Tibor Jager and Andy Rupp*

Black-Box Accumulation: Collecting Incentives in a Privacy-Preserving Way

Abstract: In this paper, we formalize and construct black-box accumulators (BBAs), a useful building block for numerous important user-centric protocols including loyalty systems, refund systems, and incentive systems (as, e.g., employed in participatory sensing and vehicle-to-grid scenarios). A core requirement all these systems share is a mechanism to let users collect and sum up values (call it incentives, bonus points, reputation points, etc.) issued by some other parties in a privacy-preserving way such that curious operators may not be able to link the different transactions of a user. At the same time, a group of malicious users may not be able to cheat the system by pretending to have collected a higher amount than what was actually issued to them.

As a first contribution, we fully formalize the core functionality and properties of this important building block. Furthermore, we present a generic and non-interactive construction of a BBA system based on homomorphic commitments, digital signatures, and non-interactive zero-knowledge proofs of knowledge. For our construction, we formally prove security and privacy properties. Finally, we propose a concrete instantiation of our construction using Groth-Sahai commitments and proofs as well as the optimal structure-preserving signature scheme of Abe et al. and analyze its efficiency.

Keywords: Loyalty systems, incentive collection, refund systems, participatory sensing, vehicle-to-grid, provable security.

1 Introduction

Loyalty programs are an important market mechanism for customer retention. These programs enjoy a great popularity all over the world and are employed from small shops to retail chains and even across different stores. Examples are the Payback program in Germany [3], with more than 60% of German households enrolled, the Coop Supercard program in Switzerland [4], or the Nectar program in the UK [2]. The basic principle underlying such programs, is that customers collect loyalty points for every purchase in a participating shop. They may redeem them later to directly pay for a purchase, get a voucher, etc. User devices range from regular PCs for online shopping to magnetic cards or smartcards to NFC-enabled smartphones for offline shopping.

We can find similarly working incentive mechanisms in other areas as well. For instance, in upcoming user-centric cyber-physical systems (CPS) such as participatory sensing (PPS) systems [13] or the vehicle-to-grid (V2G) [23], users will receive incentives (e.g., micropayments) to actively participate and contribute to the system. PPS refers to a whole family of human-centered CPS. Users, called Producers, collect fine-grained measurement data (enhanced by location and time) in exchange for incentive points, such as temperature, CO2, noise level, health data, etc. by means of mobile sensors which is processed and provided in some form to Consumers. The sensing and data transmission might be performed by ubiquitous smart phones (possibly connected to additional external sensor devices) or by dedicated constrained devices (such as, e.g., [15]). V2G is part of the overall Smart Grid vision. Here batteries of electric cars serve as projectable and controllable Smart Grid components which are used as energy source if there is a peak in demand and as buffer if there is a surplus of energy in the Smart Grid. E-Car owners will receive incentives for providing battery power to the Smart Grid.

In all mentioned applications, the anonymity and location privacy of users is a critical issue. Users should stay anonymous while collecting points, where ideally, two such transactions should not even be linkable. Otherwise, operators may trace the location and behavior of users quite well, e.g., where and when they bought which items (loyalty systems), where and when they left their cars (V2G), and even worse, where they walked around in between (PPS). Unfortunately, privacy is often neglected in these systems. For instance, Loyalty Partner, the operator of the Payback program, received the Ger-

Tibor Jager: Ruhr-University Bochum, E-mail: tibor.jager@rub.de
***Corresponding Author: Andy Rupp:** Karlsruhe Institute of Technology, E-mail: andy.rupp@kit.edu

man Big Brother Award for violating individual privacy [1].

Our Contribution

In this paper, we focus on properly formalizing the *core* functionality and security and privacy properties of a point collection mechanism which can be used, as a building block, to implement the applications mentioned above. Except from [26], no prior work seems to have separately studied this important multi-purpose building block. To the best of our knowledge our work is the first proposing a *formal* framework and model.

To do this end, we decided to restrict ourselves to *non-interactive* systems. Considering those systems only typically leads to simpler, more compact and comprehensible security models as well as less complex and error-prone security proofs (e.g., since concurrent executions of protocols are not an issue). Furthermore, non-interactive systems are preferred in many application scenarios due to lower communication latencies. We thereby assume application scenarios where points are always positive values and there is no need to withdraw points after they have been issued. Furthermore, we assume, for simplicity, that all points can be issued using a single secret key (e.g., since all shops issuing points have an online connection to the loyalty system operator). We leave extending our framework to remove those constraints as future work.

Our framework is multi-use token based: In the beginning, a user receives an accumulation token bound to a unique serial number known to both parties, the user and the issuer of the token. All points a user obtains can be collected using this *single* accumulation token. To do this in an unlinkable fashion, the user blinds and (possibly) unblinds this token before and after every point collection transaction. When redeeming the token, the sum of all collected points as well as the serial number is revealed, which allows to check for double-spending. Our security definition is inspired by the indistinguishability-based security definition for a pre-payments with refunds scheme proposed in [27]. It ensures that an adversary (possibly using multiple accumulation tokens) may not be able to redeem more points than legitimately issued to him. Our privacy definition is simulation-based and guarantees that besides the sum of all points no additional information is revealed to link transactions.

Besides a formal framework, we propose a *generic* non-interactive construction from abstract cryptographic building blocks such as homomorphic commitments, signatures, and non-interactive proofs of knowledge. The scheme offers a constant-size public key owned by the operator as well as a constant-size accumulation token owned by each user. The latter only consists of two commitments, the corresponding openings, and a signature. Blinding the token before collecting points adds an extra proof-of-knowledge to the token. As opposed to the state-of-the-art in loyalty, incentive, and refund mechanisms, we provide a generic construction of such a mechanism as well as full proof of security and privacy.

Last but not least, we exemplarily show how our generic BBA scheme can be efficiently instantiated based on Groth-Sahai proofs under the SXDH assumption [21] and the structure-preserving signature scheme of Abe et al. [6] which has been proven secure in the generic group model [28]. For security and privacy, we additionally rely on the assumption that the common reference string (CRS) used by our scheme has been generated honestly. Note that the CRS could be computed distributedly using multiparty computation techniques if one does not trust a single entity to setup the CRS properly. Using this instantiation, collecting points in an unlinkable fashion requires 70 exponentiations and 7 pairing computations on the user's device. This translates to an estimated runtime (ignoring communication costs) of about 445 ms on a low-end smartphone for a rather high security of 128 bit. Redeeming the points requires 66 exponentiations or about 324 ms on such a smartphone. A token in this setting consists of 66 group elements, which translates to about 24 kb that have to be transmitted.[1]

Related Work

BBA bear some resemblance with the notion of (additively) homomorphic signature schemes, formally introduced in [22]. These schemes allow to efficiently combine two given signatures $\mathsf{Sign}(m_1)$ and $\mathsf{Sign}(m_2)$ on messages m_1 and m_2 to a new signature $\mathsf{Sign}(m_1 + m_2)$ for the message $m_1 + m_2$. This property is clearly useful to realize accumulation of incentives. However, to achieve security and privacy in a BBA scheme, one would need additional properties: (1) a user must not be able to combine given signatures in an arbitrary fashion, but he must only be able to combine signatures issued to

1 A typical communication protocol for our applications is Near Field Communication (NFC), which has a data rate of up to 424 kB/s.

him and each such signature may only occur once and in a single aggregate. However, homomorphic signatures typically allow everyone to compute arbitrary (linear) combinations of given signatures. (2) signatures must be issued in a blind fashion. To the best of our knowledge, homomorphic signature schemes with these additional properties have not been proposed yet.

Apart from the related cryptographic primitives mentioned above, mechanisms related to the notion of BBAs have also been proposed as part of refund, loyalty, and incentive systems. One may distinguish between multi-use and single-use token systems. The former type allows to accumulate points from different collection transactions and represent these points using a single token, whereas in the latter type each collection transaction leads to one or more new tokens. Multi-use token systems typically have several important advantages compared to the other system type: more compact representations of points, no need to repeatedly execute the point collection and redemption protocols in order to issue or cash-in multiple points, less information leakage about the individual collection transactions during redemption as all points are summed up[2], etc. As we propose a multi-use token system, we will first take a closer look at prior work falling into this category before we briefly touch single-use token systems.

Multi-use token systems. In [27] Rupp et al. propose a refund collection mechanism based on BLS signatures [10] as part of an interactive pre-payments with refunds scheme for public transport. Security and privacy properties are only formalized for the overall scheme, not individually for the refund scheme. Collecting and redeeming a refund is very efficient: essentially only a single exponentiation is required on the user's side to execute these operations. Also a token is pretty compact as it only consists of two group elements and two exponents. However, the refund scheme suffers from a (potentially) large public key. The size of this key grows linearly with the number of possible refund amounts in the system. Furthermore, the security relies on a strong non-standard interactive assumption and proofs are only sketched.

Enzmann and Schneider introduce two privacy-friendly loyalty systems for electronic market places in [16]. Their counter-based solution corresponds to our notion of a multi-use token system. Their scheme builds on RSA blind signatures to provide unlinkability and, at first sight, appears to be very efficient: a token basically consists of two elements from $\mathbb{Z}_n$, collecting points requires two exponentiations on the customer's side and redeeming a token only results in communication costs for the customer. However, the bit size of the exponents involved on the customer's side grows linearly with number of points k to be issued or stored in a token, i.e., the exponent size equals $k \log(e)$, where e is the public RSA exponent. Thus, exponents can quickly become pretty large. Unfortunately, the authors also do not provide any security or privacy proof for their system nor a formal framework.

Recently, Gong et al. proposed a privacy-preserving incentive-based demand and response scheme [19] building on identity-based signatures, partially-blind signatures as well as proofs of knowledge. The integrated incentive mechanism is only a smaller part of the overall scheme. It does not match our notion of a multi-use token scheme but still allows to accumulate incentives: The idea behind this mechanism is that incentives granted to a user are first centrally collected in a pseudonymous user account at the operators side. A user can then anonymously withdraw a certain amount from this account (showing that he owns the account) leading to new token and redeem it using his real identity. The composition of these two steps is called a settlement. Due to the high computational effort of generating and revoking pseudonyms, the use of long-term pseudonyms is proposed. There are several privacy concerns with this approach. First, the operator certainly learns which incentives a pseudonymous user receives from which parties. Second, by observing all withdrawal and redeeming transactions he may also identify the real user ID behind a pseudonym. A settlement requires 66 group exponentiations and 8 pairing computations at the user's side, while an incentive token essentially consists of one group element and four exponents. The paper lacks a formal treatment of security and privacy.

In [12], a scheme is proposed allowing customers to buy database records anonymously and unlinkably using digital rechargeable wallets. This interactive protocol is based on two different signature schemes [7, 9], the set membership protocol from [11], and other zero-knowledge proofs. Concrete efficiency figures are not given and cannot be easily extracted since zero-knowledge proofs are not described in detail. A drawback of their system is that a customer needs to store signatures on all possible balance values that may ap-

2 In a single-use token system, by looking at the redeemed tokens, the operator may learn the number of collection transactions a user did and how many points have been granted to him per transaction. This may allow the operator to, e.g., deduce where points have been issued and what has been purchased.

pear in the system (which are made public by the operator) in order to execute the set of membership protocol. This lowers the practical applicability of the wallet mechanism in scenarios with resource-constrained devices. In contrast to our incentive scheme, their rechargeable wallet mechanism is an interactive protocol and relies on specific instantiations of cryptographic building blocks. We leave it as an open problem to study whether the techniques used in [12] may lead to an alternative construction of a BBA scheme (which seems plausible).

The results in [14] are only distantly related to our work. Here the authors introduce stateful anonymous credentials, where the state of such a credential is encoded as an attribute and may be updated according to a well-defined policy. It is not obvious how such credentials can be used to implement a BBA scheme.

Single-use token systems. In [18] Blanco-Justicia and Domingo-Ferrer propose a loyalty system which uses partially-blind signatures in pairing-based groups to ensure anonymity. Issuing a token results in essentially two exponentiations for the customer, where a token consists of one group element and two exponents. Redeeming a token only induces communication costs on the customer's side. Unfortunately, the authors do not provide a formal model for loyalty systems nor a formal security analysis.

