Anonymous Tokens with Public Metadata and Applications to Private Contact Tracing

Tjerand Silde[1] and Martin Strand[2]

[1] Department of Mathematical Sciences,
Norwegian University of Science and Technology – NTNU,
tjerand.silde@ntnu.no
[2] Norwegian Defence Research Establishment – FFI,
martin.strand@ffi.no

Abstract. Anonymous tokens have recent applications in private Internet browsing and anonymous statistics collection. We develop new schemes in order to include public metadata such as expiration dates for tokens. This inclusion enables planned mass revocation of tokens without distributing new keys, which for natural instantiations can give 77–90 % amortized traffic savings compared to Privacy Pass (Davidson et al., 2018) and PrivateStats (Huang et al., 2021), respectively. By transforming the public key, we are able to append public metadata to several existing protocols without having to change the security proofs in any substantial way.
Additional contributions include expanded definitions and a description of how anonymous tokens can improve the privacy in dp^{3}t-like digital contact tracing applications. We also show how to create efficient and conceptually simple tokens with public metadata and public verifiability from pairings.

Keywords: anonymous tokens, public metadata, elliptic curves, pairings, private contact tracing, cryptography

1 Introduction

Anonymous communication has been an active research area since the 1980's, involving schemes like blind signatures, partially blind signatures, anonymous credentials and tokens, group signatures, ring signatures etc. This enables more complex systems for e.g. electronic cash or electronic voting, but also to protect the privacy of the users in chat applications like Signal.

Recent work by Davidson et al. [27] presents a very practical protocol, named Privacy Pass [26], for anonymous tokens. This protocol allows users to browse anonymously, e.g. using Tor, without having to solve a CAPTCHA every time they visit a website. Privacy Pass gives the user a set of randomized tokens

whenever they solve a CAPTCHA, which they then later can redeem instead of solving a new CAPTCHA. This improves the usability of anonymous browsing, at the same time as it gives protection against spam, prevents DDoS attacks and provides fraud resistance without the need for cross-site tracking or fingerprinting. However, the only way to revoke batches of tokens is by replacing the private-public key pair, which is impractical [25].

Privacy Pass has gained a lot of attention, and is currently being integrated to improve privacy in several applications, e.g., in PrivateStorage[3] and for basic attention tokens (BATs) in the Brave browser[4]. It can also be used for private click measurement when doing a purchase or signing up for a service[5].

Facebook uses partially blind signatures for combating fraud [38], and they have developed an extension of Privacy Pass called PrivateStats [36], which is used for privately collecting client-side telemetry from WhatsApp. PrivateStats requires daily key-rotation to prevent DoS attacks, which led them to develop an attribute-based verifiable oblivious pseudo-random function for more efficient and transparent key-rotation.

The IETF [37] is currently standardizing Privacy Pass, while Trust Token [49], a more generalized API based on Privacy Pass, is currently being standardized by the W3C as a part of the Google Privacy Sandbox project. Both standardization processes mention private and public metadata as desirable extensions to the Privacy Pass protocol. Public metadata allows for more efficient key-rotation, and opens for applications using public labeling and public anonymity sets, while private metadata allows for allow/deny lists, rate-limiting or trust-indication.

Kreuter et al. [41] gave the first construction of anonymous tokens with private metadata, while we give the first construction with public metadata. Our construction can also be combined with private metadata or public verifiability.

1.1 Background

The literature provides many flavors of anonymous credentials. They all come with some minimal requirements with respect to security, and have a variation of desirable properties for practical applications.

Unlinkability and Unforgeability To ensure privacy of users, the anonymous token protocol must make sure that the information being transferred in the attestation phase cannot be correlated with the token being received in the redemption phase. This is called unlinkability. To ensure the system's integrity, the anonymous token protocol must make sure that users cannot forge tokens,

[3] PrivateStorage: `https://medium.com/least-authority/the-path-from-s4-to-privatestorage-ae9d4a10b2ae`.

[4] Brave: `https://github.com/brave/brave-browser/wiki/Security-and-privacy-model-for-ad-confirmations`

[5] Private Click Measurement: `https://privacycg.github.io/private-click-measurement`.

even after receiving valid tokens from the attestation server. This is called unforgeability. These are the minimal requirements for anonymity and integrity, and have been handled starting from the first classical constructions of blind signatures starting with David Chaum in the early 1980's [20, 21] and subsequent work on anonymous credentials [4, 12, 13, 15–19, 33, 43, 45].

Underlying Primitives Anonymous token protocols can be built from a variety of cryptographic primitives and assumptions, e.g., factoring [1,15,16], discrete logarithms [2, 27, 41, 43, 50] or bilinear pairings [8, 23, 24, 51]. Protocols based on elliptic curve discrete logarithms are the most efficient, in terms of both size and timings, while other primitives might more easily provide correctness and verifiability.

Verifiability and Key-Sharing Subject to the situation, it might be desirable with either designated or public verifiability, depending on the redemption server having access to the secret key or not. If the same party is both attesting and redeeming tokens [27, 41], it is natural for the server(s) to share a key. In the designated verifier setting, it is necessary for the attestation server to provide zero-knowledge proofs to ensure that a token is honestly generated [27,41], while pairings easily can provide public verifiability [51].

Key-Rotation and Token-Revocation To avoid misuse, we need a way to efficiently revoke batches of tokens. This may be useful for rate limiting to avoid denial-of-service attacks, or for protecting users from credential stuffing [27, 36, 47]. In Privacy Pass [27], this is solved by infrequent key-rotation where a few public keys are available at a public endpoint. PrivateStats [36], who rotates their keys every day, solve it by using a new attribute-based verifiable oblivious pseudorandom function (VOPRF). However, both solutions are inconvenient in practice, and adds significant overhead for validating the public keys.

Rounds of Interaction Blind signatures and anonymous tokens can be attested in only one round of communication, which is optimal. This saves time and computation for both the client and the server, and the server does not need to keep a state. However, the only partially blind signatures achieving one round of interaction are based on bilinear pairings or factoring [1], while protocols based on discrete logarithms [2, 50] needs two rounds. We remark that there is no one-round anonymous token protocol with efficient revocation in the literature based on elliptic curve discrete logarithms without pairings before this work.

1.2 Our Contribution

Our contribution in this work is threefold: First, we present new definitions for anonymous tokens – extending the work by Kreuter et al. [41] – to also consider

public metadata and/or public verifiability. Secondly, we present three efficient protocols for anonymous tokens with efficient batched revocation: 1) Privacy Pass [27] with public metadata, 2) Kreuter et al. [41] with public and private metadata, and 3) a Privacy Pass inspired protocol using pairings to achieve public verifiability at the same time as it includes public metadata. Thirdly, we present contact tracing as a new concrete application for anonymous tokens, and discuss a concrete implementation used in the Norwegian contact tracing app.

Updated Definitions Several works have asked for efficient batched revocation of anonymous tokens without key-rotation [25, 27]. Additionally, there is a need for anonymous tokens with public verifiability [49], so that token generation can be delegated, and verification can be performed locally for token redemption. We provide updated definitions for all of these cases: designated verifier anonymous tokens with or without public and/or private metadata and public verifier anonymous tokens with and without public metadata. Details can be found in Section 3.

Anonymous Tokens with Public Metadata We present the first anonymous tokens protocols with efficient batched revocation, meaning that the protocol only requires one round of communication based on lightweight primitives and that we avoid key-rotation. The key insight in our protocol is conceptually very simple: all parties locally update the public key based on the hash of the public metadata, and then execute the protocols with respect to the new key pair. The main challenge is to sign tokens in a way that does not allow the user to forge tokens initially signed under metadata md to be valid under metadata md' instead. Let k be the secret key and let $d = \mathtt{H}(\mathsf{md})$ be the hash of the metadata. Our solution, inspired by Zhang et al. [51], is to use the inverse $e = (d + k)^{-1}$ as the new signing key. This allows us to directly replace the secret keys in the previous protocols, and the security proofs proceed in the same manner as earlier.

Further, to avoid subliminal channels, the signer needs to prove that the signed token is computed correctly. This is easily solved for Privacy Pass [27]. In the original protocol they use a zero-knowledge protocol to prove, given generator G, public key $K = [k]G$, blinded token T' and signed token $W' = [k]T'$, the equality of discrete logarithms $\log_G K = k = \log_{T'} W'$ to ensure correctness. In our updated protocol, including metadata md, updated public key $U = [d]G + K$ and signed token $W' = [e]T'$, we prove the equality of discrete logarithms $\log_G U = d + k = \log_{W'} T'$ to ensure correctness.

However, it is not as easy to ensure correctness in the extended version of the protocol by Kreuter et al. [41] including both public and private metadata. We solve this by combining an `OR`-proof with two `AND`-proofs to make sure that the right key is used. Further improvement is an open problem.

Next, we give a protocol based on pairings. The protocol is an adapted version of the partially blind signatures by Zhang et al. [51], where we tweak it into the same structure as Privacy Pass by using asymmetric pairings instead of

symmetric. We note that the communication in the protocol is the same, but in addition to get a more streamlined protocol structure, we also allow for more efficient instantiation in practice using the BLS12-381 pairing [5]. Ideally, we would like to avoid pairings altogether, but this seems necessary in practice. See more details about the protocols in Section 4.

Finally, we compare the efficiency of the protocols in Section 5, and compare our constructions with the current state of the art with respect to efficient batched revocation in Table 1. We show that our protocols are much more efficient in practice. We also make a concrete comparison with PrivateStats [36] for collecting telemetry-data from WhatsApp, and show that our protocol in Figure 5 would decrease the size of the signed token by 90 %, saving the Facebook servers up to 1.7 TB of communication every day.

More Private Contact Tracing Many countries have recently developed contact tracing apps as one of the measurements to battle the ongoing pandemic. These apps are inherently storing sensitive information about the user, e.g., the user's location graph and social graph. To avoid large, centralized databases with such sensitive information about a large portion of a country's adult population, most apps are based on the decentralized Google/Apple Exposure Notification System (ENS). However, there are still privacy issues with regards to uploading the randomized exposure keys to the central server, as the user would have to identify themselves to ensure that only people who have tested positive for COVID-19 are able to upload keys. We implemented Privacy Pass into the Norwegian contact tracing app to improve the user's privacy. Our code is published at `https://github.com/HenrikWM/anonymous-tokens`, and the Norwegian Institute of Public Health (NIPH) have made the source code for the contact tracing infrastructure publicly available[6]. We present more details about the contact tracing infrastructure and improvements to the app in Section 6.

