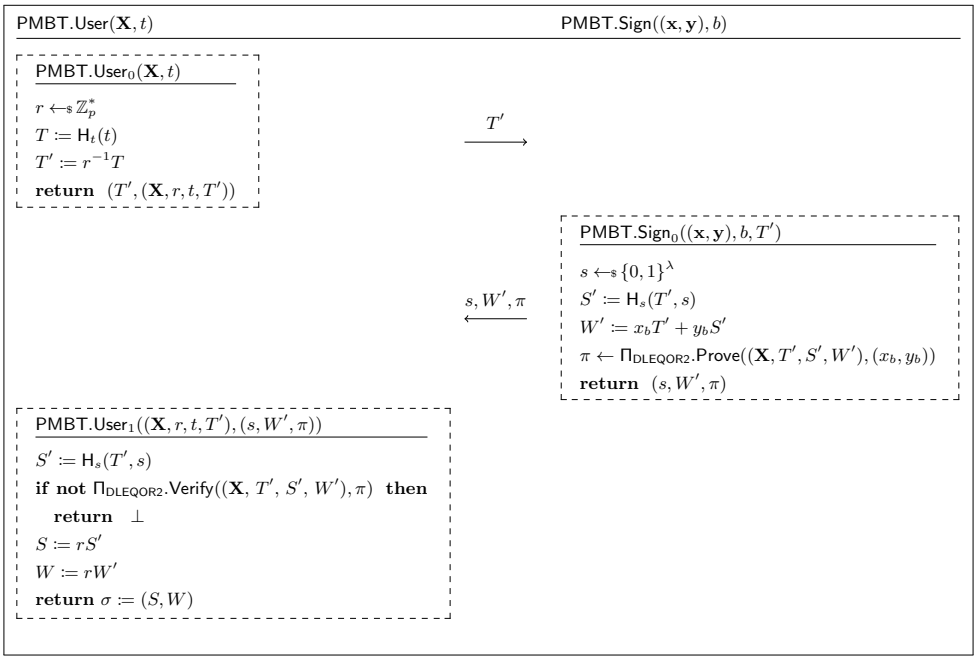

Fig. 8. Token issuance for PMBT (Construction 3).

restart if $X_0 = X_1$. Return $(\mathbf{X}, (\mathbf{x}, \mathbf{y}))$.

- $\sigma \leftarrow \langle \mathsf{PMBT.User}(\mathsf{pp}, t), \mathsf{PMBT.Sign}(\mathsf{sk}, b) \rangle$: illustrated in Figure 8.
- $bool \leftarrow \mathsf{PMBT.Verify}((\mathbf{x}, \mathbf{y}), t, \sigma)$: return **true**.
- $\mathsf{ind} \leftarrow \mathsf{PMBT.ReadBit}((\mathbf{x}, \mathbf{y}), t, \sigma)$: read $(S, W) := \sigma$. Then:
  (a) if $W = x_0\mathsf{H}_t(t) + y_0 S$ and $W \neq x_1\mathsf{H}_t(t) + y_1 S$, return 0;
  (b) if $W \neq x_0\mathsf{H}_t(t) + y_0 S$ and $W = x_1\mathsf{H}_t(t) + y_1 S$, return 1;
  (c) else, return $\perp$.

Our construction does *not* provide a (meaningful) implementation of Verify, but only a ReadBit functionality. We elaborate this point in Section 6.1; this construction can be combined with OSPP to provide an actual verification procedure (described in the full version).

Correctness. Validity of honestly-generated tokens (cf. Equation (6)) holds perfectly because PMBT.Verify always returns **true**; we focus here on proving that the bit embedded is read correctly with overwhelming probability (Equation (7)). By correctness of the underlying proof system, the protocol aborts only with negligible probability. If the user returns $(S, W) \in \mathbb{G}^2$, then there exists $b \in \{0, 1\}$ such that $W = rW' = r(x_b T' + y_b S') = x_b T + x_b S = x_b\mathsf{H}_t(t) + y_b S$. The probability that the above equation holds for both $b = 0$ and $b = 1$ (in which case, PMBT.ReadBit returns $\perp$), is statistically negligible. In fact, if:

$$W = x_0\mathsf{H}_t(t) + y_0 S = x_1\mathsf{H}_t(t) + y_1 S,$$

then we have two possibilities:

(a) $y_0 = y_1$, which in turn implies that $x_0\mathsf{H}_t(t) = x_1\mathsf{H}_t(t)$. Because $x_0 \neq x_1$ (by construction of $\mathsf{PMBT.KeyGen}$), this happens only if $\mathsf{H}_t(t)$ is the identity element. This event happens with probability $1/p$;

(b) $y_0 \neq y_1$, which in turn means that: $\mathsf{H}_t(t) = \frac{x_0 - x_1}{y_0 - y_1} S$. However, the left-hand side of the equation is distributed uniformly at random in $\mathbb{G}$ and independently from the terms on the right-hand side. This event happens with probability $1/p$.

Security. In the full version, we prove the one-more unforgeability and **2**-unlinkability of PMBT.