In [29] a privacy-preserving communication and incentive system for V2G, called P^2, is proposed. Here an incentive essentially consists in a partially-blind signature generated by the incentive issuer. When redeeming a token, the identity of the issuer is revealed. Thus, when a user redeems multiple tokens at the same time, his corresponding movements during a certain period can be traced.

In [25], Li and Cao propose a privacy-aware incentive scheme for PPS. Their single-use token mechanism is based on RSA blind signatures. The paper lacks a formal treatment of security and privacy. In fact, it seems that the ID of an incentive token (called credit token identifier) being the message to be blindly signed is actually revealed to the token issuer what makes the use of blind signatures pointless. Hence, as this ID also needs to be revealed when redeeming the incentive, issuing and redemption of an incentive are linkable.

Very recently, Milutinovic et al. presented uCentive, an unlinkable multi-purpose incentive scheme [26]. uCentive seems to be fully implemented on a smartphone and some performance figures are given. For example, the authors state that earning and redeeming a single uCent (e.g., a loyalty point) requires about 150 and 200 ms on an off-the-shelf Android smartphone. Again, uCentive's security and privacy properties are only informally stated and no rigorous proofs are given.

Basic Notation

Throughout the paper, $k \in \mathbb{N}$ denotes a security parameter. For a finite set $\mathcal{S}$, we denote by $s \leftarrow \mathcal{S}$ the process of sampling s uniformly from $\mathcal{S}$. For a probabilistic algorithm $\mathcal{A}$, we denote with $\mathcal{R}_{\mathcal{A}}$ the space of $\mathcal{A}$'s random coins. $y \leftarrow \mathcal{A}(x;r)$ denotes the process of running $\mathcal{A}$ on input x and with uniform randomness $r \in \mathcal{R}_{\mathcal{A}}$, and assigning y the result. We write $y \leftarrow \mathcal{A}(x)$ for $y \leftarrow \mathcal{A}(x;r)$ with uniform R. If $\mathcal{A}$'s running time is polynomial in k, then $\mathcal{A}$ is called probabilistic polynomial-time (PPT). We call a function η negligible if for every polynomial p there exists k_0 such that for all $k \geq k_0$ holds $|\eta(k)| \leq \frac{1}{p(k)}$.

2 Black-Box Accumulators

Before formally defining black-box accumulation (BBAs) schemes, we first give an informal overview of how BBAs are intended to be used by the different parties to realize privacy-preserving point accumulation and redemption.

Informal description. Typically, there will be the following parties involved: an issuer $\mathcal{I}$ who generates and redeems accumulation tokens of multiple users $\mathcal{U}_i$ as well as a number of accumulators $\mathcal{ACC}_j$ with whom users interact to add points to their accumulation tokens. $\mathcal{I}$ and $\mathcal{ACC}_j$ share the same secret key in order to generate tokens and add points to these tokens, respectively. They may even coincide in some scenarios. The corresponding public key is used by the issuer and the users (or potentially some external party, e.g., an authority) to verify that a given token has a certain claimed value. Users are not required to generate any key pair up front in order to participate in the system. However, they will generate some short-term secrets on-the-fly to blind their accumulation tokens for the different transactions.

More precisely, first the following steps are performed once to setup the system:

– *Parameter generation:* First, common system parameters are generated by running a setup algorithm $CRS \leftarrow \mathsf{Setup}(1^k)$. (For example, we will later describe a construction where the CRS consists of a

description of an algebraic group and a common reference string for a non-interactive zero-knowledge proof system.) We will assume that these parameters are generated in a trustworthy environment, that is, not by a possibly malicious issuer or possibly malicious users. This can be realized in practice either by assuming an external trustworthy party that generates these parameters once, or by letting a set of mutually distrusting parties perform a multiparty computation protocol to compute the *CRS*.

– *Issuer/Accumulator key generation:* Once the *CRS* is fixed, the issuer $\mathcal{I}$ generates a key pair by running a key generation algorithm $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$. We assume that the public key pk is publicly available to all users $\mathcal{U}_i$. The secret key sk is shared with all $\mathcal{ACC}_j$.

After that, the different parties will be able to engage in the following protocols to generate tokens, add values to them, and redeem claimed amounts.

– *Token issuing:* $\mathcal{I}$ can generate a new accumulator token τ with initial value v_0 (usually equal to 0) and unique serial number s by running some algorithm $\mathsf{Issue}(sk, s, v_0)$. A user $\mathcal{U}$ which is given τ along with s and v_0 may verify that indeed τ is a valid accumulation token by running a verification algorithm $\mathsf{CheckTokVal}(pk, \tau, s, v_0) = 1$.

– *Point accumulation:* Let τ denote the accumulation token of user $\mathcal{U}$, and assume that $\mathsf{CheckTokVal}(pk, \tau, s, w) = 1$, thus, τ has value w. In order to add a value v to τ, $\mathcal{U}$ and $\mathcal{ACC}$ perform the following steps:

1. $\mathcal{U}$ first runs a blinding algorithm $(\{\tau\}, r) \leftarrow \mathsf{Blind}(pk, \tau, s, w)$ to create a blinded version of the accumulation token τ, denoted by $\{\tau\}$, and an unblinding trapdoor r (i.e., a secret value only known to the user which is required to remove the blinding after accumulation). The blinded token $\{\tau\}$ is sent to $\mathcal{ACC}_j$. Tokens are blinded in order to achieve unlinkability of executions of the accumulation protocol.
2. $\mathcal{ACC}$ runs $\{\tilde{\tau}\} \leftarrow \mathsf{Acc}(sk, \{\tau\}, v)$ and returns a token $\{\tilde{\tau}\}$. This token is supposed to have a value of $w + v$ points but might not be a valid "regular" token in the sense of $\mathsf{CheckTokVal}$ yet. To transform it to such a token an unblinding operation might be required.
3. $\mathcal{U}$ runs $\tilde{\tau} \leftarrow \mathsf{Unblind}(pk, \{\tilde{\tau}\}, s, r, w{+}v)$ to obtain an unblinded version of his updated token $\{\tilde{\tau}\}$. Note that $\mathsf{Unblind}$ may return $\perp$ in case it is not able to return a valid token in the sense of $\mathsf{CheckTokVal}$.

– *Token verification and redemption.* Finally, in order to convince any party (either the issuer or some other third party) that an accumulation token τ has a certain value w in a privacy-preserving way (that is, unlinkable to any execution of the accumulation protocol), user $\mathcal{U}$ runs algorithm $\rho \leftarrow \mathsf{PrepVer}(pk, \tau, s, w)$ to prepare a so-called verification token for τ. Then it sends (ρ, s, w) to the verifying party. A verification token can be publicly verified by running a verification algorithm Ver. Given (ρ, s, w), the token is accepted with serial number s and value w if and only if $\mathsf{Ver}(pk, \rho, s, w) = 1$.

Formal definition of BBA schemes. The following definition specifies the different algorithms we have already seen in use above in a more formal fashion.

Definition 2.1 (BBA Scheme)**.** *A non-interactive black-box accumulation scheme* BBA *with serial number space $\mathcal{S}$ and aggregation value space $\mathcal{V} \subset \mathbb{N}_0$ consists of the following PPT algorithms:*

– $\mathsf{Setup}(1^k)$ *takes a security parameter 1^k as input and returns some public parameters CRS, called the common reference string. We assume that CRS is given as implicit input to all other algorithms described below.*

– $\mathsf{Gen}(1^k)$ *takes a security parameter 1^k as as input and returns a public and private key pair (pk, sk).*

– $\mathsf{Issue}(sk, s, v_0)$ *is given a secret key sk, a serial number $s \in \mathcal{S}$, and a value $v_0 \in \mathcal{V}$ and returns an accumulation token τ.*

– $\mathsf{CheckTokVal}(pk, \tau, s, w)$ *is given a public key pk, accumulation token τ, serial number s, and a value $w \in \mathcal{V}$. It returns 0 or 1.*

– $\mathsf{Blind}(pk, \tau, s, w)$ *is given the public key pk, an accumulation token τ, a serial number $s \in \mathcal{S}$, and the current token value $w \in \mathcal{V}$ and returns $(\{\tau\}, r)$ or $\perp$, where $\{\tau\}$ is a blinded accumulation token and r is an unblinding-trapdoor.*

– $\mathsf{Acc}(sk, \{\tau\}, v)$ *takes the secret key sk, a blinded accumulation token $\{\tau\}$, and a value $v \in \mathcal{V}$ as input and returns an updated blinded accumulation token $\{\tilde{\tau}\}$ or $\perp$.*

– $\mathsf{Unblind}(pk, \{\tilde{\tau}\}, s, r, w)$ *is given the public key pk, a blinded accumulation token $\{\tilde{\tau}\}$, a serial number $s \in \mathcal{S}$, an unblinding-trapdoor r, and the current token value $w \in \mathcal{V}$ and returns an unblinded accumulation token $\tilde{\tau}$ or $\perp$.*

– $\mathsf{PrepVer}(pk, \tau, s, w)$ *is given the public key* pk*, an accumulation token* τ*, a serial number* $s \in \mathcal{S}$*, and the current token value* $w \in \mathcal{V}$ *and returns some verification token* ρ *for* τ *(which might be* τ *itself) or* $\perp$*.*

– $\mathsf{Ver}(pk, \rho, s, w)$ *takes the public key* pk*, a verification token* ρ*, a serial number* $s \in \mathcal{S}$*, and some value* $w \in \mathcal{V}$ *as input and returns 1 or 0.*

BBA *is correct if all the following properties hold for all* $k, n \in \mathbb{N}$*,* $s \in \mathcal{S}$*,* $CRS \leftarrow \mathsf{Setup}(1^k)$*, and* $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$*.*

Correctness of issuing. *For all values* $v_0 \in \mathcal{V}$ *and tokens* $\tau \leftarrow \mathsf{Issue}(sk, s, v_0)$ *holds that* $\Pr[\mathsf{CheckTokVal}(pk, \tau, s, v_0) = 1] = 1$*.*

Correctness of accumulation. *For all* $w \in \mathcal{V}$ *and tokens* τ *with* $\Pr[\mathsf{CheckTokVal}(pk, \tau, s, w) = 1] = 1$ *and all* $v \in \mathcal{V}$ *with* $w + v \in \mathcal{V}$*, we have*

$$\Pr\left[\mathsf{CheckTokVal}(pk, \tilde{\tau}, s, w + v) = 1\right] = 1$$

Here, $\tilde{\tau}$ *is computed by first running* $(\{\tau\}, r) \leftarrow \mathsf{Blind}(pk, \tau, s, w)$*, and then* $\{\tilde{\tau}\} \leftarrow \mathsf{Acc}(sk, \{\tau\}, v)$*, followed by* $\tilde{\tau} \leftarrow \mathsf{Unblind}(pk, \{\tilde{\tau}\}, s, r, v + w)$*.*

Correctness of token verification. *For all* $w \in \mathcal{V}$ *and tokens* τ *with* $\Pr[\mathsf{CheckTokVal}(pk, \tau, s, w) = 1] = 1$ *we have* $\Pr\left[\mathsf{Ver}(pk, \rho, s, w) = 1\right] = 1$*, where* $\rho \leftarrow \mathsf{PrepVer}(pk, \tau, s, w)$*.*

Security. Our security definition will capture the following notion: an adversary (who possibly impersonates a group of colluding malicious users) should not be able to redeem more points than actually issued to him. To reach his goal the adversary is not restricted to use a single accumulation token but he may use as many tokens in parallel as he needs. This seems to properly model an adversary's capabilities in many practical scenarios. Note that we will not consider it a security breach if an adversary (or regular user) may be able to "transfer" points from one accumulation token to another as long as these point cannot be redeemed twice.[3]

Our formal security definition is inspired by the indistinguishability-based security definition for a prepayments with refunds scheme proposed in [27]. In security experiment (cf. Figure 1), we consider a PPT adversary $\mathcal{A}$ interacting with an (honest) Issue oracle issuing m tokens and an (honest) Acc oracle performing

Experiment $\mathsf{Exp}^{\text{acc-sec}}_{\mathsf{BBA},\mathcal{A}}(k)$

$CRS \leftarrow \mathsf{Setup}(1^k)$

$(pk, sk) \leftarrow \mathsf{Gen}(1^k)$

$(\rho_1^*, s_1^*, w_1^*), \ldots, (\rho_\ell^*, s_\ell^*, w_\ell^*)$

$\leftarrow \mathcal{A}^{\mathsf{Issue}(sk,\cdot,\cdot),\mathsf{Acc}(sk,\cdot,\cdot)}(1^k, CRS, pk)$

Let $(s_1, u_1), \ldots, (s_m, u_m)$ denote $\mathcal{A}$'s inputs to the Issue oracle and $v_1, \ldots, v_n$ its inputs as third parameter to the Acc oracle during the execution.