1.3 Relevant work

Our work achieving designated verification and public metadata extends a long line of publications. Freedman et al. [29] introduced oblivious pseudo-random functions, and Jarecki et al. [39,40] gave an efficient instantiation based on DDH in the random oracle model. Papadopoulos et al. [44] gave a verifiable PRF from elliptic curves, and Burns et al. [11] gave an oblivious PRF from elliptic curves. Privacy Pass combined these results with an extended version of the Chaum-Pedersen zero-knowledge protocol [22] given by Henry and Goldberg [34, 35] to prove knowledge of batches of elements having the same discrete logarithm, and Kreuter et al. [41] added private metadata to Privacy Pass.

To achieve public verifiability we use parings, inspired by the seminal work of Boneh et al. [10] for short and efficient signatures and a series of constructions of anonymous credentials based on pairings [14, 17, 23, 24, 30, 31, 51].

[6] NIPH: `http://github.com/folkehelseinstituttet/?q=Smittestopp`

2 Preliminaries

We assume that the reader is familiar with the basics of elliptic curve cryptography. To fix notation, let q be a prime and let $\mathbb{F}_{q^\ell}$ for some $\ell > 0$ be a field of characteristic q. Let E be all points (x, y) that satisfy the elliptic curve equation $y^2 = x^3 + ab + b$ in the algebraic closure of $\mathbb{F}_{q^\ell}$, and let $E(\mathbb{F}_{q^\ell})$ denote the set of all such points in $\mathbb{F}_{q^\ell} \times \mathbb{F}_{q^\ell}$ along with the point at infinity $\mathcal{O}$. By abuse of notation, we often let E be a group of order p inside $E(\mathbb{F}_{q^\ell})$. Define the group law in the usual additive way. In particular, let $[m] : E \to E$ be the multiplication-by-m map, which takes the same role as exponentiation in multiplicative groups. Now follows a brief discussion of the Chosen-Target Gap Diffie-Hellman problem and some zero-knowledge proofs we will need as primitives.

2.1 DDH vs. CDH in Pairings

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two additive cyclic groups of prime order, and let $\mathbb{G}_T$ be another cyclic group of same prime order, written multiplicatively. A pairing $\hat{e}$ is a map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$$

such that the following properties hold:

Bilinearity For all $P_1, P_2 \in \mathbb{G}_1$ and $Q_1, Q_2 \in \mathbb{G}_2$, it holds that $\hat{e}(P_1 + P_2, Q_1) = \hat{e}(P_1, Q_1)\hat{e}(P_2, Q_1)$ and $\hat{e}(P_1, Q_1 + Q_2) = \hat{e}(P_1, Q_1)e(P_1, Q_2)$.
Non-degeneracy For all $P \neq \mathcal{O}$, $\hat{e}(P, P) \neq 1$.
Computability $\hat{e}$ can be efficiently computed.

The bilinearity property implies that for any scalars a, b, we have $\hat{e}([a]P, [b]Q) = \hat{e}(P, Q)^{ab}$, which is the crucial property used for verification later. Galbraith, Paterson and Smart [32] sort pairings into three categories:

I $\mathbb{G}_1 = \mathbb{G}_2$, where $\phi : \mathbb{G}_2 \to \mathbb{G}_1$ is the identity.
II $\mathbb{G}_1 \neq \mathbb{G}_2$, but there is an efficiently computable homomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$.
III $\mathbb{G}_1 \neq \mathbb{G}_2$ with no efficiently computable homomorphisms ϕ between the groups.

Bilinear maps lend themselves to a variant of the well-known Diffie-Hellman problem, the Chosen-Target Gap Diffie-Hellman problem [6]. Even if the adversary is given oracle access to ℓ instances of the Computational Diffie-Hellman (CDH) problem and arbitrary many queries to a Decision Diffie-Hellman (DDH) oracle, it should still not be able to compute the final Diffie-Hellman instance $\ell + 1$. We repeat the game and definition by Kreuter et al. [41].

Definition 1 (Chosen-Target Gap Diffie-Hellman). *Let $\mathbb{G}$ be a cyclic group of order p with generator G produced by the algorithm $\mathsf{Gen}(1^\lambda)$. Let* CTGDH *be the game defined in Figure 1. Chosen-Target Gap Diffie-Hellman holds for $\mathbb{G}$ if for any* PPT *adversary $\mathcal{A}$ and any $\ell \geq 0$,*

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

We will later base our security on the assumption that this problem is hard.

Game $\mathrm{CTGDH}_{\mathsf{Gen},\mathcal{A},\ell}(\lambda)$	Oracle $\textsc{Target}(t)$	Oracle $\textsc{Help}(Y)$
$\Gamma = (\mathbb{G}, p, G) \leftarrow \mathsf{Gen}(1^\lambda)$	**if** $t \in \mathcal{Q}$ **then**	$q := q + 1$
$x \leftarrow\!\$\, \mathbb{Z}_p$; $X := [x]G$	$Y := \mathcal{Q}[t]$	**return** $[x]Y$
$q := 0$; $\mathcal{Q} := []$	**else**	Oracle $\mathrm{DDH}(Y, Z)$
$(t_i, Z_i)_{i\in[\ell+1]} \leftarrow \mathcal{A}^{\textsc{Target},\textsc{Help},\mathrm{DDH}}(\Gamma, X)$	$Y \leftarrow\!\$\, \mathbb{G}$	**return** $(Z = [x]Y)$
for $i \in [\ell + 1]$	$\mathcal{Q}[t] := Y$	
if $t_i \notin \mathcal{Q}$ **then return** 0	**return** Y	
$Y_i := \mathcal{Q}[t_i]$		
return $(q \leq \ell$ **and**		
$\forall i \neq j \in [\ell + 1], t_i \neq t_j$ **and**		
$\forall i \in [\ell + 1], [x]Y_i = Z_i)$		

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

2.2 Proof of Equal Discrete Logs

Chaum and Pedersen [22] introduced an elegant honest-verifier zero-knowledge protocol to prove that two group elements have the same discrete logarithm relative to their respective bases, $\log_G K = k = \log_T W$. We describe the protocol loosely to ensure the reader is familiar with the idea. Let $\mathbb{G}$ be an additive group of prime order p with independent generators G and T, and let $K := [k]G$, $W := [k]T$ where k is a number private to the verifier $\mathcal{V}$.

$\mathcal{P}$**.1** Choose a random number r in the underlying field, compute $A := [r]G, B := [r]T$ and send (A, B) to $\mathcal{V}$.
$\mathcal{V}$**.1** Choose a random number c and send it to $\mathcal{P}$.
$\mathcal{P}$**.2** Compute the response $z := r - ck$ modulo p, and then send z to $\mathcal{V}$.
$\mathcal{V}$**.2** Verify that $A = [z]G + [c]K$ and $B = [z]T + [c]W$.

This protocol satisfies unconditional special soundness and special honest-verifier zero-knowledge. One can make the protocol non-interactive by applying the Fiat-Shamir transformation [28]. The prover queries the oracle on the tuple $(\mathbb{G}, G, T, K, W, A, B)$. In addition, one can reduce communication by sending the oracle response c instead of (A, B), and modifying the final verification step to querying the oracle on $(\mathbb{G}, G, T, K, W, [z]G + [c]K, [z]T + [c]W)$, and then verify that it indeed returns c. We will use a shorthand notation to refer to this proof as $\Pi_{\mathrm{DLEQ}}(T, W; e)$, meaning that $\log_G([e]G) = \log_W T$.

The proof can be batched for many instances with respect to the same secret scalar using the techniques by Henry [34] as showed in [27, Section 3.2.1].

2.3 AND-Proof of Equal Discrete Logs

Let $\mathbb{G}$ be an additive group of prime order p with generators G, H, T, S, and let $K := [k_0]G + [k_1]H$ and $V := [k_0]T + [k_1]S$, where k_0, k_1 are numbers private to

the verifier $\mathcal{V}$. We want to prove that V is correctly computed with respect to T, S using the same secret numbers as K with respect to G, H. We present a simple protocol to prove this relation, by essentially computing two Chaum-Pedersen proofs in parallel.

$\mathcal{P}$.1 Choose two random numbers r_0, r_1 from the underlying field. Compute $A := [r_0]G, B := [r_1]H, C := [r_0]T, D := [r_1]S$, and send (A, B, C, D) to $\mathcal{V}$.
$\mathcal{V}$.1 Choose a random number c and send it to $\mathcal{P}$.
$\mathcal{P}$.2 Compute $z_0 := r_0 - ck_0$ and $z_1 := r_1 - ck_1$ in the field and send (z_0, z_1) to $\mathcal{V}$.
$\mathcal{V}$.2 Verify that $A + B = [c]K + [z_0]G + [z_1]H$ and $C + D = [c]V + [z_0]T + [z_1]S$.

It is straightforward to verify that this is a Σ-protocol with special soundness and special honest-verifier zero-knowledge. As above, we can apply the Fiat-Shamir [28] transformation to get a non-interactive protocol. We will refer to this proof as $\Pi_{\text{DLEQ2}}(V; k_0, k_1)$.

2.4 OR-Proof of Equal Discrete Logs

We present the honest-verifier zero-knowledge `OR`-proof of equal discrete logarithms instantiated by Kreuter et al. [42, Appendix B]. Let $\mathbb{G}$ be an additive group of prime order p with generators G, H, T, S, and let $V_0 := [e_{0,0}]G + [e_{0,1}]H$ and $V_1 := [e_{1,0}]G + [e_{1,1}]H$, where $e_{i,j}$ are distinct numbers private to the verifier $\mathcal{V}$. Furthermore, let $W := [e_{b,0}]T + [e_{b,0}]S$ for a $b \in \{0, 1\}$. We want to prove that W is computed using the same secret numbers as either V_0 or V_1.