Theorem 8. *If CTGDH holds for* GrGen *and* Π_{DLEQ2} *is a zero-knowledge proof system for relation* R_{DLEQ2}*, then* $\mathsf{PMBT}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQ2}}]$ *is one-more unforgeable with advantage:*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{PMBT},\ell}(\lambda) \leq 2\mathsf{Adv}^{\mathrm{ctgdh}}_{\mathsf{GrGen},\ell}(\lambda) + \mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLEQ2}},\mathsf{R}_{DLEQ}}(\lambda).$$

Consider an adversary A in the game $\mathrm{OMUF}_{\mathsf{PMBT},\mathsf{A}}(\lambda)$. A wins the game if it returns $\ell + 1$ tokens $(t_i, (S_i, W_i))_{i \in [\ell+1]}$ such that there exists a $b \in \{0, 1\}$ satisfying:

$$\text{(a) } \forall i \in [\ell + 1] : \; x_b\mathsf{H}_t(t_i) + y_b S_i = W_i, \qquad \text{(b) } \forall j \neq i : \;\; t_j \neq t_i. \tag{9}$$

During its execution, A can query $\ell + 1$ times the signing oracle for $b = 0$, *and* $\ell+1$ times for $b = 1$. We claim (in a similar way to Eq. (8)), that A can be used to construct an adversary B that solves CTGDH. We embed the CTGDH challenge $A \in \mathbb{G}$ in one of the two keys: we sample $b^* \leftarrow_{\$} \{0, 1\}$, and set $X_{b^*} \coloneqq A + y_{b^*}H$, where $y_{b^*} \leftarrow_{\$} \mathbb{Z}_p$. We construct X_{1-b^*} following the key generation algorithm. Queries by A to the oracle $\textsc{Sign}$ are responded in the following way: if the adversary demands issuance for hidden metadata b^*, we use the $\textsc{Help}$ oracle to return $W' = \textsc{Help}(t, T') + y_{b^*}S'$; otherwise B just follows the signing protocol and computes W' using (x_{1-b^*}, y_{1-b^*}). The zero-knowledge proof is simulated. Queries to the oracle $\textsc{Read}$ can still be answered by B with the help of the oracle $\textsc{Ddh}$ available in the game $\mathrm{CTGDH}_{\mathsf{GrGen},\mathsf{B},\ell}(\lambda)$. Queries to $\textsc{Verify}$ are trivially dealt with, by answering **true**. If at the end of its execution A presents forgeries for the bit $b = b^*$, then by winning condition (a) (cf. Eq. (9)), for all $i \in [\ell + 1]$, $(W_i - y_{b^*}S_i)$ is the CDH of the challenge A with $\mathsf{H}_t(t_i)$, which we replace with the CTGDH oracle $\textsc{Target}$ thus presenting $\ell + 1$ CDH solutions for the challenge $A \in \mathbb{G}$. If the guess b^* was not correct, then B outputs $\perp$, and we consider the game lost. It follows that B wins the game $\mathrm{CTGDH}_{\mathsf{GrGen},\mathsf{B},\ell}(\lambda)$ half the time that A wins $\mathrm{OMUF}_{\mathsf{PMBT},\mathsf{A},\ell}(\lambda)$.

Theorem 9. *If DDH holds for* GrGen *and* Π_{DLEQOR2} *is a knowledge-sound proof system for relation* $\mathsf{R}_{\mathsf{DLEQOR2}}$*, then* $\mathsf{PMBT}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQOR2}}]$ *is* 2*-unlinkable.*

The key idea is that adversary can now embed different private metadata bits at issuance time, at most halving the anonymity set. We use the knowledge extractor to partition the sessions in two buckets: U_0, those associated to the bit 0, and U_1, those with bit 1. We sample a biased $b \in \{0,1\}$ (depending on the distribution of the private metadata bit in the tokens) and select two sessions coming from the same bucket U_b. The probability of success of the adversary will be upper bounded by $2/m + \mathsf{negl}(\lambda)$.

Theorem 10. *If DDH holds for the group generator* GrGen*, and* Π_{DLEQOR2} *is a zero-knowledge proof system for relation* $\mathsf{R}_{\mathsf{DLEQOR2}}$ *then* $\mathsf{PMBT}[\mathsf{GrGen}, \Pi_{\mathsf{DLEQOR2}}]$ *provides private metadata bit with advantage:*

$$\mathsf{Adv}^{\mathrm{pmb}}_{\mathsf{PMBT}}(\lambda) \leq \frac{O(q^2)}{2^\lambda} + 2\mathsf{Adv}^{\mathrm{ddh}}_{\mathsf{GrGen}}(\lambda) + 2\mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLEQOR2}},\mathsf{R}_{\mathsf{DLEQOR2}}}(\lambda),$$

where q *is the number of queries the adversary makes to* H_s *or* Sign.