The experiment returns 1 iff the following conditions are satisfied:

– $s_1^*, \ldots, s_\ell^*$ are pairwise distinct elements of $\mathcal{S}$

– $w_1^*, \ldots, w_\ell^*$ are elements of $\mathcal{V}$

– for all $1 \le i \le \ell$ it holds that $1 \leftarrow \mathsf{Ver}(pk, \rho_i, s_i^*, w_i^*)$

– there exists $i \in \{1, \ldots, \ell\}$ such that $s_i^* \notin \{s_1, \ldots, s_m\}$ or $\sum_{i=1}^{\ell} w_i^* > \sum_{i=1}^{m} u_i + \sum_{i=1}^{n} v_i$

Fig. 1. A security experiment for BBAs.

n accumulations. Note that all other operations do not depend on the secret key of the issuer and, thus, the adversary can perform those operations on its own. In this way, the adversary may generate an arbitrary (but polynomial) number of accumulation tokens which he can use to collect an arbitrary (but polynomial) number of points. In the end, the adversary outputs an arbitrary number of verification tokens including the claimed serial numbers and accumulated points corresponding to these tokens: $(\rho_1^*, s_1^*, w_1^*), \ldots, (\rho_\ell^*, s_\ell^*, w_\ell^*)$. He wins if all tokens are valid and (1) one of these tokens belongs to a new serial number which has not been issued or (2) the claimed total amount of collected points is higher than what has actually been issued during the game. Of course, in the formal definition also some side conditions need to be satisfied (a claimed serial number may not appear twice, etc.).

Definition 2.2 (Security)**.** *A BBA scheme* BBA *is called accumulation secure if for all PPT adversaries* $\mathcal{A}$ *in the experiment* $\mathsf{Exp}^{\text{acc-sec}}_{\mathsf{BBA},\mathcal{A}}(k)$ *from Figure 1 the advantage of* $\mathcal{A}$ *defined by*

$$\mathsf{Adv}^{\text{acc-sec}}_{\mathsf{BBA},\mathcal{A}}(k) := \Pr[\mathsf{Exp}^{\text{acc-sec}}_{\mathsf{BBA},\mathcal{A}}(k) = 1]$$

is negligible in k*.*

Privacy (intuition). Intuitively, we want to ensure that it is not possible to "track users", that is, to link a given accumulation token to any accumulation or oper-

3 In fact, it seem to be pretty difficult to model non-transferability in a scenario where an adversary may use multiple tokens in parallel as blinded accumulation transactions do not allow to associate an issued refund with a specific token.

ation. This should hold even if the issuer and the accumulator collude and the issuer generates its public key in a malicious way (to enable tracking of users).

To formally capture this intuition, we model the collusion of issuer and accumulator as an adversary in the security experiment described in Figure 2. We want to ensure that any two blinded tokens that the adversary receives in the accumulation protocol "look independent of each other" (for the adversary), which implies the desired unlinkability. Moreover, we want that it is not possible to link a given verification token to any previous accumulation operation.

To this end, we compare this "real" experiment with an "ideal" experiment (defined in Figure 3). The main difference in the ideal experiment is that blinded tokens are generated in a different way. In particular, all blinded tokens are generated *independent* of each other, and independent of any sequence number or token value, only depending on the adversary's public key. Moreover, the token verification at the end of the experiment depends only on the final accumulated value and serial number of the token. Thus, blinded tokens and verification tokens are trivially unlinkable in the "ideal" experiment.

If we can show that indeed both these experiments are computationally indistinguishable for a given BBA scheme, then this implies that no collusion of issuer and accumulator is efficiently able to link any two transactions (neither two accumulation operations, nor a verification operations and any accumulation operation).

Privacy definition. More precisely, we define privacy by two experiments $\mathsf{Exp}^{\text{priv-real}}_{\mathsf{BBA},\mathcal{A}}$ and $\mathsf{Exp}^{\text{priv-ideal}}_{\mathsf{BBA},\mathcal{A}}$ involving an adversary $\mathcal{A}$ and a BBA scheme BBA, cf. Figures 2 and 3. In the "real" experiment $\mathsf{Exp}^{\text{priv-real}}_{\mathsf{BBA},\mathcal{A}}$ we consider an adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ which first generates a public key pk along with a triplet (τ_0, s, w_0). The triplet consists of an accumulation token τ_0 with serial number s and initial token value w_0. Note that τ_0 *may* be generated by $\mathcal{A}$ by running $\tau_0 \leftarrow \mathsf{Issue}(sk, s, w_0)$ for arbitrary (s, w_0), however, $\mathcal{A}$ may also have computed τ_0 completely differently, possibly maliciously in order to break privacy. Then the adversary interacts with a user $\mathcal{U}$ to accumulate points.

In the "ideal" experiment $\mathsf{Exp}^{\text{priv-ideal}}_{\mathsf{BBA},\mathcal{A}}$, we replace procedures Setup, $\mathsf{PrepVer}$, and Blind with "simulated" procedures $\mathsf{SimSetup}$, $\mathsf{SimPrepVer}$, and $\mathsf{SimBlind}$. The main difference is that $\mathsf{SimSetup}$ produces a CRS along with an additional trapdoor td. This trapdoor is used in $\mathsf{SimBlind}$ and $\mathsf{SimPrepVer}$. In particular, note that the blinded tokens $\{\tau_{i-1}\}$ computed by $\mathsf{SimBlind}$ in experiment $\mathsf{Exp}^{\text{priv-ideal}}_{\mathsf{BBA},\mathcal{A}}$ depend only on pk, and thus in particular are *independent* of s, w, and any previously received tokens. This guarantees unlinkability of different executions of the accumulation protocol. Moreover, $\mathsf{SimPrepVer}$ does not depend on any previously received tokens, but only on the serial number s and the corresponding accumulated value w. This guarantees that the adversary learns only the trivial information that the token with serial number s has value w, but it is impossible to link it to previous executions of the accumulation protocol, because they are independent.

Experiment $\mathsf{Exp}^{\text{priv-real}}_{\mathsf{BBA},\mathcal{A}}(1^k)$

$CRS \leftarrow \mathsf{Setup}(1^k)$

$(pk, (\tau_0, s, w_0), state_0) \leftarrow \mathcal{A}_0(1^k, CRS)$

If $\mathsf{CheckTokVal}(pk, \tau_0, s, w_0) = 0$ set $\tau_0 := \bot$

$state_1 \leftarrow \mathcal{A}_1^{\mathcal{U}(pk,(\tau_0,s,w_0))}(state_0)$

$\rho \leftarrow \mathsf{PrepVer}(pk, \tau_n, s, w_n)$

$b \leftarrow \mathcal{A}_2(state_1, \rho)$

Here $\mathcal{U}$ is a stateful oracle, which interacts with $\mathcal{A}_1$ exactly n times to mimic a user of the BBA scheme. The i-th interaction between $\mathcal{A}_1$ and $\mathcal{U}$ proceeds as follows:

1. $\mathcal{A}_1$ outputs $v_i \in \mathcal{V}$,
2. $\mathcal{U}$ runs $(\{\tau_{i-1}\}, r_i) \leftarrow \mathsf{Blind}(pk, \tau_{i-1}, s, w_{i-1})$ and outputs $\{\tau_{i-1}\}$ to $\mathcal{A}_1$. $\mathcal{A}_1$ responds with an accumulated blinded token $\{\tilde{\tau}_i\}$.
3. $\mathcal{U}$ sets $w_i := w_{i-1} + v_i$ and computes τ_i by running $\tau_i \leftarrow \mathsf{Unblind}(pk, \{\tilde{\tau}_i\}, s, r_i, w_i)$.

Fig. 2. A real experiment for BBAs.

For the privacy definition it is sufficient to consider a single serial number, because it implies privacy in a setting with many serial numbers.

Definition 2.3. *We say that a BBA scheme* BBA *achieves* privacy*, if there exist PPT algorithms* $(\mathsf{SimSetup}, \mathsf{SimBlind}, \mathsf{SimPrepVer})$ *such that for all PPT adversaries* $\mathcal{A}$ *in the experiments from Figures 2 and 3, the advantage* $\mathsf{Adv}^{\text{priv}}_{\mathsf{BBA},\mathcal{A}}(k)$ *of* $\mathcal{A}$ *defined by*

$$\left|\Pr[\mathsf{Exp}^{\text{priv-real}}_{\mathsf{BBA},\mathcal{A}}(k) = 1] - \Pr[\mathsf{Exp}^{\text{priv-ideal}}_{\mathsf{BBA},\mathcal{A}}(k) = 1]\right|$$

is negligible in k.

Experiment $\mathsf{Exp}^{\mathsf{priv\text{-}ideal}}_{\mathsf{BBA},\mathcal{A}}(1^k)$

$(CRS, td) \leftarrow \mathsf{SimSetup}(1^k)$

$(pk, (\tau_0, s, w_0), state_0) \leftarrow \mathcal{A}_0(1^k, CRS)$

If $\mathsf{CheckTokVal}(pk, \tau_0, s, w_0) = 0$ set $\tau_0 := \perp$

$state_1 \leftarrow \mathcal{A}_1^{\mathcal{U}_{\mathsf{sim}}(pk,s,w_0)}(state_0)$

$\rho \leftarrow \mathsf{SimPrepVer}(td, pk, s, w_n)$

$b \leftarrow \mathcal{A}_2(state_1, \rho)$

Here $\mathcal{U}_{\mathsf{sim}}$ is a stateful oracle, which interacts with $\mathcal{A}_1$ exactly n times. The i-th interaction between $\mathcal{A}_1$ and $\mathcal{U}_{\mathsf{sim}}$ proceeds as follows:

1. $\mathcal{A}_1$ outputs $v_i \in \mathcal{V}$,
2. $\mathcal{U}_{\mathsf{sim}}$ runs $(\{\tau_{i-1}\}, r_i) \leftarrow \mathsf{SimBlind}(td, pk)$ and outputs $\{\tau_{i-1}\}$ to $\mathcal{A}_1$. $\mathcal{A}_1$ responds with $\{\tilde{\tau}_i\}$.
3. $\mathcal{U}_{\mathsf{sim}}$ sets $w_i := w_{i-1} + v_i$ and computes τ_i by running $\tau_i \leftarrow \mathsf{Unblind}(pk, \{\tilde{\tau}_i\}, s, r_i, w_i)$.

Fig. 3. An ideal privacy experiment for BBAs.

3 A Generic Construction

In the following, we will present our generic construction of a BBA scheme building on abstract cryptographic building blocks. We start by reviewing these standard building blocks in Section 3.1 before considering the construction in Section 3.2. The proofs of security and privacy for the scheme can be found in the appendix in Sections B and C, respectively.

3.1 Building Blocks

We will make use of homomorphic commitments, (ordinary) digital signatures, and non-interactive proofs of knowledge. In the following, these building blocks are only described in an informal fashion. Formal definitions can be found in Appendix A.

Commitment Schemes. A commitment scheme allows a user to commit to a message and publish the result, called commitment, in a way that the message is hidden from others, but also the user cannot change the message he has committed to afterwards when he opens the commitment.

More precisely, a non-interactive commitment schemes consists of two algorithms $\mathsf{Setup}_{\mathsf{com}}$ and Com. $\mathsf{Setup}_{\mathsf{com}}$ generates some public parameters CRS_{com} for the scheme. Com takes a message m and a randomness r besides the public parameters and outputs a commitment c to m. To open the commitment, i.e., prove that c contains m, it suffices to reveal m, r (called *opening* information) and check that $\mathsf{Com}(CRS_{\mathsf{com}}, m, r)$ indeed yields c. On the one hand, the value c should not reveal information about m (*hiding*) to an efficient adversary. On the other hand, it should also be hard to find $(m', r') \neq (m, r)$ such that $c = \mathsf{Com}(CRS_{\mathsf{com}}, m', r')$ (*binding*). A commitment scheme is called *additively homomorphic* if given two commitments $c = \mathsf{Com}(CRS_{\mathsf{com}}, m, r)$ and $c' = \mathsf{Com}(CRS_{\mathsf{com}}, m', r')$, one can efficiently combine these commitments, using an algorithm CAdd, resulting in c'' such that $c'' = \mathsf{Com}(CRS_{\mathsf{com}}, m + m', r + r')$.