$\mathcal{P}$.1 Choose random numbers $r_0, r_1, c_{b-1}, u_{b-1}, v_{b-1}$ in the underlying field and compute the following:

$$\begin{aligned} A_{b,0} &:= [r_0]G + [r_1]H, \\ A_{b,1} &:= [r_0]T + [r_1]S, \\ A_{1-b,0} &:= [u_{b-1}]G + [v_{b-1}]H - [c_{b-1}]V_{b-1}, \\ A_{1-b,1} &:= [u_{b-1}]T + [v_{b-1}]S - [c_{b-1}]W. \end{aligned}$$

Finally, send $(A_{0,0}, A_{0,1}A_{1,0}, A_{1,1})$ to $\mathcal{V}$.
$\mathcal{V}$.1 Choose a random number c and send it to $\mathcal{P}$.
$\mathcal{P}$.2 Compute the responses

$$c_b := c - c_{1-b}, \quad u_b := r_0 + c_b e_{b,0}, \quad v_b := r_1 + c_b e_{b,1},$$

in the underlying field. Send $(c_i, u_i, v_i)_{i=0,1}$ to $\mathcal{V}$.
$\mathcal{V}$.2 Verify that $c = c_0 + c_1$ and that

$$\begin{aligned} A_{0,0} &= [u_0]G + [v_0]H - [c_0]V_0, \\ A_{0,1} &= [u_0]T + [v_0]S - [c_0]W, \\ A_{1,0} &= [u_1]G + [v_1]H - [c_1]V_1, \\ A_{1,1} &= [u_1]T + [v_1]S - [c_1]W. \end{aligned}$$

Again, using Fiat-Shamir [28], we can make the proof non-interactive by evaluating a random oracle on input $(\mathbb{G}, G, H, T, S, W, A_{0,0}, A_{0,1}, A_{1,0}, A_{1,1})$ to generate the challenge c, and then the verifier can query the random oracle on the response from the verifier to ensure correctness and soundness. We will refer to this protocol as $\Pi_{\text{DLEQOR2}}(W; e_{b,0}, e_{b,1})$.

Π_{DLEQOR2} can be batched for many instances with respect to the same secret scalars using the techniques by Henry [34] as shown in [41, Appendix B.1].

3 Definitions for Anonymous Tokens

Anonymous tokens as used in Privacy Pass are conceptually simple: both issuance and verification require the private key, and the final token is uniquely determined by the token seed t and the private key. Kreuter et al. [41] extended this notion by adding a private bit in the token. We further extend the definition in two different directions: we want to add public metadata, and we want to make the token publicly verifiable. Now, private bits do not make immediate sense in the context of a publicly verifiable token scheme, but public metadata can be relevant in both settings.

The metadata can for instance be used to indicate an expiry date, replacing the need for frequent key rotation in certain applications [36] as we discussed in Section 1.1. We model it as a value that the user and issuer must agree upon, which should restrict the issuer from using an identifiable value.

Lending terminology from programming, we would like the definition to provide backwards compatibility, and handle the notational incompatibility between private and public verifiability. To this end, we imitate the notion of [optional arguments] from programming. The notation [$\mathsf{vk}|\mathsf{sk}$] is meant as "either the public or secret key, but at least one". We align our definitions as close as possible to those by Kreuter et al. [41].

Definition 2 (Anonymous tokens). *An anonymous token scheme with zero or more of the following features:*

- *private metadata bit*
- *public metadata*
- *public verifiability*

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 common reference string* crs *and trapdor* td*. All the remaining algorithms take* crs *as their first input, but we omit it for notational clarity.*
- $(\mathsf{pp}, \mathsf{sk}, [\mathsf{vk}]) \leftarrow \mathsf{AT.KGen}(\mathsf{crs})$*, the key generation algorithm that generates a signing key* sk *and a verification key* vk *along with public parameters* pp*.*
- $\sigma \leftarrow \langle \mathsf{AT.User}(\mathsf{pp}, [\mathsf{vk}], t, [\mathsf{md}]), \mathsf{AT.Sign}(\mathsf{sk}, [\mathsf{md}], [b]) \rangle$*, the token issuance protocol, which involves interactive algorithms* $\mathsf{AT.User}$ *and* $\mathsf{AT.Sign}$*. The user algorithm takes as input values the public parameters and the token seed*

$t \in \{0,1\}^\lambda$*, and potentially the verification key* vk *and the public metadata* md*. The signing algorithm takes the private key* sk *and potentially metadata* md *and the private bit* b*. At the end of the interaction, the issuer outputs nothing, while the user outputs* σ*, or* $\perp$ *to indicate error.*

- $bool \leftarrow \mathsf{AT.Vf}([\mathsf{vk}|\mathsf{sk}], t, [\mathsf{md}], \sigma)$*, the verification algorithm that takes as input either the public verification key* vk *or the private key* sk*, a token seed* t*, metadata* md *and the signature* σ*. It returns a bit indicating if the token was valid or not.*
- $[\mathsf{ind} \leftarrow \mathsf{AT.ReadBit}(\mathsf{sk}, t, [\mathsf{md}], \sigma)]$*, the private bit extraction algorithm that takes as input the private key* sk *and token* $(t, [\mathsf{md}], \sigma)$*. It returns an indicator* $\mathsf{ind} \in \{\perp, 0, 1\}$ *which is either the private bit, or* $\perp$*.*

The notation of the above definition should be interpreted in a global sense. If one – for example – wants to use public metadata, it should be included everywhere it is mentioned. As such, this listing then specifies the following six notions:

1. With designated verification:
   (a) Anonymous tokens
   (b) Anonymous tokens with private metadata bit
   (c) Anonymous tokens with public metadata
   (d) Anonymous tokens with public and private metadata
2. With public verification:
   (a) Anonymous tokens
   (b) Anonymous tokens with public metadata

Examples of 1a and 1b are well known from previous work. A previous example of 2b is known as a partially blind signature scheme. We will provide new examples of the last four (2a is implicit in 2b) in Section 4. To reduce notational clutter, we collectively refer to all of these as anonymous tokens.

We follow the convention of dividing the interactive protocol $\langle \mathsf{AT.User}, \mathsf{AT.Sign} \rangle$ into the non-interactive algorithms $\mathsf{AT.User}_0$, $\mathsf{AT.Sign}_0$ and $\mathsf{AT.User}_1$.

An anonymous token scheme must satisfy correctness, unforgeability and unlinkability:

Definition 3 (Token correctness). *An anonymous token scheme* AT *is correct if any honestly generated token verifies. For any honestly generated* crs*,* $(\mathsf{pp}, \mathsf{sk}, [\mathsf{vk}])$*,* t *and* $[\mathsf{md}]$*,*

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

We split correctness of the private metadata bit into a separate definition in order to reduce notational clutter. This definition only applies in the private-key setting, and the parameters have been fixed accordingly.

Game $\mathrm{OMUF}_{\mathsf{AT},\mathcal{A},\ell}(\lambda)$	Oracle $\textsc{Sign}(\mathsf{msg}, [\mathsf{md}], [b])$
$(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{AT.Setup}(1^\lambda)$	$q_b := q_b + 1$
$(\mathsf{pp}, \mathsf{sk}, [\mathsf{vk}]) \leftarrow \mathsf{AT.KGen}(\mathsf{crs})$	**return** $\mathsf{AT.Sign}_0(\mathsf{sk}, \mathsf{msg}, [\mathsf{md}], [b])$
for $b \in \{0,1\}, q_b := 0$	Oracle $\textsc{Verify}(t, [\mathsf{md}], \sigma)$
$(t_i, \sigma_i)_{i \in [\ell+1]} \leftarrow \mathcal{A}^{\textsc{Sign},\textsc{Verify},\textsc{Read}}(\mathsf{crs}, \mathsf{pp})$	**return** $\mathsf{AT.Vf}([\mathsf{sk}\|\mathsf{vk}], t, [\mathsf{md}], \sigma)$
return $(\forall b \in \{0,1\}, q_b \leq \ell$ **and**	Oracle $\textsc{Read}(t, \sigma)$
$\forall i \neq j$ **in** $[\ell+1]$ $t_i \neq t_j$ **and**	**return** $\mathsf{AT.ReadBit}(\mathsf{sk}, t, [\mathsf{md}], \sigma)$
$\forall i$ **in** $[\ell+1]$ $\mathsf{AT.Vf}([\mathsf{sk}\|\mathsf{vk}], t_i, \sigma_i) =$ **true and**	
$\exists b \in \{0,1\} : \forall i \in [\ell+1],$	
$\mathsf{AT.ReadBit}(\mathsf{sk}, t_i, \sigma_i) = b)$	

Fig. 2. One-more unforgeability with metadata.

Definition 4 (Correct private bit). *An anonymous token scheme* AT *is correct with respect to the private metadata bit if the correct bit is retrieved successfully, i.e. that*

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

No adversary should be able to redeem other tokens than those that have been correctly issued. The *one-more unforgeability* notion has become the common notion for anonymous credentials. It allows the adversary to claim ℓ tokens from the issuer, and the adversary should not be able to redeem $\ell + 1$ tokens. We require the tokens to be unique with respect to the value of the seed t.

Definition 5 (One-more unforgeability). *An anonymous token scheme* AT *is one-more unforgeable if for any* PPT *adversary* $\mathcal{A}$*, and any* $\ell \geq 0$*:*

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

where $\mathrm{OMUF}_{\mathsf{AT},\mathcal{A},\ell}$ *is the game defined in Figure 2.*

Next, we want to provide user anonymity. The right notion for this is unlinkability, which guarantees that even colluding issuers and verifiers are unable to link tokens. Metadata is a strong way of creating a link, and we omit this problem by only considering fixed public metadata for this notion. This is in line with for example expiry dates, which would otherwise have been solved in practice using key rotation, and the definition is (as usual) also using a fixed key. Private metadata is outside the control of the user, and is therefore included, in line with Krauter et al. [41].