The proof is done by means of a hybrid argument, where the first hybrid is $\mathrm{PMB}^0_{\mathsf{A},\mathsf{PMBT}}(\lambda)$ and the last hybrid is $\mathrm{PMB}^1_{\mathsf{A},\mathsf{PMBT}}(\lambda)$. In the first hybrid, instead of generating the proof using the prover $\Pi_{\mathsf{DLEQOR2}}.\mathsf{Prove}$ in the Sign$'$ oracle, we use the zero-knowledge simulator $\Pi_{\mathsf{DLEQOR2}}.\mathsf{Sim}$. The advantage in distinguishing is trivially $\mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLEQOR2}},\mathsf{R}_{\mathsf{DLEQOR2}},\mathsf{B}}(\lambda)$. Then, instead of computing the signature as $W' = x_0T' + y_0S'$, it computes $W' := x_0T' + y'S'$, for some $y' \leftarrow\!\!\$\ \mathbb{Z}_p$ sampled after the key generation phase. This hybrid can be shown indistinguishable from the previous one under DDH assumption: $(X - x_0G, S', W' - x_0T') \in \mathbb{G}$ is in fact a DDH triple in the previous hybrid, and a random triple now. Using random self-reducibility of DDH we can answer all queries to Sign$'$ using a single DDH challenge. At this point, we remark that W' is distributed uniformly at random (because $y'S'$ is so) and we can therefore swap x_0 with x_1. A final sequence of hybrid replaces y' with y_1 (again indistinguishable from DDH) and then the simulator with the honest prover. The full proof is available in the full version of this paper [KLOR20].

6.1 Enabling token verification

The anonymous token scheme PMBT (Construction 3) does not provide a meaningful verification algorithm, as it always output **true**. Indeed, we note that, given two tokens $(t_0, (S_0, W_0))$ and $(t_1, (S_1, W_1))$, if $t_0 = t_1$, then $\big(t^* = t_0 = t_1, (S^* = 2S_0 - S_1, W^* = 2W_0 - W_1)\big)$ is a triple of random elements satisfying $W^* = x_b\mathsf{H}(t^*) + y_bS^*$ only if the same metadata bit b was used. Henceforth, having a verification oracle that checks for the above relation allows an adversary in the game $\mathrm{PMB}^\beta_{\mathsf{PMBT},\mathsf{A}}(\lambda)$ to check if two tokens corresponding to the same t were issued with the same private metadata bit.

Instead, we propose to enable such a functionality by combining the PMBT scheme with Okamoto–Schnorr Privacy Pass, into one token that has two parts: a token that has no private metadata and can be used for validity verification, and a second token, which provides a private metadata bit. It is important that these

two parts could not be separated (they will depend on the same $\mathsf{H}_t(t), S$ values) and used independently for the purpose of reading the metadata bit. We present in details the design combining Constrs. 2 and 3 in the full version [KLOR20].

Jumping ahead, we can also instantiate this design by combining Constrs. 4 and 5 that do not use NIZK during issuance. In the later case the unlinkability will degrade to 6-unlinkability since the issuer can cause each of the two token to be invalid independently.

7 Removing the NIZKs during issuance

In this section, we present a general technique for removing the NIZK sent at issuance time in the previous schemes, and replace it with a proof of possession sent only once. We present a formal analysis of it for PP and PMBT (Constructions 1 and 3). We recall that the role of the NIZK is to provide unlinkability for the user, as they can check that the tokens received are consistent with the issuer's public parameters. In particular they prevent the issuer from fingerprinting users by using a unique key per user. In this section, we consider a weaker notion of unlinkability, which guarantees that the user either receives a valid token, or a completely random value (unpredictable by the issuer). This implies that the issuer *can* distinguish valid tokens from invalid tokens since the user cannot verify herself whether they have a valid token or not. In other words, the issuer can partition the users into two sets: one that receives valid tokens, and one that receives invalid tokens. The issuer will be able to identify which of these sets a user belongs to, at redemption time. For a more careful analysis on how this affect the success probability of the adversary in the unlinkability game, we refer the reader to the full version [KLOR20].

7.1 PP without issuance NIZK

We start with our new construction for the functionality of Privacy Pass. The change that we make is that the user blinds her token hash $\mathsf{H}_t(t)$ using both multiplicative and additive blinding. The additive part can be removed during the unblinding if the issuer used the correct secret key. Otherwise, the generated token $(t, W) \in \{0,1\}^\lambda \times \mathbb{G}$ will be invalid and distributed uniformly at random.

Construction 4 (Privacy Pass without issuance NIZK). Let GrGen be a group generator algorithm; let Π_{DLOG} be a proof system for relation $\mathsf{R}_{\mathsf{DLOG}}$; let H_t be a random oracle $\{0,1\}^* \to \mathbb{G}$. We construct an anonymous token scheme PPB defined by the following algorithms:

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{PPB.Setup}(1^\lambda)$: invoke the group generator $\Gamma \leftarrow \mathsf{GrGen}(1^\lambda)$ and the CRS generation algorithm of the underlying proof system $(\mathsf{pcrs}, \mathsf{ptd}) \leftarrow \Pi_{\mathsf{DLOG}}.\mathsf{Setup}(\Gamma)$. Return $\mathsf{crs} := (\Gamma, \mathsf{pcrs})$ and $\mathsf{td} := \mathsf{ptd}$.
- $((X, \pi), x) \leftarrow \mathsf{PPB.KeyGen}(\mathsf{crs})$: sample the secret key $x \leftarrow_\$ \mathbb{Z}_p^*$. Define $\pi \leftarrow \Pi_{\mathsf{DLOG}}.\mathsf{Prove}(\mathsf{pcrs}, X, x)$. Return the public parameters (X, π), and the secret key x.