Digital Signatures. A digital signature scheme consists of a key generation algorithm $\mathsf{Gen}_{\mathsf{sig}}$ which outputs a secret key sk and a public key pk, a signing algorithms Sign which outputs a signature σ on input of a message m and sk, and the verification algorithm $\mathsf{Ver}_{\mathsf{sig}}$ which on input pk, m, and σ decides whether σ is a correct signature on m. A signature scheme is called *EUF-CMA secure* if for any efficient adversary given pk and access to a signature oracle which signs arbitrary messages of his choice, the adversary is not able to compute a signature to a new message.

NIZK-PoK. Let R be a witness relation for some NP language $L_R = \{x \mid \exists z \text{ s.t. } (x, z) \in R\}$ (e.g, $(x, z) \in R$ if $g^z = x$ for some fixed group generator g). Informally speaking, a zero-knowledge proof of knowledge scheme is a system that allows a prover P to convince a verifier V that he knows a *witness* z to some x given to V, i.e., that (x, z) satisfies the relation R, without V learning anything beyond that fact. In a non-interactive zero-knowledge proof of knowledge (NIZK-PoK) only one message is sent from P to V for that purpose.

More precisely, a NIZK-PoK consists of three algorithms $\mathsf{Setup}_{\mathsf{pok}}, \mathsf{Prove}, \mathsf{Ver}_{\mathsf{pok}}$. $\mathsf{Setup}_{\mathsf{pok}}(1^k)$ outputs a common reference string (CRS) CRS. $\mathsf{Prove}(CRS, x, z)$ on input of a statement x and a witness z outputs a proof π, and $\mathsf{Ver}_{\mathsf{pok}}(CRS, x, \pi)$ outputs 1 if π is a valid proof for $x \in L_R$, and 0 otherwise. The proof system is called *complete* if $\mathsf{Ver}_{\mathsf{pok}}(CRS, x, \pi)$ always accepts proofs generated by $\mathsf{Prove}(CRS, x, z)$ for $(x, z) \in R$. It is called *extractable* (or sound) if there exists an extractor algorithm that is able to (1) output some special CRS (including trapdoor information) indistinguishable from a real CRS and (2) extract a witness from a valid proof that has been generated using this special CRS. (As we can extract a witness from any valid proof, we can be sure that the prover must have known the witness.) The proof system is called *zero-knowledge* if there

exists a simulator algorithm that is able to (1) output some special CRS (including trapdoor information) indistinguishable from a real CRS and (2) by means of this special CRS and the trapdoor to generate a valid proof for $x \in L_R$ without knowing a witness for x.

3.2 Our Construction

In the following, we present a generic construction of a BBA scheme building on homomorphic commitments, signatures, and NIZK-PoKs. The underlying idea of our scheme is as follows: An initial accumulation token τ essentially consists of a commitment c_1 on the serial number s of the token, a commitment c_2 on its initial value v_0, and a signature σ on these two commitments generated by the issuer. Note that all these values, and in particular τ, are known to the issuer. The signature certifies that τ is a valid token with value v_0. To blind τ before accumulation, the two commitments are re-randomized, yielding c_1' and c_2'. Additionally, the signature is replaced by a NIZK-PoK π showing that the new commitments are indeed re-randomizations of commitments for which the user knows a valid signature. This yields the blinded token $\{\tau\}$. During accumulation, the homomorphic property of the commitment c_2' is used to blindly add the value v to the token, yielding a new commitment c_2''. To certify the new token value, c_1' and c_2'' are simply signed by the accumulator. The resulting token $\{\tilde{\tau}\}$ is a valid accumulation token again which does (actually) not need to be unblinded to be further used to accumulate additional values. To prepare an accumulation token for verification, the blinding procedure as described above is applied but additionally the content of the token, i.e., the serial number s and the value w, are revealed.

Details of our scheme BBA. The algorithms of our BBA scheme are described in Figures 4 to 6 in more detail. More precisely, Figure 4 specifies the setup and key generation algorithms, Figure 5 describes the algorithms required by the issuer/accumulator party, and Figure 6 gives the details of the algorithms required by the user.

The scheme has serial number space $\mathcal{S} \subset \mathbb{N}_0$ and aggregation value space $\mathcal{V} \subset \mathbb{N}_0$, which both may depend on the security parameter. As an important ingredient, we need an additively homomorphic commitment scheme $\mathsf{COM} := (\mathsf{Setup}_{\mathsf{com}}, \mathsf{Com}, \mathsf{CAdd})$ that is perfectly binding and computationally hiding (cf. Definition A.1). Its message space needs to include $\mathcal{S}$ and $\mathcal{V}$ and addition of messages needs to coincide with the usual addition over $\mathbb{N}_0$.[4] Furthermore, we make use of an EUF-CMA secure signature scheme $\mathsf{SIG} := (\mathsf{Gen}_{\mathsf{sig}}, \mathsf{Sign}, \mathsf{Ver}_{\mathsf{sig}})$ to sign commitments. Finally, we need a non-interactive zero-knowledge proof of knowledge (NIZK-PoK) to show that two given commitments c_1', c_2' are re-randomizations of commitments c_1, c_2 for which a valid signature σ is known. More precisely, we consider a NIZK-PoK $\mathsf{POK} = (\mathsf{Setup}_{\mathsf{pok}}, \mathsf{Prove}, \mathsf{Ver}_{\mathsf{pok}})$ for a witness relation R describing the language $L^{\mathsf{pok}}_{(CRS_{\mathsf{com}}, pk)}$ defined by

$$\left\{ (c_1', c_2') \;\middle|\; \begin{array}{l} \exists c_1, c_2, \sigma, t_1', t_2' : \\ c_1' = \mathsf{CAdd}(c_1, \mathsf{Com}(CRS_{\mathsf{com}}, 0, t_1')) \\ c_2' = \mathsf{CAdd}(c_2, \mathsf{Com}(CRS_{\mathsf{com}}, 0, t_2')) \\ \mathsf{Ver}_{\mathsf{sig}}(pk, \sigma, c_1 || c_2) = 1 \end{array} \right\}$$

Here, $CRS_{\mathsf{com}} \leftarrow \mathsf{Setup}_{\mathsf{com}}(1^k)$ and $(sk, pk) \leftarrow \mathsf{Gen}_{\mathsf{sig}}(1^k)$. We demand that for any $(x, z) \in R$ the witness z contains c_1, c_2, and σ in "plain" but not necessarily the commitment randomness t_1' and t_2'. That means, the witness may be of the form $z = (c_1, c_2, \sigma, \phi_1(t_1'), \phi_2(t_2'))$, where ϕ_1, ϕ_2 are some functions from $\mathcal{R}$ to $\mathcal{R}' \subset \{0,1\}^{\mathsf{poly}(k)}$ for some polynomial poly. In the simplest case, $\mathcal{R}' = \mathcal{R}$ and $\phi_1 = \phi_2$ is the identity function. For understanding the construction it is best to think of this case. However, when we use Groth-Sahai proofs for implementing the NIZK-PoK later, ϕ_1 and ϕ_2 will be some injective functions, mapping t_1' and t_2' to a unique implicit representation, respectively. For our security proofs, we will only need to be able to extract c_1, c_2, and σ and do not care about the rest of the witness.

$\underline{\mathsf{Setup}(1^k)}$	$\underline{\mathsf{Gen}(1^k)}$
$CRS_{\mathsf{com}} \leftarrow \mathsf{Setup}_{\mathsf{com}}(1^k)$	$(pk, sk) \leftarrow \mathsf{Gen}_{\mathsf{sig}}(1^k)$
$CRS_{\mathsf{pok}} \leftarrow \mathsf{Setup}_{\mathsf{pok}}(1^k)$	return (pk, sk)
$CRS := (CRS_{\mathsf{com}}, CRS_{\mathsf{pok}})$	
return CRS	

Fig. 4. Our BBA scheme: Setup and Key Generation

Correctness. Let $s \in \mathcal{S}$, $v_0, v, w, v+w \in \mathcal{V}$ and let us assume the system has been properly setup, i.e., $CRS \leftarrow \mathsf{Setup}(1^k)$, $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$.

4 Actually, to be even more generic, we could let $\mathcal{S}$ be an arbitrary set for which there is a (separate) re-randomizable commitment scheme.

Issue(sk, s, v_0)
- $t_1, t_2 \leftarrow \mathcal{R}$
- $c_1 :=$ **Com**$(CRS_{\mathsf{com}}, s, t_1)$
- $c_2 :=$ **Com**$(CRS_{\mathsf{com}}, v_0, t_2)$
- $\sigma :=$ **Sign**$(sk, c_1 || c_2)$
- **return** $\tau := (c_1, c_2, \sigma, t_1, t_2)$

Acc$(sk, \{\tau\}, v)$
- **parse** $(c_1', c_2', \pi) := \{\tau\}$
- **If** **Ver**$_{\mathsf{pok}}(CRS_{\mathsf{pok}}, \pi, (c_1', c_2')) = 0$ **then return** $\perp$
- $c_1 := c_1'$, $c_2 :=$ **CAdd**$(c_2',$ **Com**$(CRS_{\mathsf{com}}, v, 0))$
- $\sigma :=$ **Sign**$(sk, c_1 || c_2)$
- **Return** $\{\tilde{\tau}\} := (c_1, c_2, \sigma)$

Ver(pk, ρ, s, w)
- **parse** $(c_1', c_2', r_1, r_2, \pi) := \rho$
- **if** $c_1' =$ **Com**$(CRS_{\mathsf{com}}, s, r_1)$ **and**
  - $c_2' =$ **Com**$(CRS_{\mathsf{com}}, w, r_2)$ **and**
  - **Ver**$_{\mathsf{pok}}(CRS_{\mathsf{pok}}, \pi, (c_1', c_2')) = 1$
- **return** 1
- **else return** 0

Fig. 5. Our BBA scheme: Issuer/Accumulator Algorithms

CheckTokVal(pk, τ, s, w)
- **parse** $(c_1, c_2, \sigma, r_1, r_2) := \tau$
- **if** $c_1 =$ **Com**$(CRS_{\mathsf{com}}, s, r_1)$ **and**
  - $c_2 =$ **Com**$(CRS_{\mathsf{com}}, w, r_2)$ **and**
  - **Ver**$_{\mathsf{sig}}(pk, \sigma, c_1 || c_2) = 1$
- **return** 1
- **else return** 0

Blind(pk, τ, s, w)
- **If** $\tau = \perp$ **return** $\perp$
- **parse** $(c_1, c_2, \sigma, r_1, r_2) := \tau$
- $t_1', t_2' \leftarrow \mathcal{R}$
- $c_1' :=$ **CAdd**$(c_1,$ **Com**$(CRS_{\mathsf{com}}, 0, t_1'))$
- $c_2' :=$ **CAdd**$(c_2,$ **Com**$(CRS_{\mathsf{com}}, 0, t_2'))$
- $\pi :=$ **Prove**$(CRS_{\mathsf{pok}}, (c_1', c_2'), (c_1, c_2, \sigma, \phi_1(t_1'), \phi_2(t_2')))$
- $\{\tau\} := (c_1', c_2', \pi)$, $r := (t_1 + t_1', t_2 + t_2')$
- **return** $(\{\tau\}, r)$

Unblind$(pk, \{\tilde{\tau}\}, s, r, w)$
- **parse** $(c_1, c_2, \sigma) := \{\tilde{\tau}\}$
- **parse** $(r_1, r_2) := r$
- $\tilde{\tau} := (c_1, c_2, \sigma, r_1, r_2)$
- **if** **CheckTokVal**$(pk, \tilde{\tau}, s, w) \neq 1$ **return** $\perp$
- **return** $\tilde{\tau}$

PrepVer(pk, τ, s, w)
- **if** $\tau = \perp$ **return** $\perp$
- $((c_1', c_2', \pi), (r_1, r_2)) \leftarrow$ **Blind**(pk, τ, s, w)
- **return** $\rho := (c_1', c_2', r_1, r_2, \pi)$

Fig. 6. Our BBA scheme: User Algorithms

Correctness of issuing: Consider the accumulation token $\tau = (c_1, c_2, \sigma, t_1, t_2) \leftarrow \mathsf{Issue}(sk, s, v_0)$. Here c_1 is a commitment on s with randomness t_1, c_2 is a commitment on v_0 with randomness t_2 and σ is a signature under sk on the two commitments. This is exactly what CheckTokVal checks, so clearly $\mathsf{CheckTokVal}(pk, \tau, s, v_0) = 1$.