Game $\text{UNLINK}_{\mathsf{AT},\mathcal{A},m,[\mathsf{md}]}(\lambda)$	Oracle $\textsc{User}_0()$
$(\mathsf{crs},\mathsf{td}) \leftarrow \mathsf{AT.Setup}(1^\lambda)$	$q_0 := q_0 + 1$
$(\mathsf{st},\mathsf{pp},[\mathsf{vk}]) \leftarrow \mathcal{A}(\mathsf{crs},[\mathsf{md}])$	$t_{q_0} \leftarrow\$ \{0,1\}^\lambda$
$q_0 := 0; q_1 := 0, \mathcal{Q} := \emptyset$	$(\mathsf{msg}_{q_0},\mathsf{st}_{q_0}) \leftarrow \mathsf{AT.User}_0(\mathsf{pp},[\mathsf{vk}],t_{q_0},[\mathsf{md}])$
$(\mathsf{st},(\mathsf{msg}_i)_{i\in\mathcal{Q}}) \leftarrow \mathcal{A}^{\textsc{User}_0,\textsc{User}_1}(\mathsf{st})$	$\mathcal{Q} := \mathcal{Q} \cup \{q_0\}$
if $\mathcal{Q} = \emptyset$ **then return** 0	**return** $(q_0, \mathsf{msg}_{q_0})$
$j \leftarrow\$ \mathcal{Q}; \mathcal{Q} = \mathcal{Q} \setminus \{j\}$	Oracle $\textsc{User}_1(j,\mathsf{msg})$
$\sigma_j \leftarrow \mathsf{AT.User}_1(\mathsf{st}_j,[\mathsf{vk}],\mathsf{msg}_j,[\mathsf{md}])$	**if** $j \notin \mathcal{Q}$ **then**
for $i \in \mathcal{Q}$	**return** $\perp$
$\sigma_i \leftarrow \mathsf{AT.User}_1(\mathsf{st}_i,[\mathsf{vk}],\mathsf{msg}_i,[\mathsf{md}])$	$\sigma \leftarrow \mathsf{AT.User}_1(\mathsf{st}_j,[\mathsf{vk}],\mathsf{msg},[\mathsf{md}])$
$\phi \leftarrow\$ \mathcal{S}_\mathcal{Q}$	**if** $\sigma \neq \perp$ **then**
$j' \leftarrow \mathcal{A}(\mathsf{st},(t_j,\sigma_j),(t_{\phi(i)},\sigma_{\phi(i)})_{i\in\mathcal{Q}})$	$\mathcal{Q} := \mathcal{Q} \setminus \{j\}$
return $q_0 - q_1 \geq m$ **and** $j' = j$	$q_1 := q_1 + 1$
	return σ

Fig. 3. Public-key unlinkability with fixed metadata. If X is a set, then $\mathcal{S}_X$ is the symmetric group of X.

Definition 6 (Unlinkability). *An anonymous token scheme* AT *is* κ*-unlinkable if for any* PPT *adversary* $\mathcal{A}$*, a fixed* md*, and any* $m > 0$*,*

$$\begin{aligned}\mathsf{Adv}^{\text{unlink}}_{\mathsf{AT},\mathcal{A},m,[\mathsf{md}]}(\lambda) &:= \Pr[\text{UNLINK}_{\mathsf{AT},\mathcal{A},m}(\lambda) = 1] \\ &\leq \frac{\kappa}{m} + \mathsf{negl}(\lambda),\end{aligned}$$

where $\text{UNLINK}_{\mathsf{AT},\mathcal{A},m}$ *is the game defined in Figure 3.*

We finally consider the private metadata bit. We give the adversary access to two signing oracles: One uses the adversary's chosen private bit, the other is using a fixed bit for the game. The adversary can also query a verification oracle. At the end of the game, the adversary outputs its guess for the fixed challenge bit.

Definition 7 (Private metadata bit). *An anonymous token scheme* AT *provides private metadata bit if for any* PPT *adversary* $\mathcal{A}$*,*

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

where $\text{PMB}^\beta_{\mathsf{AT},\mathcal{A}}$ *is the game defined in Figure 4.*

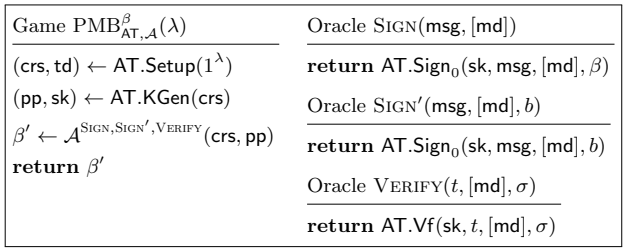

Fig. 4. Game for private metadata bits for anonymous tokens.

4 Anonymous Token Protocols

The Privacy Pass protocol [27] and its siblings [36, 41] are based on verifiable oblivious pseudo random functions (VOPRF). Here, a user holds some secret input x and the signer holds a secret key k and they evaluate the function F obliviously such that the user learns $F(x,k)$ but nothing about k, and the signer learns nothing about the input x nor the output $F(x,k)$. Additionally, the user is ensured that the function is evaluated by the correct secret key.

We construct three protocols for anonymous tokens (AT) with 1) public metadata, 2) public and private metadata, and 3) public metadata and public verifiability, respectively, constructed from the same framework.

At the core of our protocols lies a verifiable key transformation. Let $d := \mathtt{H}_m(\mathsf{md})$ and the curve point $U := [d]G + K$, where G is a public generator and K is the public key with a corresponding private key k. Let $e = (d+k)^{-1}$ and $W = [e]T$. Notice the relation

$$\mathsf{KT} : \log_G([d]G + K) = (d+k) = \log_W T. \tag{1}$$

The keys may consist of several such pairs. In order to break a scheme with transformed keys, one can either break properties of the original scheme, or one can break the integrity or privacy of the transformation process. We summarize this observation in a lemma:

Lemma 1. *Let* AT *be a scheme with keys* $(\mathsf{pk},\mathsf{vk})$ *with security property* P *within adversarial advantage* $\mathsf{Adv}^{\mathrm{p}}_{\mathsf{AT},\mathcal{A}}(\lambda)$*, and assume we can prove the relation in Equation 1 within adversarial advantage* $\mathsf{Adv}^{\mathrm{rel}}_{\mathsf{KT},\mathcal{A}}(\lambda)$*. Then* $\mathcal{A}$ *has advantage* $\mathsf{Adv}^{\mathrm{p}}_{\mathsf{AT},\mathcal{A}}(\lambda) + \mathsf{Adv}^{\mathrm{rel}}_{\mathsf{KT},\mathcal{A}}(\lambda)$ *against property* P *in the scheme* AT *with transformed keys* $(\{e = e(\mathsf{md},\mathsf{pk})\},\{[e]G\})$*.*

We note that [52, Theorem 1] states that the inverse computational Diffie-Hellman (ICDH) problem is as hard as plain CDH, and the security of the transform $e = (d+k)^{-1}$ as signing key instead of k follows.

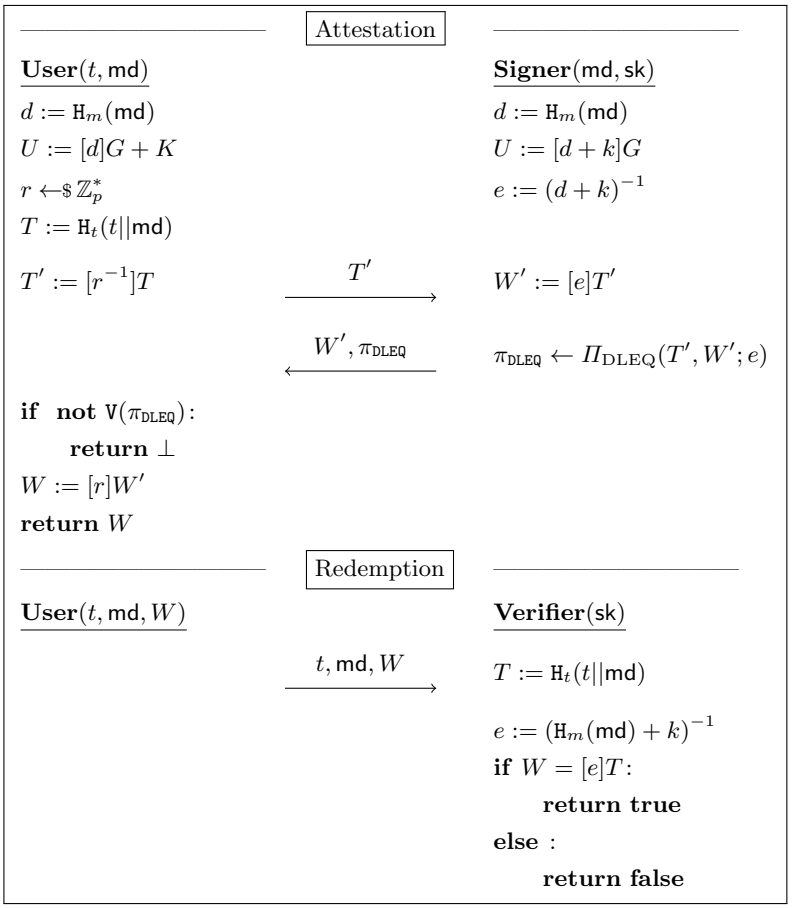

Fig. 5. Designated verifier anonymous tokens with public metadata. Our protocol is a direct extension of Privacy Pass [27].

4.1 AT with Public Metadata

In Figure 5 we present an extension of Privacy Pass [27] with public metadata. The protocol is designated verifier, as the secret key is needed to verify tokens.

Parameters Let λ be the security parameter, let p be a prime and let E be an elliptic curve group of order p with generator G. Let $\mathtt{H}_t : \{0,1\}^* \to E$ and $\mathtt{H}_m : \{0,1\}^* \to \mathbb{Z}_p$ be hash functions, and assume that group elements and integers can be encoded uniquely as strings. Furthermore, let metadata md be an element of a public set of valid strings. Finally, let $\mathsf{sk} := k \leftarrow\!\$\, \mathbb{Z}_p^*$ be the signing key, and let $\mathsf{pk} := K := [k]G$ be the public key. We consider $G, E, p, \mathtt{H}_t, \mathtt{H}_m$ and K to be implicit knowledge in Fig. 5.

Zero-Knowledge Protocol `DLEQ` The anonymous tokens protocol in Figure 5 uses the Π_{DLEQ}-protocol defined in Section 2.2. The signer computes a proof $\pi_{\texttt{DLEQ}} := (c, z)$ of equality of discrete logarithms by instantiating the protocol $\Pi_{\text{DLEQ}}(T', W'; e)$. Given the public parameters G and K, and $U := [d]G + K$, this is a proof that $\log_G U = d + k = \log_{W'} T'$. This proves that $W' = [e]T'$, where $e := (d + k)^{-1}$, is computed correctly with respect to d and K. To verify, the user instantiates the verification algorithm, denoted by $\texttt{V}(\pi_{\texttt{DLEQ}})$ in Figure 5.

Correctness and Security

Theorem 1 (Completeness). *The anonymous token protocol with public metadata in Figure 5 is complete according to Definition 3.*

Proof. The completeness follows from expanding W:

$$W = [r]W' = [r][e]T' = [r][e][r^{-1}]T = [e]\texttt{H}_t(t||\mathsf{md}).$$

Theorem 2 (Unforgeability). *The anonymous token protocol with public metadata in Figure 5 achieve one-more unforgeability with respect to Definition 5.*

Proof. Except for the updated public-private key-pair, the protocol is identical to Privacy Pass. The security against one-more unforgeability follows directly from Lemma 1 and [27, Theorem 2], with respect to U.