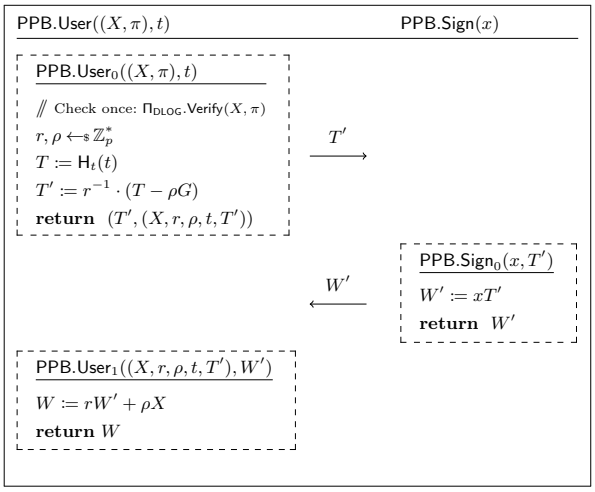

Fig. 9. Token issuance for PPB (Construction 4).

– $W \leftarrow \langle \mathsf{PPB.User}((X,\pi),t), \mathsf{PPB.Sign}(x)\rangle$: illustrated in Figure 9.
– $bool \leftarrow \mathsf{PPB.Verify}(x,t,W)$: return **true** if $W = x\mathsf{H}_t(t)$; else, return **false**.

Correctness. By correctness of Π_{DLOG}, at the end of the protocol the user returns $\perp$ only with negligible probability. If the user returns $W \in \mathbb{G}$, then W always satisfies the verification equation, since: $W = rW' + \rho X = rxr^{-1}(T - \rho G) + \rho X = xT - \rho(xG) + \rho X = xT$.

Security. PPB satisfies unforgeability and 2-unlinkability.

Theorem 11. *If CTGDH holds for* GrGen*, and* Π_{DLEQ} *is a zero-knowledge proof system for realtion* Π_{DLOG}*, then* PPB *is one-more unforgeable with advantage:*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{PPB},\ell}(\lambda) \leq \mathsf{Adv}^{\mathrm{ctgdh}}_{\mathsf{GrGen},\ell}(\lambda) + \mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLOG}},\mathsf{R}_{\mathsf{DLOG}}}(\lambda).$$

The proof can be found in the full version. Intuitively, unforgeability must hold because the issuer is sending strictly less information than in PP. The adversary A in game $\mathrm{OMUF}_{\mathsf{PPB},\mathsf{A}}(\lambda)$ takes as input the pair (X,π), where $X \in \mathbb{G}$ is the CTGDH challenge, and π is a DLOG proof. By zero-knowledge of Π_{DLOG}, the proof can be simulated; the adversary is now asked to return $\ell+1$ pairs (t_i, W_i), all different, such that W_i is the CDH of $(X, \mathsf{H}_t(t_i))$. During its execution, the adversary has at disposal the random oracle H_t (which behaves exactly as the TARGET oracle), the VERIFY oracle, which given as input $(Y,W) \in \mathbb{G}^2$ returns 1 if (X,Y,W) is a DH tuple (just as the DDH oracle in the game for CTGDH), and the SIGN oracle, which computes at most ℓ times the CDH with an arbitrary group element (just as the HELP oracle in the game for CTGDH).

Theorem 12. *Let* GrGen *be a group generator. If* Π_{DLOG} *is a knowledge-sound proof system for relation* $\mathsf{R}_{\mathsf{DLOG}}$*, then* PPB *is 2-unlinkable with advantage:*

$$\mathsf{Adv}^{\mathrm{unlink}}_{\mathsf{PPB},m}(\lambda) \leq \frac{2}{m} + \mathsf{Adv}^{\mathrm{ksnd}}_{\Pi_{\mathsf{DLOG}},\mathsf{R}_{\mathsf{DLOG}}}(\lambda).$$

First of all, we note that by knowledge soundness of Π_{DLOG}, for any adversary A that produces th public parameters (X, π) with $X \in \mathbb{G}$, it is possible to extract a witness $x \in \mathbb{Z}_p$ such that $xG = X$, except with negligible probability $\mathsf{Adv}^{\mathrm{ksnd}}_{\Pi_{\mathsf{DLOG}},\mathsf{R}_{\mathsf{DLOG}},\mathsf{A}}(\lambda)$. We use this witness to partition the sessions in two sets, U_0 where W is correctly computed, and U_1 where W isn't. In the latter case, we remark that because the additive blinding $\rho \in \mathbb{Z}_p$ is sampled uniformly at random, in U_1 all tokens are distributed uniformly at random. In a similar way to Theorem 7, we select two sessions k and j from the same U_b (for a $b \in \{0, 1\}$ that sampled at random, following the distribution of the open sessions) and swap them. If k and j are in U_0, unlinkability follows the same reasoning used for proving unlinkability of PP. If k and j are in U_1, then both tokens are uniformly random elements in $\mathbb{G}$ independent from the elements used at issuance time.