Correctness of accumulation: Let us assume we have some $\tau = (c_1, c_2, \sigma, t_1, t_2)$ such that $\mathsf{CheckTokVal}(pk, \tau, s, w) = 1$. Blinding τ results in $\{\tau\} := (c_1', c_2', \pi)$, $r := (t_1 + t_1', t_2 + t_2')$, where due to the homomorphic property of COM, $c_1' = \mathsf{Com}(CRS_{\mathsf{com}}, s, t_1 + t_1')$ and $c_2' = \mathsf{Com}(CRS_{\mathsf{com}}, w, t_1 + t_1')$ are re-randomizations of c_1 and c_2. Moreover, π is a NIZK-PoK showing that c_1' and c_2' are indeed re-randomizations for which the user knows the original commitments c_1 and c_2 as well as a signature σ on those. This proof is valid as it has been generated with a correct witness $z = (c_1, c_2, \sigma, \phi_1(t_1'), \phi_2(t_2'))$. The result of $\mathsf{Acc}(sk, \{\tau\}, v)$ is $\{\tilde{\tau}\} := (c_1'', c_2'', \sigma'')$ as the verification of π will pass when everything is generated honestly. As COM is additively homomorphic, it holds that $c_2'' = \mathsf{Com}(CRS_{\mathsf{com}}, w + v, t_1 + t_1')$. The commitment c_1'' equals c_1' from before and σ'' is a valid signature on c_1'' and c_2''. Finally, $\mathsf{Unblind}(pk, \{\tilde{\tau}\}, s, r, w + v)$ will yield $\tilde{\tau} = (c_1'', c_2'', \sigma'', t_1 + t_1', t_2 + t_2')$, a valid accumulation token with value $w + v$. Hence $\mathsf{CheckTokVal}(pk, \tilde{\tau}, s, w + v)$ will return 1.

Correctness of token verification: Let us assume we have some $\tau = (c_1, c_2, \sigma, t_1, t_2)$ such that $\mathsf{CheckTokVal}(pk, \tau, s, w) = 1$. When we apply $\mathsf{PrepVer}(pk, \tau, s, w)$, we obtain $\rho := (c_1', c_2', r_1, r_2, \pi)$ which results from simply running $\mathsf{Blind}(pk, \tau, s, w)$. As we argued before, Blind will output valid re-randomized commitments c_1' and c_2' to s and w with randomness r_1 and r_2, respectively, as well as a valid NIZK-PoK π. Hence, $\mathsf{Ver}(pk, \rho, s, w) = 1$.

Security and Privacy. Informally speaking, the security of our scheme follows from the fact that the integrity and authenticity of an accumulation token is

protected at any time by a "chain of certificates" (realized by means of the binding property of the commitment, the security of the signature, and the soundness of the NIZK-PoK): The initial content of the two commitments which are part of the token, i.e., the serial number and the initial value, as well as the commitments itself are known to the issuer which certifies the token by signing the commitments. In the first accumulation, the accumulator can be sure that it has received a well-formed re-randomization of a certified token (due to the NIZK-PoK). It then manipulates the token by adding v points and certifies this altered token. This add-and-certify step is repeated until the token is finally redeemed, where the verifier can also be sure that it received a certified token.

Privacy essentially follows from the fact that the commitments belonging to a token are always re-randomized before they are handed over to an accumulator. This prevents a trivial way of linking a blinded token to the original token by means of the contained commitments. Moreover, the commitment scheme (computationally) hides the content of these randomized commitments. Finally, the zero-knowledge property of the proof of knowledge contained in the blinded token also does not reveal anything about the original commitments nor the signature on those commitments.

More formally, we can prove the following theorems. Their proofs can be found in Appendix B and C.

Theorem 3.1 (Accumulation Security)**.** *If* COM *is perfectly binding,* SIG *is EUF-CMA secure, and* POK *is extractable (sound), then* BBA *is accumulation secure.*

Theorem 3.2. *If* COM *is computationally hiding and* POK *is zero-knowledge, then* BBA *achieves privacy.*

4 Instantiation and Efficiency

Pairings. Both the Groth-Sahai-based proof system and the signature scheme that we will use to instantiate our generic construction are based on algebraic groups with bilinear pairing maps. Pairings are a widely-used standard tool in cryptography, therefore we recall their definition only briefly.

Let $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_T$ be groups of prime order p with generators g_1, g_2. Let $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$ be a map which (i) is efficiently computable, (ii) it holds that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $a, b \in \mathbb{Z}_p$, and (iii) $e(g_1, g_2)$ is a generator of $\mathcal{G}_T$.

Bilinear maps are useful, because they allow to compute a single "multiplication in the exponent". That is, given two group elements g_1^a and g_2^b, we can efficiently compute $e(g_1^a, g_2^b) = g_T^{ab}$, where $e(g_1, g_2) = g_T$.

Implicit notation. In the sequel we use the "implicit" notation introduced in [17] to simplify our notation. That is, for groups $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_T$ of prime order p with generators g_1, g_2, g_T, respectively, and pairing $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$, we define

$$[a]_i := g_i^a$$

for $a \in \mathbb{Z}_p$ and $i \in \{1, 2, T\}$. We generalize this notation to vectors and matrices, writing

$$[\mathbf{a}]_i := (g_i^{a_1}, \ldots, g_i^{a_n})^\top \quad \text{and} \quad [\mathbf{A}]_i := (g_i^{a_{u,v}})_{u,v} \in \mathcal{G}_i^{n \times m}$$

for $\mathbf{a} = (a_1, \ldots, a_n)^\top \in \mathbb{Z}_p^n$ and $\mathbf{A} = (a_{u,v})_{u,v} \in \mathbb{Z}_p^{n \times m}$. Note that we use bold-face notation for vectors and matrices.

We use additive notation to denote the group operation for group elements in implicit representation. That is,

$$g_i^a \cdot g_i^b = [a]_i + [b]_i$$

We use the multiplication symbol $\cdot$ to denote the application of the bilinear map for elements in implicit representation. That is,

$$[a]_1 \cdot [b]_2 = e(g_1^a, g_2^b) = g_T^{ab} = [ab]_T$$

for $e(g_1, g_2) = g_T$. This notation generalizes to vectors and matrices in the natural way.

4.1 Building Blocks

Let us instantiate the abstract building blocks introduced in Section 3.1.

Additively and multiplicatively homomorphic commitments. For $\mathcal{G} \in \{\mathcal{G}_1, \mathcal{G}_2\}$, we use the SXDH-based commitment scheme of [21] to commit to elements of $\mathcal{G}$. This commitment scheme can be viewed both as an additively-homomorphic commitment scheme with message space $\mathbb{Z}_p$, or alternatively as a multiplicatively-homomorphic scheme with message space $\mathcal{G}$. A commitment to a single $\mathcal{G}$-element (or $\mathbb{Z}_p$-element) consists of two elements of $\mathcal{G}$.

A commitment to $[m] \in \mathcal{G}$ (resp. $m \in \mathbb{Z}_p$) is computed as follows.

- $\mathsf{Setup}_{\mathsf{com}}$ defines $CRS_{\mathsf{com}} := [\mathbf{U}] = \left(\begin{smallmatrix} 1 & \beta \\ \alpha & \alpha\beta \end{smallmatrix}\right)$ for uniformly random $\alpha, \beta \leftarrow \mathbb{Z}_p$, and outputs CRS_{com}.

– Com takes as input $CRS_{\mathsf{com}} = [\mathbf{U}]$, message $[m] \in \mathcal{G}$, and randomness $\mathbf{r} \leftarrow \mathbb{Z}_p^2$, and computes the commitment as $\mathbf{c} := [\binom{0}{m} + \mathbf{U}\mathbf{r}] \in \mathcal{G}^2$.

Theorem 4.1 ([21])**.** *The above commitment scheme is perfectly binding and computationally hiding under the SXDH-assumption.*

Groth-Sahai Proofs for Pairing-Product Equations. Classical non-interactive proof systems for general NP languages are rather theoretical tools, because known constructions are too inefficient to be used in practice. At Eurocrypt 2008, Groth and Sahai [21] presented very efficient proof systems for certain, restricted languages defined over algebraic groups (actually modules, which is more general). In particular, Groth and Sahai described efficient proof systems for the satisfiability of so-called *pairing-product equations* (PPEs, formally defined below). Many useful languages can be described as sets of PPEs. In particular, when we instantiate our generic BBA construction with the "right" building blocks (namely, compatible commitment and signature schemes), then we can use the very efficient Groth-Sahai proof system for PPEs to efficiently instantiate our generic BBA construction.

In the sequel let $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_T$ be groups of prime order p with pairing $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{G}_T$ and fixed generators g_1, g_2, g_T defining our implicit notation. We consider the special case of the SXDH-based instantiation of GS-proofs from [21].

Definition 4.2. *A* pairing product equation (PPE) *has the form*

$$[\mathbf{a}]_1 \cdot [\mathbf{y}]_2 + [\mathbf{x}]_1 \cdot [\mathbf{b}]_2 + [\mathbf{x}]_1 \cdot \mathbf{\Gamma}\, [\mathbf{y}]_2 = [t]_T \qquad (1)$$

where $[\mathbf{a}]_1 \in \mathcal{G}_1^n$*,* $[\mathbf{b}]_2 \in \mathcal{G}_2^m$*,* $\mathbf{\Gamma} \in \mathbb{Z}_p^{m \times n}$*, and* $[t]_T \in \mathcal{G}_T$ *are constants, and* $[\mathbf{x}]_1 \in \mathcal{G}_1^m$ *and* $[\mathbf{y}]_2 \in \mathcal{G}_2^n$ *are variables.*

As explained in [21], GS-proofs are zero knowledge for pairing product equations with $[t]_T = [0]_T$. Moreover, GS-proofs can be set up in a "witness-extractable" mode, so that they form efficient proofs-of-knowledge. Thus, they possess all required properties from Definition A.3. In the SXDH-based setting, proofs consist of two group elements for each variable, plus eight group elements for each pairing product equation.

Structure-Preserving Signatures. We finally need a signature scheme which is "compatible" with Groth-Sahai proofs to instantiate our construction. We need a signature scheme, where public keys, messages, and signatures are group elements, and verification corresponds to checking whether a given set of PPEs is satisfied. Such signature schemes are called "structure-preserving" [5].

We use the optimal structure-preserving signature scheme of Abe *et al.* [6], which currently is the most efficient known structure-preserving signature scheme (and shown to be optimal in [6]), to instantiate the signature scheme in the generic construction from Section 3.2. The scheme of Abe *et al.* has a security proof in the generic group model [28], which we consider as reasonable to find an as-efficient-as-possible instantiation. Alternatively, we could have used the currently most-efficient standard model scheme of [24], which, however, is significantly less efficient than [6].

Key generation. The key generation algorithm $\mathsf{Gen}_{\mathsf{sig}}$ takes as input group parameters $\Pi_{\mathcal{G}} = (p, \mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_T, e, [1]_1, [1]_2)$ (which are assumed to be generated according to a given security parameter 1^k) and integers $\mu, \nu \in \mathbb{N}_0$ defining the message space of the signature scheme as $\mathcal{G}_1^\nu \times \mathcal{G}_2^\mu$. It chooses random vectors $\mathbf{g} \leftarrow \mathbb{Z}_p^\mu$ and $\mathbf{h} \leftarrow \mathbb{Z}_p^\nu$ and integers $\lambda, \lambda' \leftarrow \mathbb{Z}_p$. The public verification key pk is defined as

$$pk := ([\mathbf{g}]_1, [\mathbf{h}, \lambda, \lambda']_2) \in \mathcal{G}_1^\mu \times \mathcal{G}_2^{\nu+2}$$

and the corresponding secret key is sk as

$$sk := (\mathbf{g}, \mathbf{h}, \lambda, \lambda') \in \mathbb{Z}_p^{\mu+\nu+2}$$

Signing. Given a secret key $sk = (\mathbf{g}, \mathbf{h}, \lambda, \lambda')$ and a message $([\mathbf{m}]_1, [\mathbf{n}]_2) \in \mathcal{G}_1^\nu \times \mathcal{G}_2^\mu$, the signing algorithm Sign chooses $r \leftarrow \mathbb{Z}_p$ computes $\sigma_1 = [r]_1$ and

$$\sigma_2 = \left[\lambda' - r\lambda - \mathbf{m}^\top \mathbf{h}\right]_1, \quad \sigma_3 = \left[(1 - \mathbf{n}^\top \mathbf{g})/r\right]_2$$

and returns $\sigma = (\sigma_1, \sigma_2, \sigma_3)$.