Theorem 3 (Unlinkability). *The anonymous token protocol with public metadata in Figure 5 achieve unlinkability with respect to Definition 6.*

Proof. This proof is identical to [27, Theorem 1]: As we sample $r \leftarrow\!\$\, \mathbb{Z}_p$ uniformly at random, it follows that our protocol is unconditionally unlinkable. Since T is a generator of E, then $T' = [r^{-1}]T$ is uniformly random and contain no information about t nor T. As the signer only sees T', and the verifier only receive t, and they are independent, there is no link between the view of the signer and the view of the verifier.

4.2 AT with Public and Private Metadata

In Figure 6, we present an extension of the PMBTokens [41, Figure 8] with public metadata. This protocol is also designated verifier, requiring the secret key for verification.

Parameters Let λ be the security parameter, let p be a prime and let E be an elliptic curve group of order p with generators G_0, G_1. Let $\texttt{H}_t : \{0,1\}^* \to E$, $\texttt{H}_m : \{0,1\}^* \to \mathbb{Z}_p^*$ and $\texttt{H}_s : \{0,1\}^* \to \mathbb{Z}_p^*$ be hash-functions, and assume that group elements and integers can be encoded uniquely as strings. Furthermore, let metadata md be an element of a public set of valid strings. Finally, let $\mathsf{sk} := (k_{0,0}, k_{0,1}, k_{1,0}, k_{1,1}) \leftarrow\!\$\, (\mathbb{Z}_p^*)^4$ (all $k_{i,j}$ being distinct) be the signing key, and let $\mathsf{pk} := \{K_{i,j}\} = \{[k_{i,j}]G_i\}$, for $i, j = 0, 1$, be the public key. This is implicit knowledge in the protocol description.

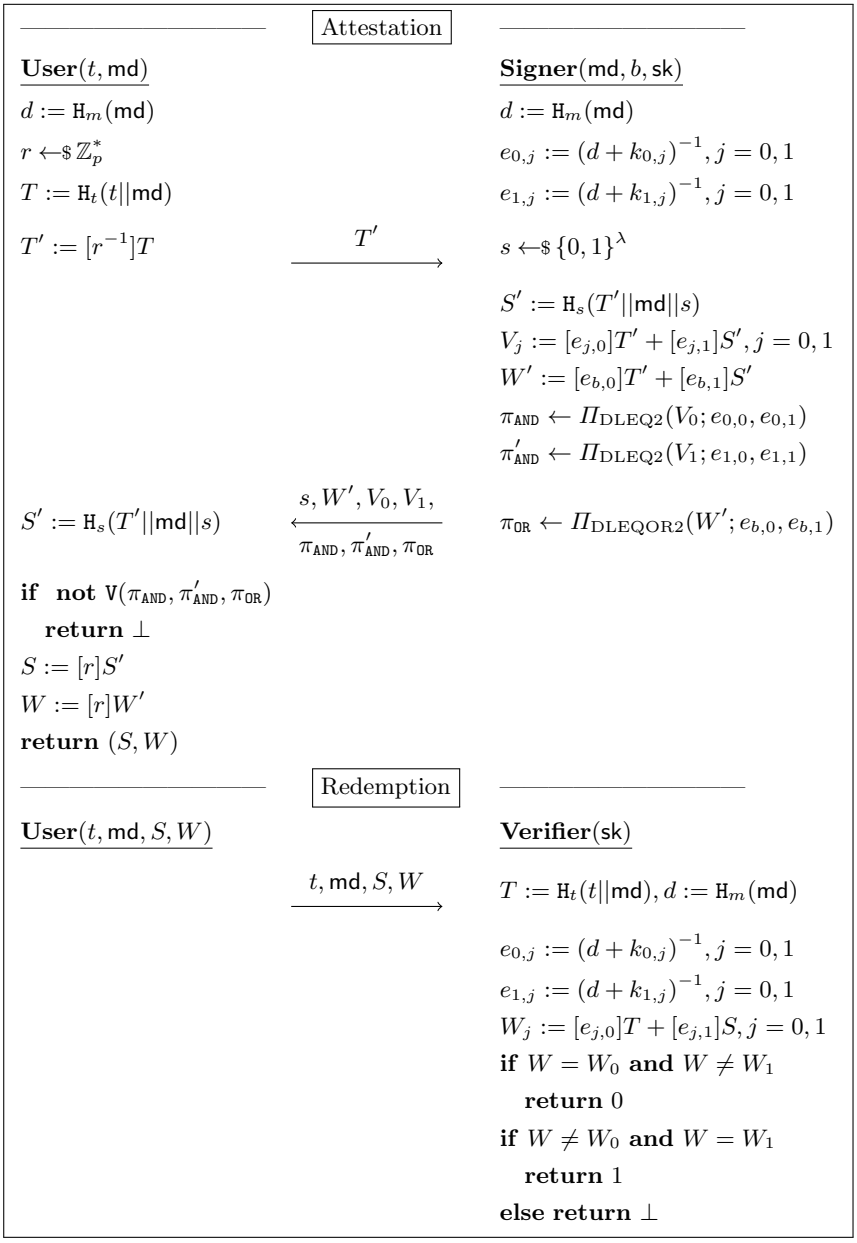

Fig. 6. Designated verifier anonymous tokens with public and private metadata, an adjusted extension of Kreuter et al. [41].

Zero-Knowledge Protocol `DLEQAND` The anonymous tokens protocol in Figure 6 uses the Π_{DLEQ2}-protocol defined in Section 2.3 twice as a subroutine to ensure that we afterwards can prove that the signed token W' was computed correctly. Given the generators G_0, G_1, T', S', the public keys $K_{i,j} := [k_{i,j}]G_i$ and the elements $V_i := [e_{i,0}]T' + [e_{i,1}]S'$, for $i, j = 0, 1$, we want to prove that the following relations hold:

$$\begin{bmatrix} G_0 + G_1 \\ V_i \end{bmatrix} = [e_{i,0}] \begin{bmatrix} [d]G_0 + K_{i,0} \\ T' \end{bmatrix} + [e_{i,1}] \begin{bmatrix} [d]G_1 + K_{i,1} \\ S' \end{bmatrix}.$$

We do this by instantiating $\Pi_{\text{DLEQ2}}(V_0; e_{0,0}, e_{0,1})$ and $\Pi_{\text{DLEQ2}}(V_1; e_{1,0}, e_{1,1})$ in Figure 6 to get proofs $\pi_{\texttt{AND}}$ and $\pi'_{\texttt{AND}}$. We denote the verification by $\texttt{V}(\pi_{\texttt{AND}}, \pi'_{\texttt{AND}})$.

Zero-Knowledge Protocol `DLEQOR` The anonymous tokens protocol in Figure 6 uses the Π_{DLEQOR2}-protocol defined in Section 2.4. The signer computes an `OR`-proof of equality of discrete logs by instantiating $\Pi_{\text{DLEQOR2}}(W'; e_{b,0}, e_{b,1})$. Consider the generators G_0, G_1, T', S', hashed metadata d and computed value W'. The signer then proves that W' is correctly computed, with respect to T' and S', and in the same way as one of the committed values V_0 or V_1, with respect to G and H. That is, for either $b = 0$ or $b = 1$:

$$V_b = [e_{b,0}]G_0 + [e_{b,1}]G_1 \quad \wedge \quad W' = [e_{b,0}]T' + [e_{b,1}]S'.$$

We denote the verification of the proof $\pi_{\texttt{OR}}$ by $\texttt{V}(\pi_{\texttt{OR}})$.

Correctness and Security

Theorem 4 (Completeness). *The anonymous token protocol with public and private metadata in Figure 6 is complete according to Definition 3 and will, according to Definition 4, return the correct metadata bit except with negligible probability.*

Proof. If the user submits (t, md, S, W), completeness follows from expanding W_b:

$$\begin{aligned} W_b &= [r]([e_{b,0}]T' + [e_{b,1}]S') = [r]([e_{b,0}][r^{-1}]T + [e_{b,1}]S') \\ &= [e_{b,0}]T + [r][e_{b,1}]S' = [e_{b,0}]\texttt{H}_t(t||\mathsf{md}) + [e_{b,1}]S. \end{aligned}$$

Furthermore, the probability that this equation holds for both $b = 0$ and $b = 1$ is negligible. If that was the case, then

$$[e_{0,0}]T + [e_{0,1}]S = [e_{1,0}]T + [e_{1,1}]S.$$

As we require all keys $k_{i,j}$ to be distinct, it follows that all $e_{i,j}$ are distinct. Then, we have that

$$T = \left[\frac{e_{1,1} - e_{0,1}}{e_{0,0} - e_{1,0}}\right] S.$$

Since $T = \texttt{H}_t(t||\mathsf{md})$ is sampled independently and uniformly at random, the probability that this equation holds is $1/p$, which is negligible.

Theorem 5 (Unforgeability). *The anonymous token protocol with public and private metadata in Figure 6 achieves one-more unforgeability with respect to Definition 5.*

Proof. For fixed metadata md we let the adversary query the signing oracle ℓ times for both $b = 0$ and $b = 1$. Then one-more unforgeability follows directly from [41, Theorem 8] and Lemma 1.

Theorem 6 (Unlinkability). *The anonymous token protocol with public and private metadata in Figure 6 achieves unlinkability with respect to Definition 6.*

Proof. We note that it is easy to create many different anonymity sets to distinguish users based on private metadata being $b = 0$ or $b = 1$, and in combination with different values of public metadata md. We restrict the unlinkability to hold for users within the same anonymity sets based on b and md, both sampled according to the real distribution of private and public metadata. Let $\mathfrak{U}_{b,\mathsf{md}}$ be this set, and select two sessions from $\mathfrak{U}_{b,\mathsf{md}}$. Then it follows directly from [41, Theorem 9] that the probability of success of the adversary will be upper bounded by $2/m + \mathsf{negl}(\lambda)$

Theorem 7 (Private metadata bit). *The anonymous token protocol with public and private metadata in Figure 6 provides private metadata bit with respect to Definition 7.*

Proof. This statement follows directly from the proof of [41, Theorem 10], which describes a hybrid argument to prove that instances with private bit 0 are indistinguishable from instances with private bit 1. Notice in particular that the extra $\mathtt{OR}$-proofs in our protocol are independent of the private bit b, and therefore need no additional simulation.