User Verifiability. The protocol presented in the previous section does not enable the user to verify that she has received valid token at the end of an execution. We can enable such verifiability for any number of tokens at the cost of one additional issuance interaction between the user and the issuer. In particular, let $(t_i, W_i)_{i\in[m]}$ be m tokens that the user has been issued. She sends a token issuance request $T' = \sum_{i\in[m]} c_i \mathsf{H}_t(t_i)$ where $c_i \leftarrow_{\$} \mathbb{Z}_p$ for $i \in [m]$. Let W be the issuer's response after unblinding, then the user checks that $W = \sum_{i\in[m]} c_i W_i$.

If the issuer was honest, then $W_i = x\mathsf{H}_t(t_i)$ and

$$W = x\left(\sum_{i\in[m]} c_i \mathsf{H}_t(t)\right) = \sum_{i\in[m]} c_i(x\mathsf{H}_t(t_i)) = \sum_{i\in[m]} c_i W_i\,.$$

Next, we argue that if $W = \sum_{i\in[m]} c_i W_i$, then the issuer could be cheating on any of the m token executions only with negligible probability. We will prove this by induction on m. Let $m = 1$, then we have $W = c_1 W_1$ and at the same time $W_1 \neq x\mathsf{H}_t(t_1)$. By the unlinkability argument above we know that W_1 and hence $c_1 W_1$ are uniformly distributed. Hence, the adversary has only negligible probability to guess the value W.

Now, let us assume that the statement holds for $m \leq k$ and we will be prove it for $m = k+1$. We have $W = \sum_{i\in[m]} c_i W_i$ and at the same time there exists an index j such that $W_j \neq x\mathsf{H}_t(t_j)$. If there is an index k such that $W_k = x\mathsf{H}_t(t_k)$, then $W - x\mathsf{H}_t(t_k) = \sum_{i\in[m]\setminus\{k\}} c_i W_i$ and $j \in [m]\setminus\{k\}$, which contradicts the induction assumption. Therefore, it must be the case that $W_i \neq x\mathsf{H}_t(t_i)$ for any $i \in [m]$. However, by the arguments in the unlinkability proof, we know that all W_i's, and hence $\sum_{i\in[m]} c_i W_i$, will be distributed uniformly at random. Hence, the adversary has only negligible probability in guessing the value of W, which concludes the inductive step.

7.2 PMBT without issuance NIZK

The challenge to generalizing the construction of the previous section to the setting of private metadata is that the user should not find out what metadata bit value the issuer used and hence which public key it should use when unblinding. Our solution will be to have the user run the unblinding with both keys where only one of the resulting values will be a valid token under the corresponding key for the bit value while the other unblinded value will be completely random. When we do this we need to be careful that the issuer who also generates the public keys should not be able to make the two unblinded values correlated, which would open an avenue for fingerprinting.

To guarantee that the unblinded value with the public key that does not correspond to the embedded private metadata bit is random and hence it is independent of the other unblinded value, even in the case when the issuer is misbehaving, we will need to have that the user generate two independent blinded values which it sends in its first message. The issuer will be using only one of the received blinded tokens to sign and embed his metadata bit, however, the user will be unblinding the message coming from the issuer using two independent sets of blinding parameters, which would thwart the issuer from embedding correlations.

Construction 5 (PMBTokens without issuance NIZK). Let GrGen be a group generator algorithm; let Π_{DLOGAND2} be a proof system for the relation $\mathsf{R}_{\mathsf{DLOGAND2}}$; let $\mathsf{H}_t, \mathsf{H}_s$ be two random oracles $\{0,1\}^* \to \mathbb{G}$. We construct an anonymous token scheme PMBTB defined by the following algorithms:

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{PMBTB.Setup}(1^\lambda)$: invoke the group generator $\Gamma \leftarrow \mathsf{GrGen}(1^\lambda)$ and the CRS generation algorithm $(\mathsf{pcrs}, \mathsf{ptd}) \leftarrow \Pi_{\mathsf{DLOGAND2}}.\mathsf{Setup}(\Gamma)$. Return $\mathsf{crs} := (\Gamma, \mathsf{pcrs})$ and $\mathsf{td} := \mathsf{ptd}$.
- $((\mathbf{X}, \pi), (\mathbf{x}, \mathbf{y})) \leftarrow \mathsf{PMBTB.KeyGen}(1^\lambda)$: let $(\mathbf{x}, \mathbf{y}) \leftarrow\!\!\$\, (\mathbb{Z}_p^*)^2 \times (\mathbb{Z}_p^*)^2$ be the secret key. Define:

$$\mathbf{X} := \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} := \begin{bmatrix} x_0 G + y_0 H \\ x_1 G + y_1 H \end{bmatrix},$$