Verification. The verification algorithm $\mathsf{Ver}_{\mathsf{sig}}$ is given a verification key $pk = ([\mathbf{g}]_1, [\mathbf{h}, \lambda, \lambda']_2)$ and a purported signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ as input, and returns 1 if and only if

$$\sigma_1 \cdot [\lambda]_2 + \sigma_2 \cdot [1]_2 + \left[\mathbf{m}^\top\right]_1 \cdot [\mathbf{h}]_2 = \left[\lambda'\right]_T$$

and

$$\sigma_1 \cdot \sigma_3 + \left[\mathbf{g}^\top\right]_1 \cdot [\mathbf{n}]_2 = [1]_T$$

Remark 1. We will instantiate this scheme later with $\nu = 4$ and $\mu = 0$, because we will only sign 4-tuples of elements of $\mathcal{G}_1$. In this case we have $\sigma_3 = 1/r$.

Theorem 4.3 ([6])**.** *The above signature scheme is (strongly) EUF-CMA secure in the generic group model.*

4.2 Instantiation

We instantiate our generic scheme from Section 3.2 with the commitment scheme from Section 4.1, where commitments to s and v live in $\mathcal{G}_1$, and the structure-preserving signature scheme from Section 4.1 with message space $\mathcal{G}_1^4$ (that is, we instantiate the more general construction from Section 4.1 with $\nu := 4$ and $\mu := 0$). It remains to construct a NIZK-PoK for language $L^{\mathsf{pok}}_{(CRS_{\mathsf{com}},pk)}$ defined in Section 3.2.

Our NIZK-PoK consists of two components:

1. A GS-based ZK-PoK of commitments $c_1 = [\mathbf{c}_1]_1 \in \mathcal{G}_1^2$ and $c_2 = [\mathbf{c}_2]_1 \in \mathcal{G}_1^2$ which commit to the same values as the commitments $c_1' = \left[\mathbf{c}_1'\right]_1 \in \mathcal{G}_1^2$ and $c_2' = \left[\mathbf{c}_2'\right]_1 \in \mathcal{G}_1^2$ from the proof statement, respectively.
2. A GS-based ZK-PoK of a signature σ over $(c_1, c_2) = ([\mathbf{c}_1]_1, [\mathbf{c}_2]_1) \in \mathcal{G}_1^4$.

First component. We first show how to prove knowledge of a commitment $[\mathbf{c}]_1 = \mathsf{Com}(CRS_{\mathsf{com}}, s, \mathbf{r}) = [\binom{0}{s} + \mathbf{U}\mathbf{r}]_1 \in \mathcal{G}_1^2$, which commits to the same value as a public commitment $\left[\mathbf{c}'\right]_1 = \left[\binom{0}{s} + \mathbf{U}\mathbf{r}'\right]_1 \in \mathcal{G}_1^2$, using $\mathbf{r}$ and $\mathbf{r}'$ as witnesses.

Since the commitment scheme is additively homomorphic, we can achieve this by proving that $[\mathbf{c}]_1 - \left[\mathbf{c}'\right]_1$ is a commitment to zero. That is, we prove that

$$\begin{aligned} [\mathbf{c}]_1 - \left[\mathbf{c}'\right]_1 = \left[\mathbf{U}\mathbf{r}''\right]_1 &\iff \\ [\mathbf{c}]_1 - \left[\mathbf{c}'\right]_1 - \left[\mathbf{U}\mathbf{r}''\right]_1 = [\mathbf{0}]_1 &\iff \\ \left[\mathbf{c} - \mathbf{c}'\right]_1 \cdot [\mathbf{1}]_2 + [\mathbf{U}]_1 \cdot \left[\mathbf{r}''\right]_2 = [\mathbf{0}]_T & \qquad (2) \end{aligned}$$

Note that (2) consists of two pairing-product equations (PPEs), where $[\mathbf{U}]_1$ (given by the CRS of the commitment scheme) and $[\mathbf{1}]_2$ are constants and $\left[\mathbf{r}''\right]_2 = \left[(\mathbf{r} - \mathbf{r}')\right]_2 \in \mathbb{Z}_p^2$ and $\left[\mathbf{c} - \mathbf{c}'\right]_1$ are variables.

Since we have to prove the above for two commitments, namely $\left[\mathbf{c}_1'\right]$ and $\left[\mathbf{c}_2'\right]$, in total the first component of the proof consists of

- 4 $\mathcal{G}_1$-elements for the statement $\left[(\mathbf{c}_1', \mathbf{c}_2')^\top\right]_1 \in \mathcal{G}_1^4$
- 8 $\mathcal{G}_1$-elements for the component-wise commitments to variables $\left[(\mathbf{c}_1, \mathbf{c}_2)^\top\right]_1 \in \mathcal{G}_1^4$.
- 4 $\mathcal{G}_2$-elements containing the component-wise commitment to variables $\left[\mathbf{r}''\right]_2$
- 16 $\mathcal{G}_1$-elements plus 16 $\mathcal{G}_2$-elements for the proofs for two PPEs for each of the two commitments.

Second component. It remains to prove knowledge of a signature $\sigma = (\sigma_1, \sigma_2, \sigma_3) \in \mathcal{G}_1^2 \times \mathcal{G}_2$ over $\left[(\mathbf{c}_1, \mathbf{c}_2)^\top\right]_1 \in \mathcal{G}_1^4$ that verifies under pk. Note that the verification equation of the signature scheme from Section 4.1 consists of two PPEs. The first PPE is

$$\sigma_1 \cdot [\lambda]_2 + \sigma_2 \cdot [1]_2 + \left[\mathbf{m}^\top\right]_1 \cdot [\mathbf{h}]_2 - [1]_1 \cdot \left[\lambda'\right]_2 = [0]_T$$

where σ_1 and σ_2 and $\mathbf{m} = \left[(\mathbf{c}_1, \mathbf{c}_2)^\top\right]_1 \in \mathcal{G}_1^4$ are variables and all other values are constants. The second PPE is

$$\sigma_1 \cdot \sigma_3 - [1]_1 \cdot [1]_2 = [0]_T$$

where $\sigma_1 \in \mathcal{G}_1$ and $\sigma_3 \in \mathcal{G}_2$ are variables.

Since a commitment to $\left[(\mathbf{c}_1, \mathbf{c}_2)^\top\right]_1$ is already contained in the first part of the proof, we have only two additional variables $\sigma_1, \sigma_2 \in \mathcal{G}_1$ and one additional variable $\sigma_3 \in \mathcal{G}_2$. This contributes the following additional group elements to the proof:

- 4 $\mathcal{G}_1$-elements for commitments to σ_1 and σ_2,
- 2 $\mathcal{G}_2$-elements for the commitment to σ_3, and
- 8 $\mathcal{G}_1$-elements plus 8 $\mathcal{G}_2$-elements for the proofs for two PPEs.

4.3 Efficiency Analysis

Finally, let us analyze the efficiency of our construction. We consider pairing-friendly Barreto-Naehrig (BN) curves [8] defined over a 256-bit field, which corresponds to "128-bit security". We assume for simplicity that each element of $\mathcal{G}_1$ can be represented by 256 bits, and each element of $\mathcal{G}_2$ by 512 bits. We base the analysis of the computational efficiency on mobile devices of our scheme on the work of Grewald *et al.* [20], who presented a software implementation of bilinear pairings with BN curves, and analyzed this implementation on various mobile devices, including a Samsung Galaxy Nexus smartphone with TI OMAP 4460 Cortex-A9 CPU (1.2 GHz)and an Apple iPad 2 with Apple A5 Cortex-A9 CPU (1 GHz). Grewald *et al.* [20] report that a pairing computation in this setting takes below 15 ms on the aforementioned devices. Depending on the implementation (C or assembler) and the representation of elliptic curve points (affine or projective coordinates), even below 10 ms are achievable, however, in the sequel we will calculate with the "worst case" 15 ms. A full exponentiation costs below 4 ms in $\mathcal{G}_1$ and 6 ms in $\mathcal{G}_2$ on the aforementioned devices.

Analysis of User-algorithms. Let us first consider the user algorithms of our scheme. We envision that in typical application scenarios these algorithms will be implemented as smartphone apps that run on hardware comparable to the aforementioned devices considered by Grewald *et al.* [20] and its successors. In order to be use-

able in practice, we consider an expected running time below 0.5 seconds of each single algorithm as acceptable.

CheckTokVal: The running time of this algorithm is dominated by the computation of 2 commitments (which requires to compute 4 exponentiations in $\mathcal{G}_1$) and the verification of one digital signature over four group elements, which takes 7 pairing computations. In total, we expect these two steps to take about 121 ms.

The output of this algorithm is a single bit.

Blind: This algorithm is dominated by the computation of the GS-proof, which takes 24 exponentiations in $\mathcal{G}_1$ and 20 exponentiations in $\mathcal{G}_2$ to compute the first component of the proof, plus 12 $\mathcal{G}_1$-exponentiations and 10 $\mathcal{G}_2$-exponentiations for the second component. This sums up to about 324 ms.

The output of this algorithms is a blinded token (two commitments and a proof), which consist of 44 $\mathcal{G}_1$-elements and 22 $\mathcal{G}_2$-elements (total size $\approx$ 23 kb), and two tuples of $\mathbb{Z}_p$-elements (1 kb).

Unblind: This procedure is dominated by running algorithm CheckTokVal.

The unblinded token consists of two commitments (four $\mathcal{G}_1$-elements), one digital signature (two $\mathcal{G}_1$-elements and one $\mathcal{G}_2$-element), and two tuples $(t_1, t_2) \in \mathbb{Z}_p^2 \times \mathbb{Z}_p^2$. Thus, the total size is about 3 kb.

PrepVer: This procedure is dominated by running algorithm Blind.

The size of a publicly verifiable token, which consists of two commitments, a proof (44 $\mathcal{G}_1$-elements and 22 $\mathcal{G}_2$-elements), and two tuples of $\mathbb{Z}_p$-elements, is about 24 kb.

Analysis of Issuer/Accumulator-algorithms. Now let us consider the algorithms for issuing and accumulation of tokens. Even though these algorithms may in practice be executed on more powerful machines, we use the same hardware platform as for the user algorithms for simplicity. Moreover, for some application scenarios it may even be desirable to use relatively inexpensive devices here as well. Again, we consider an expected running time below 0.5 seconds as acceptable in practice.

Issue: The running time of this algorithm is dominated by the computation of two commitments (each corresponding to 4 exponentiations) in $\mathcal{G}_1$ and the computation of one signature (5 exponentiations in $\mathcal{G}_1$), which in total cost about 36 ms.

The output of Issue is a token $\tau := (c_1, c_2, \sigma, t_1, t_2)$ that consists of two commitments (c_1, c_2) (4 $\mathcal{G}_1$-elements), one signature (2 $\mathcal{G}_1$-elements plus one $\mathcal{G}_2$-element) and two tuples $(t_1, t_2) \in \mathbb{Z}_p^2 \times \mathbb{Z}_p^2$. Thus, the total size of a token is about 3 kb.

Acc: This algorithm computes one commitment (2 exp.) and one signature (5 exp.) in $\mathcal{G}_1$, which takes about 28 ms. The most expensive step here is the verification of the Groth-Sahai proof, which takes 16 pairing evaluations that cost about 240 ms. Thus, in total we expect a running time of this algorithm of about 268 ms.