4.3 Public Verifiability from Pairings

The authors of Privacy Pass [27] described an application where the issuer and the recipient of a token would be the same entity, possibly separated by time. For the application we present in Section 6, those two roles are in fact separate, and one should therefore have a scheme that supports public verifiability. It remains an open problem to achieve this without pairings, unless we allow for two rounds of communication [2, 50].

We move on to provide a new variant of a little known partially blinded signature by Zhang, Safavi-Naini and Susilo [51]. The protocol allows a user and a signer to generate a signature on a user-private message m and agreed-upon metadata md. Both the issuance protocol and the signature consists of a single curve point. The authors prove security against one-more forgery, assuming that the CTCDH problem is hard.

We show that the idea underlying this scheme can be viewed as a combination of Boneh-Lynn-Shacham signatures [10] and Privacy Pass, inheriting its attractive properties from both. We update the ZSS protocol to use different source

groups $\mathbb{G}_1 \neq \mathbb{G}_2$, where there are no efficiently computable homomorphism, i.e., a type III pairing (cf. Section 2.1). This allows more efficient concrete instantiations. This is secure in our setting, but would not be in the case of type I or II pairings.

Parameters Let λ be the security parameter, let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a pairing, where G_1, G_2 and g_T are generators for their respective prime p order groups. Furthermore, let $\mathtt{H}_1 : \{0,1\}^* \to \mathbb{G}_1$ and $\mathtt{H}_m : \{0,1\}^* \to \mathbb{Z}_p^*$ be hash functions, and assume that group elements and integers can be encoded uniquely as strings. Also, let md be an element of a public set of valid metadata strings. Finally, let $\mathsf{sk} := k \leftarrow\!\$\, \mathbb{Z}_p^*$ be the signing key, and let $\mathsf{pk} := K = [k]G_2$ be the public key. This is implicit knowledge in the protocol description.

Protocol Layout Recall that the BLS-scheme signs a message m by hashing it to the group generated by G_1 and multiplying it with the secret key k; $W := [k]\mathtt{H}_1(m)$. The signature can then be verified by checking that

$$\hat{e}(\mathtt{H}_1(m), K) = \hat{e}(W, G_2).$$

Correctness follows from the linearity of the pairing.

We replace m by a token seed t, and use the same trick as earlier to concurrently update the key-pair based on metadata. Then we get the following anonymous token scheme:

Signing The user sends $T' := [r^{-1}]\mathtt{H}_1(t)$ to the issuer, who returns $W' := [e]T'$, for $e = (d{+}k)^{-1}$. The user can verify that the signature is correct by checking $\hat{e}(W', U) = \hat{e}(T', G_2)$, for $U := [d{+}k]G$, and then storing $(t, \mathsf{md}, W = [r]W')$.

Verification The user sends (t, md, W), and the recipient can verify the token by checking if $\hat{e}(W, U) = \hat{e}(T, G_2)$.

This scheme hides the token similarly to Privacy Pass, it can be verified without using the private key, and its unforgeability comes directly from BLS. We note that the check $\hat{e}(W', U) = \hat{e}(T', G_2)$ ensures that the tokens are signed correctly with respect to the public key. The complete protocol is listed in Figure 7. Finally, we note that we can batch-verify n tokens under the same key and metadata and check for equality in the following way:

$$\hat{e}\left(\sum_i W_i, U\right) = \hat{e}\left(\sum_i T_i, G_2\right).$$

This saves the verifier of $2(n-1)$ expensive pairing-computations, which is especially useful in systems with large same anonymity sets.

Correctness and Security

Theorem 8 (Completeness). *The anonymous token protocol with public metadata and public verifiability in Figure 7 is complete according to Definition 3.*

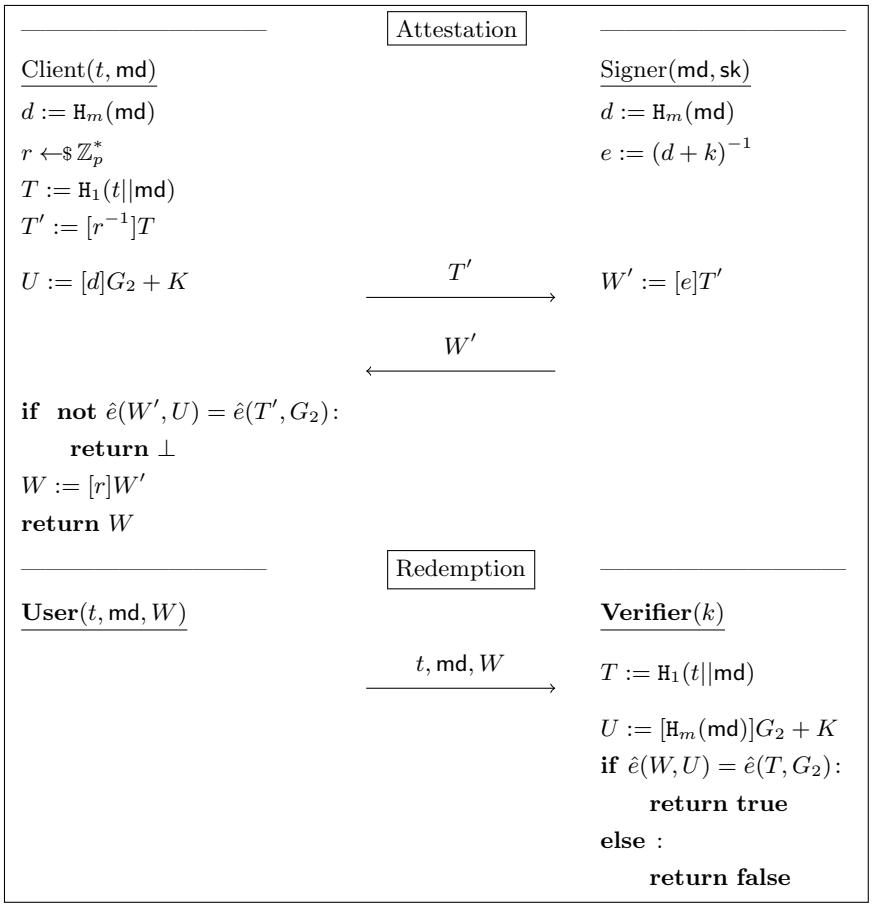

Fig. 7. Anonymous tokens with public metadata and public verifiability by adjusting Zhang et al. [51] for type III pairings.

Proof. Completeness follows from expanding $\hat{e}(W, U)$:

$$\begin{aligned}\hat{e}(W, U) &= \hat{e}([r]W', [d + k]G_2) = \hat{e}([r][e]T', [d + k]G_2)\\ &= \hat{e}([r][e][r^{-1}]T, [d + k]G_2) = \hat{e}([e]T, [d + k]G_2)\\ &= \hat{e}(T, G_2)^{e\cdot(d+k)} = \hat{e}(T, G_2).\end{aligned}$$

Theorem 9 (Unforgeability). *The anonymous token protocol with public metadata and public verifiability in Figure 7 achieve one-more unforgeability with respect to Definition 5.*

Proof. Unforgeability of the anonymous token protocol follows directly from the unforgeability of the BLS-signature scheme [10] and Lemma 1. The constructions are identical, except for the fact that we sign with $e := (d + k)^{-1}$ instead of k

to embed the public metadata into the token. The relation $\log_G[d]G + K = (d + k) = \log_{W'} T'$ is immediate from the client's verification equation.

Unlinkability holds only because we use a type III pairing, otherwise it would be possible to check if $\hat{e}(W, \phi(T')) = \hat{e}(W', \phi(T))$. Recall that we fix the metadata md for this notion.

Theorem 10 (Unlinkability). *The anonymous token protocol with public metadata and public verifiability in Figure 7 achieve unlinkability with respect to Definition 6.*

Proof. Assume that no efficiently computable $\phi : \mathbb{G}_2 \to \mathbb{G}_1$ exists. Observe that given any valid token (t, md, W) and any view (T', W') there exists a value r' such that $W - [r']W'$ and $T - [r']T'$, and hence, T is independent of any W. It follows that the anonymous token is unlinkable.

5 Performance and Comparison

In this section, we briefly describe the most efficient anonymous token protocols with public metadata in the literature, for example to enable batched revocation. We compare the protocols with our schemes in Table 1. To streamline the comparison, we assume that all parties know the public metadata, for example that md is the current date, and assume that this implicit knowledge is not sent. We instantiate the schemes with $\lambda = 128$ bits of security. Finally, we present a concrete example in Section 5.6 to show that we can replace PrivateStats with our protocol in Figure 5 to improve both communication size and computational efficiency.

5.1 Privacy Pass

Our protocol in Figure 5 is inspired by Privacy Pass [27], and they have identical structure and communication. The main difference is the change of private key used for signing, and the updated zero-knowledge proof with respect to the new public key, both depending on the public metadata. The zero-knowledge proofs are of the same size, and it follows that the communication sizes are equal. However, Privacy Pass does not allow public metadata unless we have one public key for each valid string of metadata, and hence, to allow for 2^N possible messages md, Privacy Pass must publish 2^N public keys. A batch of n signatures under the same public key includes n group elements in the request, and n group elements and only one proof in the signature since the zero-knowledge proof can be batched for all instances.

5.2 PrivateStats

PrivateStats [36] is also inspired by Privacy Pass [27], but uses a attribute-based VOPRF to generate new public keys on the fly. To allow for 2^N strings of public

metadata, there are two main differences: 1) the public key consists of $N + 2$ group elements, and 2) the token consists of an additional N group elements and zero-knowledge proofs to ensure that the correct public key is used in the signature. We note that a batch of n signatures under the same public key includes n group elements in the request, and n group elements and only one proof in the signature since the final zero-knowledge proof can be batched for all instances.

5.3 Tokens from RSA

Abe and Fujisaki [1] presents a partially blind signature scheme based on RSA. The public exponent e must be at least two bits longer than the public metadata, and we fix this to be of length 130 bits. The user updates the public key to $e_{\mathsf{md}} = e \cdot \tau(\mathsf{md})$, for a public formatting function τ, when they blind the message, and the signer updates the secret key $d_{\mathsf{md}} = (e \cdot \tau(\mathsf{md}))^{-1} \mod N$ when signing. Otherwise, the partially blind signature scheme [1] is similar to the blind signature by Chaum [20]. Each message consists of one element in $\mathbb{Z}_N^*$, and we note that a batch of n signatures under the same public key expands the messages by a factor n.