  and let $\pi \leftarrow \Pi_{\mathsf{DLOGAND2}}.\mathsf{Prove}(\mathsf{pcrs}, \mathbf{X}, (\mathbf{x}, \mathbf{y}))$. The public parameters are $(\mathbf{X}, \pi)$. Return $((\mathbf{X}, \pi), (\mathbf{x}, \mathbf{y}))$.
- $\sigma \leftarrow \langle \mathsf{PMBTB.User}((\mathbf{X}, \pi), t), \mathsf{PMBTB.Sign}((\mathbf{x}, \mathbf{y})) \rangle$: illustrated in Figure 10.
- $\mathit{bool} \leftarrow \mathsf{PMBTB.Verify}((\mathbf{x}, \mathbf{y}), t, \sigma)$: return **true**.
- $\mathsf{ind} \leftarrow \mathsf{PMBTB.ReadBit}((\mathbf{x}, \mathbf{y}), t, \sigma)$: read σ as $(S_0, S_1, W_0, W_1) \in \mathbb{G}^4$. Then,
  (a) if $W_0 = x_0 \mathsf{H}_t(t) + y_0 S_0$ and $W_1 \neq x_1 \mathsf{H}_t(t) + y_1 S_1$, return 0;
  (b) if $W_0 \neq x_0 \mathsf{H}_t(t) + y_0 S_0$ and $W_1 = x_1 \mathsf{H}_t(t) + y_1 S_1$, return 1;
  (c) else, return $\perp$.

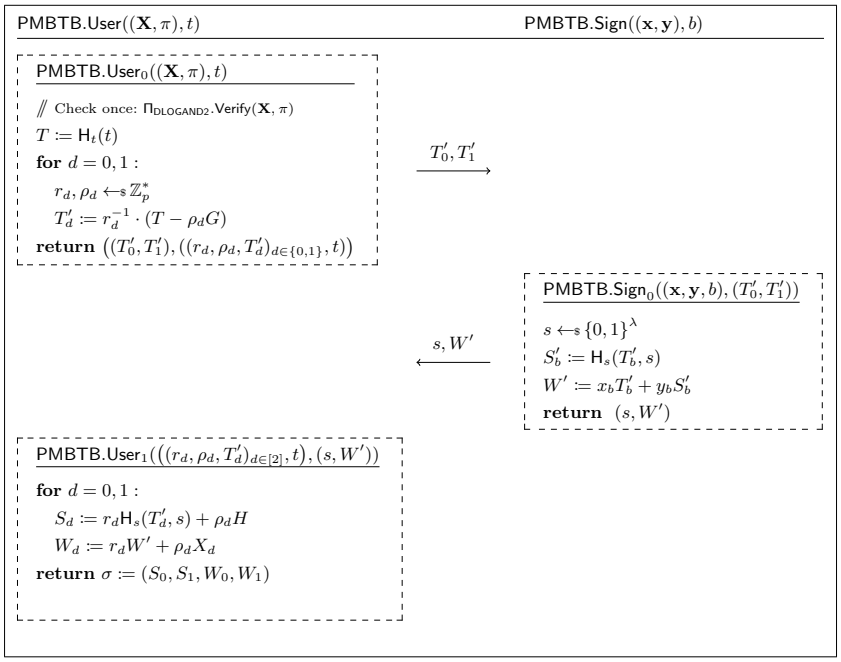

Fig. 10. Token issuance for PMBTB (Construction 5).

Correctness. Honestly-generated tokens are always valid, because the verification algorithm PMBTB.Verify always outputs **true**. We thus focus on Equation (7). By correctness of the underlying proof system, the protocol aborts only with negligible probability. If the user returns a tuple $(S_0, S_1, W_0, W_1) \in \mathbb{G}^4$, then there exists $b \in \{0, 1\}$ such that:

$$\begin{aligned} W_b &= r_b W' + \rho_b X_b = r_b(x_b T'_b + y_b S'_b) + \rho_b X_b \\ &= r_b x_b (r_b^{-1}(T - \rho_b G)) + y_b r_b \mathsf{H}_s(T'_b, s) + \rho_b X_b \\ &= x_b T + y_b (r_b \mathsf{H}_s(T'_b, s) + \rho_b H) \\ &= x_b T + y_b S_b. \end{aligned}$$

The probability that the above equation holds for both $b = 0$ and $b = 1$ (in which case, PMBTB.ReadBit returns $\perp$) is statistically negligible. In fact, if:

$$W_0 = x_0 \mathsf{H}_t(t) + y_0 S_0 = x_1 \mathsf{H}_t(t) + y_1 S_1 = W_1,$$

we have two possible cases:

(a) $x_0 = x_1$, which in turn implies that: $S_0 = {y_1}^{-1} y_0 S_1$. However, because S_0 is distributed uniformly at random in $\mathbb{G}$, this happens with probability $1/p$.

(b) $x_0 \neq x_1$, which in turn implies that $\mathsf{H}_t(t) = \frac{1}{x_1 - x_0}(y_0 S_0 - y_1 S_1)$, which happens with probability $1/p$.

Security. We provide the proofs for the security properties of the construction in the full version.