The output of this algorithm is a token, which consists of two commitments (four group elements of $\mathcal{G}_1$) and one digital signature (two $\mathcal{G}_1$-elements and one $\mathcal{G}_2$-element), which sums up to 2 kb.

Ver: This algorithm computes two commitments in $\mathcal{G}_1$ (four exponentiations in $\mathcal{G}_1$) and verifies one GS-proof (16 pairing evaluations), which in total takes about 256 ms.

The output of this algorithm is a single bit.

Analysis of algorithms for setup and key generation. Finally, let us consider the algorithms for parameter and key generation. Since these algorithms are executed only once, we could in principe accept a much longer running time here (say, several minutes). However, this is not necessary for our construction.

Essentially, the Setup algorithm generates commitment keys for $\mathcal{G}_1$ and $\mathcal{G}_2$. Generating a commitment key for $\mathcal{G}_1$ takes three exponentiations in $\mathcal{G}_1$ (about 12 ms), the key consists of four $\mathcal{G}_1$-elements (1 kb). Generating a commitment key for $\mathcal{G}_2$ takes three exponentiations in $\mathcal{G}_2$ (about 18 ms), the key contains four $\mathcal{G}_2$-elements of size 2 kb. Thus, the total size of the parameters is 3 kb (not counting the description of the BN curve).

The key generation algorithm Gen essentially generates a key pair for the digital signature scheme, which takes $\nu + 2 = 6$ exponentiations in $\mathcal{G}_2$ (about 36 ms). A public key consists of 6 $\mathcal{G}_2$-elements (3 kb), a secret key of 6 $\mathbb{Z}_p$-elements ($\approx$ 1.5 kb).

5 Conclusion

Summary of results. We formalize the functionality, security and privacy properties of a point collection mechanism, an important building block implicitly used in numerous applications. We provide a generic construction for this building block along with a full proof

for its security and privacy. The efficiency analysis shows that one may expect a reasonable efficiency of our instantiation in smartphone-based application scenarios. Even though our estimates do not include the time to perform minor operations (like random sampling), our upper bound of 0.5 seconds on the running time of each algorithm appears achievable for "128-bits of security" in practice. The size of the data to be transmitted during accumulation and verification also seems reasonable for a typical communication protocol in real-world applications, like Near Field Communication (achieving a data rate of up to 424 kB/s).

Future work. The computations that need to be performed on the user's side are too costly for a smartcard-based implementation, which may be desirable in some scenarios. We leave a provably-secure construction for such applications with extreme resource constraints as an open problem. Also, an experimental reference implementation of our scheme in form of a smartphone app on state-of-the-art hardware is left for future work. The main focus of this paper lies on the first foundational formal definition and provably-secure construction of BBAs.

References

[1] The German Big Brother Award. https://bigbrotherawards.de/en.

[2] The Nectar loyalty program. https://www.nectar.com/, 2015.

[3] The Payback loyalty program. https://www.payback.de/, 2015.

[4] The Coop Supercard loyalty program. https://www.coop.ch/pb/site/supercard/node/80441723/Lde/index.html?secure=true, 2015.

[5] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In *Proceedings of CRYPTO 2010*, number 6223 in Lecture Notes in Computer Science, pages 209–236. Springer, 2010.

[6] M. Abe, J. Groth, K. Haralambiev, and M. Ohkubo. Optimal structure-preserving signatures in asymmetric bilinear groups. In *Proceedings of CRYPTO 2011*, number 6841 in Lecture Notes in Computer Science, pages 649–666. Springer, 2011.

[7] M. H. Au, W. Susilo, and Y. Mu. Constant-size dynamic k-taa. *IACR Cryptology ePrint Archive*, 2008:136, 2008.

[8] P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In *Proceedings of Selected Areas in Cryptography 2005*, number 3897 in Lecture Notes in Computer Science, pages 319–331. Springer, 2005.

[9] D. Boneh and X. Boyen. Short signatures without random oracles. In , *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 2004.

[10] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, volume 2248 of *LNCS*, pages 514–532. Springer, 2001.

[11] J. Camenisch, R. Chaabouni, and A. Shelat. Efficient protocols for set membership and range proofs. In , *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*, volume 5350 of *Lecture Notes in Computer Science*, pages 234–252. Springer, 2008.

[12] J. Camenisch, M. Dubovitskaya, and G. Neven. Unlinkable priced oblivious transfer with rechargeable wallets. In , *Financial Cryptography and Data Security, 14th International Conference, FC 2010, Tenerife, Canary Islands, January 25-28, 2010, Revised Selected Papers*, volume 6052 of *Lecture Notes in Computer Science*, pages 66–81. Springer, 2010.

[13] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick. A survey on privacy in mobile participatory sensing applications. *Journal of Systems and Software*, 84(11):1928–1946, 2011.

[14] S. E. Coull, M. Green, and S. Hohenberger. Controlling access to an oblivious database using stateful anonymous credentials. In , *Public Key Cryptography - PKC 2009, 12th International Conference on Practice and Theory in Public Key Cryptography, Irvine, CA, USA, March 18-20, 2009. Proceedings*, volume 5443 of *Lecture Notes in Computer Science*, pages 501–520. Springer, 2009.

[15] P. Dutta, P. M. Aoki, N. Kumar, A. M. Mainwaring, C. Myers, W. Willett, and A. Woodruff. Common sense: participatory urban sensing using a network of handheld air quality monitors. In *SenSys*, pages 349–350. ACM, 2009.

[16] M. Enzmann and M. Schneider. A privacy-friendly loyalty system for electronic marketplaces. In *2004 IEEE International Conference on e-Technology, e-Commerce, and e-Services (EEE 04), 29-31 March 2004, Taipei, Taiwan*, pages 385–393. IEEE Computer Society, 2004.

[17] A. Escala, G. Herold, E. Kiltz, C. Ràfols, and J. L. Villar. An algebraic framework for Diffie-Hellman assumptions. In *Proceedings of CRYPTO (2) 2013*, number 8043 in Lecture Notes in Computer Science, pages 129–147. Springer, 2013.

[18] *Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance - 9th International Workshop, DPM 2014, 7th International Workshop, SETOP 2014, and 3rd International Workshop, QASA 2014, Wroclaw, Poland, September 10-11, 2014. Revised Selected Papers*, volume 8872 of *Lecture Notes in Computer Science*, 2015. Springer.

[19] Y. Gong, Y. Cai, Y. Guo, and Y. Fang. A privacy-preserving scheme for incentive-based demand response in the smart grid. *IEEE Transactions on Smart Grid*, 2015.

[20] G. Grewal, R. Azarderakhsh, P. Longa, S. Hu, and D. Jao. Efficient implementation of bilinear pairings on ARM processors. In *Proceedings of Selected Areas in Cryptography 2012*, number 7707 in Lecture Notes in Computer Science, pages 149–165. Springer, 2012.

[21] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Proceedings of EUROCRYPT*

2008, number 4965 in Lecture Notes in Computer Science, pages 415–432. Springer, 2008.

[22] R. Johnson, D. Molnar, D. X. Song, and D. Wagner. Homomorphic signature schemes. In *Proceedings of CT-RSA 2002*, number 2271 in Lecture Notes in Computer Science, pages 244–262. Springer, 2002.

[23] W. Kempton and J. Tomic. Vehicle-to-grid power fundamentals: Calculating capacity and net revenue. *Elsevier Journal of Power Sources*, 144(1):268–279, 2005.

[24] E. Kiltz, J. Pan, and H. Wee. Structure-preserving signatures from standard assumptions, revisited. In , *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 275–295. Springer, 2015.

[25] Q. Li and G. Cao. Providing efficient privacy-aware incentives for mobile sensing. In *IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014, Madrid, Spain, June 30 - July 3, 2014*, pages 208–217. IEEE Computer Society, 2014.

[26] M. Milutinovic, I. Dacosta, A. Put, and B. De Decker. ucentive: An efficient, anonymous and unlinkable incentives scheme. In *IEEE TrustCom-15*, 2015.

[27] A. Rupp, F. Baldimtsi, G. Hinterwälder, and C. Paar. Cryptographic theory meets practice: Efficient and privacy-preserving payments for public transport. *ACM Trans. Inf. Syst. Secur.*, 17(3):10:1–10:31, 2015.

[28] V. Shoup. Lower bounds for discrete logarithms and related problems. In *Proceedings of EUROCRYPT 1997*, number 1233 in Lecture Notes in Computer Science, pages 256–266. Springer, 1997.

[29] Z. Yang, S. Yu, W. Lou, and C. Liu. P^2: Privacy-preserving communication and precise reward architecture for v2g networks in smart grid. *IEEE Trans. Smart Grid*, 2(4):697–706, 2011.

A Formal Definitions of Building Blocks

In the following we provide formal and standard definitions for homomorphic commitments, signatures, and NIZK-PoKs.

Definition A.1. *A* commitment scheme $(\mathsf{Setup}_{\mathsf{com}}, \mathsf{Com})$ *consists of two algorithms.*

- $\mathsf{Setup}_{\mathsf{com}}$ *is a PPT algorithm, which takes as input a security parameter* 1^k *and outputs parameters* CRS_{com}.
- Com *is a deterministic polynomial-time algorithm, which takes as input parameters* CRS_{com}, *message* m, *and randomness* r, *and outputs a commitment* $c = \mathsf{Com}(CRS_{\mathsf{com}}, m, r)$ *to* m.

Note that given $(CRS_{\mathsf{com}}, c, m, r)$ *where* $c = \mathsf{Com}(CRS_{\mathsf{com}}, m, r)$, *one can efficiently verify that* c *is a commitment to* m *by checking that* $c = \mathsf{Com}(CRS_{\mathsf{com}}, m, r)$.

We say that a commitment scheme $(\mathsf{Setup}_{\mathsf{com}}, \mathsf{Com})$ *is* additively homomorphic, *if there exists an additional PPT algorithm* CAdd *which takes as input two commitments* $c = \mathsf{Com}(CRS_{\mathsf{com}}, m, r)$ *and* $c' = \mathsf{Com}(CRS_{\mathsf{com}}, m', r')$, *and outputs a commitment* c'' *such that* $c'' = \mathsf{Com}(CRS_{\mathsf{com}}, m + m', r + r')$.

We say that a commitment scheme $(\mathsf{Setup}_{\mathsf{com}}, \mathsf{Com})$ *is* secure, *if it has the following two properties.*

1. ***Binding:*** *For all probabilistic polynomial-time adversaries* $\mathcal{A}$ *holds that the advantage* $\mathsf{Adv}^{\mathsf{bind}}_{\mathcal{A}}(1^k)$ *defined by*

$$\Pr\left[\begin{array}{c} c = \mathsf{Com}(CRS_{\mathsf{com}}, m, r) \\ \wedge \\ c = \mathsf{Com}(CRS_{\mathsf{com}}, m', r') \end{array} \middle| \begin{array}{c} \Pi \leftarrow \mathsf{Setup}_{\mathsf{com}}(1^k) \\ (c, m, r, m', r') \leftarrow \mathcal{A}(1^k, \Pi) \\ (m', r') \neq (m, r) \end{array}\right]$$

   is negligible in k. *We say that the commitment scheme is* perfectly *binding if this advantage is always* 0.

2. ***Hiding:*** *For all probabilistic polynomial-time adversaries* $\mathcal{A}$ *holds that the advantage* $\mathsf{Adv}^{\mathsf{hide}}_{\mathcal{A}}(1^k)$ *defined by*

$$\left|\Pr\left[b = b' \middle| \begin{array}{c} \Pi \leftarrow \mathsf{Setup}_{\mathsf{com}}(1^k) \\ (m_0, m_1, state) \leftarrow \mathcal{A}(1^k, CRS_{\mathsf{com}}) \\ b \leftarrow \{0,1\};\ r \leftarrow \{0,1\}^k \\ c := \mathsf{Com}(CRS_{\mathsf{com}}, m_b, r) \\ b' \leftarrow \mathcal{A}(c, state) \end{array}\right] - \frac{1}{2}\right|$$

   is negligible in k. *We say that the commitment scheme is* perfectly *hiding if this advantage is always* 0.