5.4 Tokens with Private Metadata

Kreuter et al. [41] presents an extension of Privacy Pass [27] to include private metadata. They publish two public keys, and the signer proves in zero-knowledge that the token is signed with one of the corresponding private keys. To ensure metadata privacy, each token is randomized based on a fresh seed s that is given to the user, and hence, the signature consists of a seed, a group element and a proof. The token consists of the initial seed t in addition to two group elements. Like Privacy Pass, this protocol must publish a new pair of public keys for each valid string of metadata, and can sign a batch of n tokens using a batched zero-knowledge proof to ensure correctness.

5.5 Comparison

We present a comparison of schemes in Table 1, where we focus on communication complexity. We note that both RSA and pairing based cryptography are usually slower than elliptic curve cryptography, in addition to requiring larger parameters. We also note that the updated keys in our protocols are only dependent on the secret key and the metadata, and can often be pre-computed. We observe that when allowing for batched token-revocation, our protocols are more efficient than the state of the art in all categories.

While RSA and elliptic curve cryptography are primitives implemented in all mainstream cryptographic libraries, there are few trustworthy implementations of pairings. Even though there exists a few implementations[7], they are mostly for academic use, maybe except for the implementation in Rust used by Zcash[8].

[7] Pairings: `https://hackmd.io/@zkteam/eccbench`
[8] Zcash: `https://github.com/zkcrypto/bls12_381`

Public Metadata (PM)	PubKey	Request	Signature	Token
Privacy Pass [27]	$257 \cdot 2^N$	257	769	385
PrivateStats [36]	$257 \cdot (N+2)$	257	$769 \cdot (N+1)$	385
Our scheme (Fig 5)	257	257	769	385
PM + Private Metadata	**PubKey**	**Request**	**Signature**	**Token**
Kreuter et al. [41]	$514 \cdot 2^N$	257	1921	642
Our Scheme (Fig 6)	1028	257	3203	642
PM + Public Verifiability	**PubKey**	**Request**	**Signature**	**Token**
Abe and Fujisaki [1]	3202	3072	3072	3200
Our scheme (Fig 7)	763	382	382	510

Table 1. All numbers are given in bits. We compare the schemes for 128 bits of security, allowing for 2^N strings md of metadata (or 2^N batches of token-revocations). Token seeds t is of size 128 bits, and metadata md is implicit knowledge. Privacy Pass, PrivateStats, Kreuter et al. and our protocols in Fig 5 and Fig 6 are instantiated with curve x25519 [7], Abe and Fujisaki is instantiated with RSA-3072 and our protocol in Fig 7 is instantiated with BLS12-381 [46]. Privacy Pass, PrivateStats, Kreuter et al., Fig 5 and Fig 6 allows for batched ZK-proofs for additional tokens.

5.6 Telemetry Collection in WhatsApp

PrivateStats [36] was designed to allow users of WhatsApp to anonymously report telemetry data to Facebook. We present a concrete comparison to our protocols in Table 2. Here, we assume that Facebook wants to update their public keys only once a year, rotate signing keys every day, and only sign one token per user each day. We let the public metadata be dates encoded as strings"YYYY-MM-DD", where the year is fixed.

Privacy Pass [27] is very efficient in terms of communication, but requires one public key per day. Hence, the public key is of size 93805 bits over a year of 365 days, that is, almost 12 KB. An alternative method to download all keys and store them until usage is to use a Merkle-tree for key-transparency and give paths corresponding to the current public key as a part of each signature. Then, the public key consists of the root of size 256 bits, while each signature consists of $\lceil \log_2(365) \rceil = 9$ hashes of 256 bits in addition to the public key, the token and the zero-knowledge proof. We give both instantiations in the table, and denote the alternative as Privacy Pass+.

Our scheme in Figure 5 has the smallest overall communication complexity of all schemes. It offers much smaller keys than Privacy Pass, and much smaller signatures than Privacy Pass+ and PrivateStats, saving up to 90 % in communication. If all 2 billion users of WhatsApp report their telemetry every day, our scheme in Figure 5 would save more than 1.7 TB of communication for the Facebook servers compared to the current implementation of PrivateStats.

Protocol	PubKey	Request	Signature	Token
Privacy Pass [27]	93805	257	769	385
Privacy Pass+	256	257	3330	385
PrivateStats [36]	2313	257	7690	385
Our scheme (Fig 5)	257	257	769	385
Our scheme (Fig 7)	763	382	382	510

Table 2. All numbers are given in bits. We compare Privacy Pass, PrivateStats, and the protocols in Fig 5 and Fig 7 with key-rotation each day in a year, only signing one token at a time.

Our scheme in Figure 7 offers similar improvements to communication, in addition to public verifiability, but at the cost of less standardized cryptography and less efficient computation.

6 Application to Contact Tracing

As nations started adopting digital contact tracing during the COVID-19 pandemic, privacy experts warned that such systems could enable the collection of people's contact graphs. The dp^{3}t protocol [48] was eventually adopted as the *de facto* method for digital contact tracing through its implementation and deployment in iOS and Android as the Exposure Notification System (ENS).

We provide a brief overview of the basic dp^{3}t idea in order to put our contribution into context. The protocol is instantiated on each participating phone, which generates a random key (Technical Exposure Key, TEK) every day. The TEK is used to generate new Rotating Proximity Identifiers (RPI) every 10–20 minutes, which is then broadcast from the phone using Bluetooth Low Energy (BLE). Other phones in the proximity store any RPI they hear, along with the signal strength.

If Alice tests positive for COVID-19 she can upload her TEKs (now renamed to diagnosis keys, DK) along with her BLE transmission strength to a health authority bulletin board. Bob's phone regularly checks the board to see if there is a sufficiently large overlap between published the DKs and the RPIs stored locally, and with sufficiently low difference between transmission strength and received strength. If this is the case, then Bob is given a suitable alert to let him know that he most likely has been in close vicinity of an infected individual, and should follow any advice given by the health authorities.

The process of uploading TEKs should depend on some sort of authorization. The dp^{3}t documentation describes a simplified model where a doctor receives the test results, and sends the patient an SMS with a short upload code. Now, this process may take precious person-hours during a pandemic. Some countries have therefore opted to connect their exposure notification with already existing centralized registries of positive test results, e.g., Norway, Denmark and Estonia.

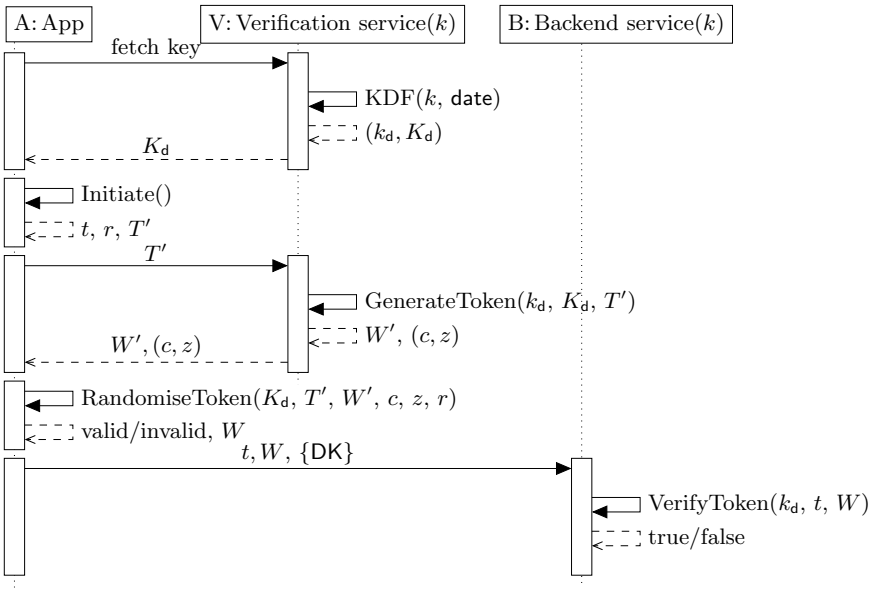

Fig. 8. A sequence diagram of anonymous tokens in Smittestopp.

When starting the upload process, the user is prompted to log in to some government service ("verification"). Once the user has identified herself, the service makes a query to the relevant health registry. The service returns an access token to the app if there exists a recent positive test, which is then used to upload the keys to "backend". Unfortunately, this token may create an identifiable link from the meant-to-be-anonymous database of DKs, and unique identities in the health registry. Using anonymous credentials, one can break this link (up to traffic analysis).

The Norwegian Institute of Public Health (NIPH) wanted the tokens to be timestamped in order to avoid users posting severely delayed keys: This would have allowed an attacker to get well again, move back out among other people, and only then upload to the backend service. Notice that merely tying the token to keys – e.g., by using a hash of the TEKs as the token seed t – would not avoid this attack, as those could have been generated and stored until the time of the attack. As a result of this, it was decided that the keys should be rotated regularly.

Privacy Pass was implemented as a reusable C# package, to ease the integration into the contact tracing app. The verification and backend services keep a master secret key k, and generate daily keys from some $\mathsf{KDF}(k, \mathsf{date})$. The public key is posted from the verification service. The full integration of anonymous tokens is described in Figure 8.

We finally note that this key distribution method suffers from a potential attack by a dishonest verification service that could serve special public keys to

track individuals. It is, however, detectable by the users if they share their view of the public keys with each other to ensure consistency. The current solution was accepted by all involved stakeholders due to limited time and a weighting of the practical risk against the potential reward. However, the challenges with respect to key-rotation and key-sharing strongly motivated the authors' work in Section 4.

7 Conclusion

In this work, we have updated the definitions for anonymous tokens to also include public metadata, and we have constructed three protocols that satisfy these definitions. Additionally, we combine public metadata with either private metadata or public verifiability, and show that all instantiations are efficient in practice. For situations with frequent key-rotation, we show that our protocols can save up to 90 % in communication over the state of the art. Furthermore, our protocols fits nicely into the Privacy Pass framework, which makes it easy to incorporate our contributions in the ongoing standardization processes by IETF and W3C, answering an open problem.

We also provide a description of how anonymous tokens can be used to improve the user's privacy in contact tracing applications, and implemented this into the solution used in Norway. The app has $870,000$ users at the time of writing[9]. As the Norwegian app is built on top of the same code base as the Danish app, we consider it to be easy to extend the adaption of anonymous tokens to their app, and most likely others as well.