Theorem 13. *If CTGDH holds for the group generator algorithm* GrGen *and* Π_{DLOGAND2} *is a zero-knowledge proof system for* $\mathsf{R}_{\mathsf{DLOGAND2}}$*, then* $\mathsf{PMBTB}[\mathsf{GrGen}, \Pi_{\mathsf{DLOGAND2}}]$ *is one-more unforgeable with advantage:*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{PMBTB},\ell}(\lambda) \leq 2\mathsf{Adv}^{\mathrm{ctgdh}}_{\mathsf{GrGen},\ell}(\lambda) + \mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLOGAND2}},\mathsf{R}_{\mathsf{DLOGAND2}}}(\lambda).$$

Inuitively, unforgeability follows a similar reasoning of Theorem 8 (unforgeability of PMBT) except that here, at issuance time, we are sending strictly less information to the user, since we removed the NIZK at issuance time. The proof of knowledge of the discrete log, published at the beginning within the public parameters, can be simulated by zero knowledge.

Theorem 14. *If DDH holds for the group generator* GrGen *and* Π_{DLOGAND2} *is a knowledge-sound proof system for relation* $\mathsf{R}_{\mathsf{DLOGAND2}}$*, then* $\mathsf{PMBTB}[\mathsf{GrGen}, \Pi_{\mathsf{DLOGAND2}}]$ *is* 3*-unlinkabile.*

A PPT adversary in the game $\mathrm{UNLINK}_{\mathsf{PMBTB},\mathsf{A}}(\lambda)$ can now: compute W' using (x_0, y_0), using (x_1, y_1), or yet another key. Specifically in the latter case, the token σ will be distributed at random independently from W'. In the full version, we prove that now the adversary can partition the set of open sessions at most in 3, and that therefore the advantage in the game $\mathrm{UNLINK}_{\mathsf{PMBTB}}(\lambda) \leq 3/m + \mathsf{negl}(\lambda)$.

Theorem 15. *If DDH holds for the group generator* GrGen *and* Π_{DLOGAND2} *is a zero-knowledge proof system for relation* $\mathsf{R}_{\mathsf{DLOGAND2}}$*, then* $\mathsf{PMBTB}[\mathsf{GrGen}, \Pi_{\mathsf{DLOGAND2}}]$ *provides private metadata bit with advantage:*

$$\mathsf{Adv}^{\mathrm{pmb}}_{\mathsf{PMBTB}}(\lambda) \leq \frac{O(q^2)}{2^\lambda} + 2\mathsf{Adv}^{\mathrm{ddh}}_{\mathsf{GrGen}}(\lambda) + 4\mathsf{Adv}^{\mathrm{zk}}_{\Pi_{\mathsf{DLOGAND2}},\mathsf{R}_{\mathsf{DLOGAND2}}}(\lambda),$$

where q *is the number of queries the adversary makes either to* H_s *or* $\textsc{Sign}$*.*

As for unforgeability, the proof follows a similar reasoning to the case of PMBT. We notice that now the issuer is sending strictly less information during signing, and that the zero-knowledge proof π can be simulated.

8 Implementation

We implemented our construction in pure Rust (stable, version `1.41.0`), The second generator $H \in \mathbb{G}$ is chosen by hashing into the group the public generator G. Hashing into the group is done with a Elligator 2 map [Tib14] with SHA-512. Our implementation is not copyrighted and is released in the public domain.[10]

[10] https://www.di.ens.fr/~orru/anonymous-tokens

Table 2. Benchmarks for our constructions.

Constructions	DLEQ/DLEQOR		User		Issuer		
	Prove	Verify	Token Gen.	Unblinding	Key Gen.	Signing	Redemption
PP [DGS+18]	212 µs	181 µs	111 µs	286 µs	84 µs	303 µs	95 µs
PMBT	576 µs	666 µs	135 µs	844 µs	234 µs	845 µs	235 µs
PPB	–	–	197 µs	164 µs	190 µs	87 µs	95 µs
PMBTB	–	–	368 µs	678 µs	512 µs	155 µs	247 µs

We benchmarked our own implementation on a single thread of an `Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz`, running Ubuntu 18.04.3 LTS (kernel version 4.15.0). They are summarized in Table 2. As expected, Constructions 4 and 5 feature very fast issuance time at a slight increase in the user computation. Our results are between ten and one thousand faster than the previous implementation proposed in [DGS+18] due to the different choice[11] of elliptic curve (NIST P-256) as well as the programming language used.

Acknowledgments

The authors thank Fabrice Benhamouda, Dan Boneh, Charlie Harrison, Michael Kleber, Hugo Krawczyk, Steven Valdez, and Moti Yung for helpful discussions. A part this work was completed while Michele Orrù was an intern at Google and at Web3 Foundation.

References

ANN06. Michel Abdalla, Chanathip Namprempre, and Gregory Neven. On the (im)possibility of blind message authentication codes. In *CT-RSA 2006*, 2006.

BFPV11. Olivier Blazy, Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Signatures on randomizable ciphertexts. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 403–422. Springer, 2011.

BLS01. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, volume 2248 of *LNCS*, pages 514–532. Springer, 2001.

BMW03. Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, volume 2656 of *LNCS*, pages 614–629. Springer, 2003.

BNPS03. Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The one-more-rsa-inversion problems and the security of chaum's blind signature scheme. *J. Cryptology*, 16(3):185–215, 2003.

[11] We comment on the choice of curve and its security in the full version of this paper [KLOR20].

Bol03. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *Public Key Cryptography*, volume 2567 of *LNCS*, pages 31–46. Springer, 2003.

BPV12. Olivier Blazy, David Pointcheval, and Damien Vergnaud. Compact round-optimal partially-blind signatures. In *SCN*, volume 7485 of *LNCS*, pages 95–112. Springer, 2012.

Cam97. Jan Camenisch. Efficient and generalized group signatures. In *EUROCRYPT*, volume 1233 of *LNCS*, pages 465–479. Springer, 1997.

Cha82. David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, pages 199–203. Plenum Press, New York, 1982.

CL01. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT*, volume 2045 of *LNCS*, pages 93–118. Springer, 2001.

CMZ14. Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. Algebraic MACs and keyed-verification anonymous credentials. pages 1205–1216, 2014.

CP92. David Chaum and Torben P. Pedersen. Wallet databases with observers. In *CRYPTO*, volume 740 of *LNCS*, pages 89–105. Springer, 1992.

CvH91. David Chaum and Eugène van Heyst. Group signatures. In *Advances in Cryptology — EUROCRYPT '91*, 1991.

DGS+18. Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, 2018.

ECS+15. Adam Everspaugh, Rahul Chatterjee, Samuel Scott, Ari Juels, and Thomas Ristenpart. The pythia PRF service. In *USENIX Security Symposium*, pages 547–562. USENIX Association, 2015.

FHKS16. Georg Fuchsbauer, Christian Hanser, Chethan Kamath, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model from weaker assumptions. In *SCN*, volume 9841 of *LNCS*, pages 391–408. Springer, 2016.

FHS15. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In *CRYPTO (2)*, volume 9216 of *LNCS*, pages 233–253. Springer, 2015.

Fis06. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In *CRYPTO*, volume 4117 of *LNCS*, pages 60–77. Springer, 2006.

FPS19. Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures in the algebraic group model. Cryptology ePrint Archive, Report 2019/877, 2019.

GG14. Sanjam Garg and Divya Gupta. Efficient round optimal blind signatures. In *EUROCRYPT*, volume 8441 of *LNCS*, pages 477–495. Springer, 2014.

Gha13. Essam Ghadafi. Formalizing group blind signatures and practical constructions without random oracles. In *ACISP*, volume 7959 of *LNCS*, pages 330–346. Springer, 2013.

HL06. Javier Herranz and Fabien Laguillaumie. Blind ring signatures secure under the chosen-target-cdh assumption. In *ISC*, volume 4176 of *LNCS*, pages 117–130. Springer, 2006.

JKK14. Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In *ASIACRYPT (2)*, volume 8874 of *LNCS*, pages 233–253. Springer, 2014.

JKR18. Stanislaw Jarecki, Hugo Krawczyk, and Jason K. Resch. Threshold partially-oblivious prfs with applications to key management. *IACR Cryptology ePrint Archive*, 2018:733, 2018.

KLOR20. Ben Kreuter, Tancrede Lepoint, Michele Orru, and Mariana Raykova. Anonymous tokens with private metadata bit. Cryptology ePrint Archive, Report 2020/072, 2020. https://eprint.iacr.org/2020/072.

LR98. Anna Lysyanskaya and Zulfikar Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *Financial Cryptography*, 1998.

Oka92. Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *CRYPTO*, volume 740 of *LNCS*, pages 31–53. Springer, 1992.

OP01. Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In *Public Key Cryptography*, volume 1992 of *LNCS*, pages 104–118. Springer, 2001.

PS00. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13:361–396, 2000.

Ram13. Zulfikar Ramzan. Group blind digital signatures : theory and applications, 2013. https://dspace.mit.edu/bitstream/handle/1721.1/80561/43557700-MIT.pdf?sequence=2&isAllowed=y.

SC12. Jae Hong Seo and Jung Hee Cheon. Beyond the limitation of prime-order bilinear groups, and round optimal blind signatures. In *TCC*, volume 7194 of *LNCS*, pages 133–150. Springer, 2012.

Sch01. Claus Peter Schnorr. Security of blind discrete log signatures against interactive attacks. In Sihan Qing, Tatsuaki Okamoto, and Jianying Zhou, editors, *Information and Communications Security*, 2001.

Sch06. Claus Peter Schnorr. Enhancing the security of perfect blind dl-signatures. *Information Sciences*, 176(10):1305 – 1320, 2006.

TAKS07. Patrick P. Tsang, Man Ho Au, Apu Kapadia, and Sean W. Smith. Blacklistable anonymous credentials: blocking misbehaving users without ttps. In *ACM Conference on Computer and Communications Security*, pages 72–81. ACM, 2007.

Tib14. Mehdi Tibouchi. Elligator squared: Uniform points on elliptic curves of prime order as uniform random strings. In *Financial Cryptography*, volume 8437 of *LNCS*, pages 139–156. Springer, 2014.