We would also like to suggest new use-cases for anonymous tokens. For example, anonymous tokens can improve the privacy of users traveling with public transport. Bus or train companies may require patrons to verify their period tickets for each journey, perhaps primarily to analyze traffic data. However, this can easily reveal the routes of single users while traveling in-between their home and work place, but also to the abortion clinic, their church or to a public demonstration etc. If all travelers with valid tickets are given a series of tokens (e.g. with public metadata being the date or week or month the ticket is valid), then these can be redeemed when boarding. This way, the companies get the statistics they are interested in, without invading the user's privacy. In general, any systems with leveled authenticated login but anonymous actions can make use of our protocols, e.g., systems with electronic locks that only care if the user has certain privileges or not.

Finally, we would like to see improvements in three directions. Firstly, the zero-knowledge proofs used by the anonymous tokens protocol with public and private metadata in Figure 6 are much larger than the ones by Kreuter et al. [41], in contrast to our protocol with public metadata in Figure 5 achieving the exact same communication cost as Privacy Pass [27]. In particular, we would like to reduce the number of proofs and extra group elements in the

[9] Smittestopp: `https://www.fhi.no/om/smittestopp/nokkeltall-fra-smittestopp`, last accessed 2021-02-24

protocol of Section 4.2. Secondly, we would like to provide protocols free of zero-knowledge proofs, to reduce the communication and computational cost, as provided in [41, Section 7]. Finally, we would like to extend our protocols to achieve post-quantum security, continuing the work by Albrecht et al. [3] on lattices and the work by Boneh et al. [9] on isogenies.

Acknowledgments The authors are very grateful to Henrik Walker Moe (Bekk) for stellar collaboration during the C# implementation phase. The final integration into Smittestopp was primarily a collaboration between Henrik, Johannes Brodwall (Sopra Steria) and Sindre Møgster Braaten (NIPH), with the authors and others as close consultants.

References

1. Masayuki Abe and Eiichiro Fujisaki. How to date blind signatures. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 244–251. Springer, Heidelberg, November 1996.
2. Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 271–286. Springer, Heidelberg, August 2000.
3. Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. Round-optimal verifiable oblivious pseudorandom functions from ideal lattices. Cryptology ePrint Archive, Report 2019/1271, 2019. `https://eprint.iacr.org/2019/1271`.
4. Foteini Baldimtsi and Anna Lysyanskaya. Anonymous credentials light. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 1087–1098. ACM Press, November 2013.
5. Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 257–267. Springer, Heidelberg, September 2003.
6. Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In Paul F. Syverson, editor, *FC 2001*, volume 2339 of *LNCS*, pages 319–338. Springer, Heidelberg, February 2002.
7. Daniel J. Bernstein. Curve25519: high-speed elliptic curve cryptography, 2005. https://cr.yp.to/ecdh.html.
8. Olivier Blazy, David Pointcheval, and Damien Vergnaud. Compact round-optimal partially-blind signatures. In Ivan Visconti and Roberto De Prisco, editors, *SCN 12*, volume 7485 of *LNCS*, pages 95–112. Springer, Heidelberg, September 2012.
9. Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 520–550. Springer, Heidelberg, December 2020.
10. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, December 2001.
11. Jonathan Burns, Daniel Moore, Katrina Ray, Ryan Speers, and Brian Vohaska. EC-OPRF: Oblivious pseudorandom functions using elliptic curves. Cryptology ePrint Archive, Report 2017/111, 2017. `http://eprint.iacr.org/2017/111`.

12. Jan Camenisch and Thomas Groß. Efficient attributes for anonymous credentials. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 345–356. ACM Press, October 2008.
13. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321. Springer, Heidelberg, May 2005.
14. Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In Stanislaw Jarecki and Gene Tsudik, editors, *PKC 2009*, volume 5443 of *LNCS*, pages 481–500. Springer, Heidelberg, March 2009.
15. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001.
16. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 268–289. Springer, Heidelberg, September 2003.
17. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, Heidelberg, August 2004.
18. Melissa Chase, Sarah Meiklejohn, and Greg Zaverucha. Algebraic MACs and keyed-verification anonymous credentials. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 2014*, pages 1205–1216. ACM Press, November 2014.
19. Melissa Chase, Trevor Perrin, and Greg Zaverucha. The signal private group system and anonymous credentials supporting efficient verifiable encryption. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 20*, pages 1445–1459. ACM Press, November 2020.
20. David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982.
21. David Chaum. Blind signature system. In David Chaum, editor, *CRYPTO'83*, page 153. Plenum Press, New York, USA, 1983.
22. David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 89–105. Springer, Heidelberg, August 1993.
23. Xiaofeng Chen, Fangguo Zhang, Yi Mu, and Willy Susilo. Efficient provably secure restrictive partially blind signatures from bilinear pairings. In Giovanni Di Crescenzo and Avi Rubin, editors, *FC 2006*, volume 4107 of *LNCS*, pages 251–265. Springer, Heidelberg, February / March 2006.
24. Sherman S. M. Chow, Lucas Chi Kwong Hui, Siu-Ming Yiu, and K. P. Chow. Two improved partially blind signature schemes from bilinear pairings. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 05*, volume 3574 of *LNCS*, pages 316–328. Springer, Heidelberg, July 2005.
25. Alex Davidson. Supporting the latest version of the privacy pass protocol. `https://blog.cloudflare.com/supporting-the-latest-version-of-the-privacy-pass-protocol`. (Accessed 24-February-2021).
26. Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: A privacy-enhancing protocol and browser extension. `https://privacypass.github.io`. (Accessed 24-February-2021).

27. Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, July 2018.
28. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.
29. Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword search and oblivious pseudorandom functions. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 303–324. Springer, Heidelberg, February 2005.
30. Georg Fuchsbauer, Christian Hanser, Chethan Kamath, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model from weaker assumptions. In Vassilis Zikas and Roberto De Prisco, editors, *SCN 16*, volume 9841 of *LNCS*, pages 391–408. Springer, Heidelberg, August / September 2016.
31. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 233–253. Springer, Heidelberg, August 2015.
32. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008. Applications of Algebra to Cryptography.
33. Jorge Guajardo, Bart Mennink, and Berry Schoenmakers. Anonymous credential schemes with encrypted attributes. In Swee-Huay Heng, Rebecca N. Wright, and Bok-Min Goi, editors, *CANS 10*, volume 6467 of *LNCS*, pages 314–333. Springer, Heidelberg, December 2010.
34. Ryan Henry. *Efficient Zero-Knowledge Proofs and Applications*. PhD thesis, University of Waterloo, 2014.
35. Ryan Henry and Ian Goldberg. Batch proofs of partial knowledge. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *ACNS 13*, volume 7954 of *LNCS*, pages 502–517. Springer, Heidelberg, June 2013.
36. Sharon Huang, Subodh Iyengar, Sundar Jeyaraman, Shiv Kushwah, Chen-Kuei Lee, Zutian Luo, Payman Mohassel, Ananth Raghunathan, Shaahid Shaikh, Yen-Chieh Sung, and Albert Zhang. Privatestats: De-identified authenticated logging at scale. Technical report, Facebook Inc., `https://research.fb.com/wp-content/uploads/2021/01/PrivateStats-De-Identified-Authenticated-Logging-at-Scale_final.pdf`, jan 2021.
37. Internet Engineering Task Force. Privacy pass datatracker. `https://datatracker.ietf.org/wg/privacypass`. (Accessed 24-February-2021).
38. Subodh Iyengar and Erik Taubeneck. Fraud resistant, privacy preserving reporting using blind signatures. `https://github.com/siyengar/private-fraud-prevention`. (Accessed 24-February-2021).
39. Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 233–253. Springer, Heidelberg, December 2014.
40. Stanislaw Jarecki, Hugo Krawczyk, and Jiayu Xu. OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 456–486. Springer, Heidelberg, April / May 2018.

41. Ben Kreuter, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. Anonymous tokens with private metadata bit. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part I*, volume 12170 of *LNCS*, pages 308–336. Springer, Heidelberg, August 2020.
42. Ben Kreuter, Tancrede Lepoint, Michele Orru, and Mariana Raykova. Efficient anonymous tokens with private metadata bit. Cryptology ePrint Archive, Report 2020/072, 2020. `https://eprint.iacr.org/2020/072`.
43. Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 31–53. Springer, Heidelberg, August 1993.
44. Dimitrios Papadopoulos, Duane Wessels, Shumon Huque, Moni Naor, Jan Včelák, Leonid Reyzin, and Sharon Goldberg. Making NSEC5 practical for DNSSEC. Cryptology ePrint Archive, Report 2017/099, 2017. `http://eprint.iacr.org/2017/099`.
45. David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 252–265. Springer, Heidelberg, November 1996.
46. S. Yonezawa, S. Chikara, T. Kobayashi and T. Saito. Pairing-Friendly Curves. `https://tools.ietf.org/id/draft-yonezawa-pairing-friendly-curves-00.html`. (Accessed 24-February-2021).
47. Kurt Thomas, Jennifer Pullman, Kevin Yeo, Ananth Raghunathan, Patrick Gage Kelley, Luca Invernizzi, Borbala Benko, Tadek Pietraszek, Sarvar Patel, Dan Boneh, and Elie Bursztein. Protecting accounts from credential stuffing with password breach alerting. In Nadia Heninger and Patrick Traynor, editors, *USENIX Security 2019*, pages 1556–1571. USENIX Association, August 2019.
48. Carmela Troncoso et al. Decentralized privacy-preserving proximity tracing. `https://arxiv.org/abs/2005.12273`, 2020.
49. World Wide Web Consortium. Trust Token API Explainer. `https://github.com/WICG/trust-token-api`. (Accessed 24-February-2021).
50. Qianhong Wu, Willy Susilo, Yi Mu, and Fanguo Zhang. Efficient partially blind signatures with provable security. In Marina Gavrilova, Osvaldo Gervasi, Vipin Kumar, C. J. Kenneth Tan, David Taniar, Antonio Laganá, Youngsong Mun, and Hyunseung Choo, editors, *Computational Science and Its Applications - ICCSA 2006*, pages 345–354, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
51. Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. Efficient verifiably encrypted signature and partially blind signature from bilinear pairings. In Thomas Johansson and Subhamoy Maitra, editors, *INDOCRYPT 2003*, volume 2904 of *LNCS*, pages 191–204. Springer, Heidelberg, December 2003.
52. Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. An efficient signature scheme from bilinear pairings and its applications. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *PKC 2004*, volume 2947 of *LNCS*, pages 277–290. Springer, Heidelberg, March 2004.