Definition A.2. *A* digital signature scheme $(\mathsf{Gen}_{\mathsf{sig}}, \mathsf{Sign}, \mathsf{Ver}_{\mathsf{sig}})$ *consists of three PPT algorithms.*

- $\mathsf{Gen}_{\mathsf{sig}}$ *takes as input a security parameter* 1^k *and outputs a key pair* (pk, sk).
- Sign *takes as input the secret key* sk *and a message* m, *and outputs a signature* $\sigma \leftarrow \mathsf{Sign}(sk, m)$.
- $\mathsf{Ver}_{\mathsf{sig}}$ *takes as input the public key* pk, *a message* m, *and a purported signature* σ, *and outputs a bit* $\mathsf{Ver}_{\mathsf{sig}}(pk, \sigma, m) \in \{0, 1\}$.

We say that a digital signature scheme is EUF-CMA secure if for all PPT adversaries $\mathcal{A}$ *holds that the advantage* $\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathcal{A}}(1^k)$ *defined by*

$$\Pr\left[\begin{array}{c} \mathsf{Ver}_{\mathsf{sig}}(pk, \sigma^*, m^*) = 1 \\ \wedge \\ m^* \notin \{m_1, \ldots, m_q\} \end{array} \middle| \begin{array}{c} (pk, sk) \leftarrow \mathsf{Gen}_{\mathsf{sig}}(1^k) \\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(sk, \cdot)}(1^k, pk) \end{array}\right]$$

is negligible in k, *where* $\mathsf{Sign}(sk, \cdot)$ *is an oracle that, on input* m, *returns* $\mathsf{Sign}(sk, m)$, *and* $\{m_1, \ldots, m_q\}$ *denotes the set of messages queried by* $\mathcal{A}$ *to its oracle.*

Definition A.3. *Let R be a witness relation for some NP language L_R. Let $\mathsf{POK} := (\mathsf{Setup}_{\mathsf{pok}}, \mathsf{Prove}, \mathsf{Ver}_{\mathsf{pok}})$ be a tuple of PPT algorithms such that*

- $\mathsf{Setup}_{\mathsf{pok}}$ *takes as input a security parameter 1^k and outputs public parameters CRS.*
- Prove *takes as input the parameters CRS, a witness x, and a statement z with $(x, z) \in R$ and outputs a proof π.*
- $\mathsf{Ver}_{\mathsf{pok}}$ *takes as input the parameters CRS, a statement z, and a proof π and outputs 1 or 0.*

POK *is called a non-interactive zero-knowledge proof of knowledge (NIZK-PoK) system for R if completeness, extractability, and zero-knowledge as defined below are satisfied.*

1. ***Completeness:*** *For all $CRS \leftarrow \mathsf{Setup}_{\mathsf{pok}}(1^k)$ and $(x, z) \in R$, $\mathsf{Ver}_{\mathsf{pok}}(CRS, x, \pi) = 1$ for all proofs $\pi \leftarrow \mathsf{Prove}(CRS, x, z)$.*
2. ***Extractability (Soundness):*** *There exists a polynomial-time extractor algorithm $E = (E_1, E_2)$ such that for all PPT adversaries $\mathcal{A}$*
   (a) the advantage $\mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{A}}(k)$ defined by

$$\left| \begin{array}{rl} & \Pr[1 \leftarrow \mathcal{A}(CRS) \mid CRS \leftarrow \mathsf{Setup}_{\mathsf{pok}}(1^k)] \\ - & \Pr[1 \leftarrow \mathcal{A}(CRS_{\mathsf{e}}) \mid (CRS_{\mathsf{e}}, \zeta_{\mathsf{e}}) \leftarrow E_1(1^k)] \end{array} \right|$$

   is negligible.
   (b) the advantage $\mathsf{Adv}^{\mathsf{pok\text{-}ext}}_{\mathsf{POK},\mathcal{A}}(k)$ defined by

$$\Pr\left[\begin{array}{c} \mathsf{Ver}(CRS_{\mathsf{e}}, x, \pi) = 1 \\ \wedge \\ (x, z) \notin R \end{array} \middle| \begin{array}{c} z \leftarrow E_2(CRS_{\mathsf{e}}, \zeta_{\mathsf{e}}, x, \pi) \\ (x, \pi) \leftarrow \mathcal{A}(CRS_{\mathsf{e}}) \\ (CRS_{\mathsf{e}}, \zeta_{\mathsf{e}}) \leftarrow E_1(1^k) \end{array} \right]$$

   is negligible.

   Perfect extractability is achieved if both advantages are 0.
3. ***Zero-knowledge:*** *There exists a polynomial-time simulator algorithm $S = (S_1, S_2)$ such that for all PPT adversaries $\mathcal{A}$ holds that the advantage $\mathsf{Adv}^{\mathsf{zk}}_{\mathsf{POK},\mathcal{A}}(k)$ defined by*

$$\left| \begin{array}{rl} & \Pr[1 \leftarrow \mathcal{A}^{S_2'(CRS_{\mathsf{s}}, \zeta_{\mathsf{s}}, \cdot, \cdot)}(1^k, CRS_{\mathsf{s}})] \\ - & \Pr[1 \leftarrow \mathcal{A}^{\mathsf{Prove}(CRS, \cdot, \cdot)}(1^k, CRS)] \end{array} \right|$$

   is negligible, where $(CRS_{\mathsf{s}}, \zeta_{\mathsf{s}}) \leftarrow S_1(1^k)$, $CRS \leftarrow \mathsf{Setup}_{\mathsf{pok}}(1^k)$, and $S_2'(CRS_{\mathsf{s}}, \zeta_{\mathsf{s}}, \cdot, \cdot)$ is an oracle which on input $(x, z) \in R$, returns $S_2(CRS_{\mathsf{s}}, \zeta_{\mathsf{s}}, x)$. Both S_2' and Prove return $\perp$ on input $(x, z) \notin R$. Perfect zero-knowledge is achieved if the advantage is zero.

B Proof of Security

In the following we prove Theorem 3.1 from Section 3.2.

We prove this theorem by showing that for any PPT adversary $\mathcal{A}$ against the security of our BBA scheme there exist PPT adversaries $\mathcal{B}, \mathcal{C}$ against the extractability of the NIZK-PoK and $\mathcal{D}$ against the security of the signature scheme such that

$$\begin{array}{rcl} \mathsf{Adv}^{\mathsf{acc\text{-}sec}}_{\mathsf{BBA},\mathcal{A}}(k) & \leq & \mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) \\ & + & \mathsf{Adv}^{\mathsf{pok\text{-}ext}}_{\mathsf{POK},\mathcal{C}}(k) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathsf{SIG},\mathcal{D}}(k) \end{array}$$

We proceed in a sequence of games. We start with the real experiment $\mathsf{Exp}^{\mathsf{acc\text{-}sec}}_{\mathsf{BBA},\mathcal{A}}(k)$, and then gradually modify this experiment, until we reach an experiment where $\mathcal{A}$ has no chance to win. We show that if any two consecutive games would significantly differ, this would also result in effective adversaries against the security of the building blocks.

Let **Game** 1 be the $\mathsf{Exp}^{\mathsf{acc\text{-}sec}}_{\mathsf{BBA},\mathcal{A}}(k)$ experiment, where, in addition, we assume that records are kept of every input, output, and random choice that the challenger or the Issue or Acc oracle gets or makes. Let $\mathsf{Adv}^{\mathrm{Game}_i}_{\mathsf{BBA},\mathcal{A}}(k) = \Pr[\mathsf{Exp}^{\mathrm{Game}_i}_{\mathsf{BBA},\mathcal{A}}(k) = 1]$ denote the advantage of $\mathcal{A}$ in Game i. Thus, by definition,

$$\mathsf{Adv}^{\mathsf{acc\text{-}sec}}_{\mathsf{BBA},\mathcal{A}}(k) = \mathsf{Adv}^{\mathrm{Game}_1}_{\mathsf{BBA},\mathcal{A}}(k). \tag{3}$$

In **Game** 2, we generate the common reference string of POK by running the corresponding extractor setup algorithm which returns $(CRS_{\mathsf{pok}}, \tau_{\mathsf{pok}}) \leftarrow E_1(1^k)$. Clearly, from a distinguisher between Games 2 and 1, we can immediately build an adversary $\mathcal{B}$ against the extractability setup property of POK having advantage

$$\mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) = \left| \mathsf{Adv}^{\mathrm{Game}_1}_{\mathsf{BBA},\mathcal{A}}(k) - \mathsf{Adv}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) \right|. \tag{4}$$

In **Game** 3, we try to extract a witness from all verifying proofs of knowledge contained in the inputs $\{\tau\}$ to the Acc oracle as well as in the final output $(\rho_1^*, s_1^*, w_1^*), \ldots (\rho_\ell^*, s_\ell^*, w_\ell^*)$ of $\mathcal{A}$, where each ρ_i contains such a proof. For a given $\{\tau\} = (c_1', c_2', \pi)$ or $\rho := (c_1', c_2', r_1, r_2, \pi)$ with $\mathsf{Ver}_{\mathsf{POK}}(CRS_{\mathsf{pok}}, \pi, (c_1', c_2')) = 1$, we run the extractor $E_2(CRS_{\mathsf{pok}}, \tau_{\mathsf{pok}}, \pi, (c_1', c_2'))$ which returns a witness $z = (c_1, c_2, \sigma, \phi_1(t_1'), \phi_2(t_2'))$. If $R((c_1', c_2'), z) = 0$ we say that the adversary loses and let the experiment return 0. We call this event failure event $\mathfrak{F}_1$. Note that we can easily build an adversary $\mathcal{C}$ against the second extractability property of POK with advantage

$$\mathsf{Adv}^{\mathsf{pok\text{-}ext}}_{\mathsf{POK},\mathcal{C}}(k) = \Pr[\mathfrak{F}_1]. \tag{5}$$

Moreover, we have

$$\mathsf{Adv}^{\mathrm{Game}_3}_{\mathsf{BBA},\mathcal{A}}(k) = \Pr[\mathsf{Exp}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) = 1 \wedge \neg\mathfrak{F}_1] \quad (6)$$

In **Game** 4, for each valid witness $z = (c_1, c_2, \sigma, \phi_1(t_1'), \phi_2(t_2'))$ we extract, we check the previous output records of the Issue and Acc oracle if there is a match for both c_1 and c_2 (within the same record). If there is no match, we say the adversary loses and let the experiment return 0. We call this event failure event $\mathfrak{F}_2$. In this case we know that σ is a valid signature for a new message $m := c_1||c_2$, the adversary created on its own. Hence, we can construct an EUF-CMA adversary $\mathcal{D}$ against SIG with advantage

$$\mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathsf{SIG},\mathcal{D}}(k) = \Pr[\mathfrak{F}_2]. \quad (7)$$

Moreover, we have

$$\mathsf{Adv}^{\mathrm{Game}_4}_{\mathsf{BBA},\mathcal{A}}(k) = \Pr[\mathsf{Exp}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) = 1 \wedge \neg(\mathfrak{F}_1 \vee \mathfrak{F}_2)] \quad (8)$$

Game 5 is a transitional game, where we do not actually modify the experiment but introduce some additional bookkeeping. We will associate each output of Issue and Acc with a serial number s and amount w. Note that since we assume that our commitment scheme is perfectly binding, there is a unique s and a unique w for each given commitments c_1 and c_2 (output of Issue) or c_1'' and c_2'' (output of Acc), respectively. For outputs of Issue this is easy since s and w are given as input. Assuming m calls to the Issue oracle, we denote the serial numbers given as input to this oracle by $s_1, \ldots, s_m$ and the initial values by $u_1, \ldots, u_m$.

Let us now consider the first call to the Acc oracle and let us assume that none of the failures defined before happened. As there is a valid signature on the extracted values c_1 and c_2, we know that there must be (at least) one match with the output records of the previous Issue calls and thus those values are indeed commitments. If there is more than one match, we know by the perfect binding property that these output records must be associated with the *same* serial number and amount. So for the bookkeeping it does not matter which of the previous records is considered. Let s_i and u_i denote the associated values for the extracted commitments. Then by the soundness of POK and the homomorphic property of COM, we know that c_1' and c_2' are re-randomized commitments to the same values. Thus the output of Acc will be associated to s_i and $w_1 := u_i + v_1$, where v_1 denotes the value that is accumulated in the current call to Acc.

Similarly, during the j-th call to the Acc oracle, we need to consider the output records of all previous calls to the Acc oracle as well as the Issue oracle. Then, again, for the extracted commitments there needs to be a match among the considered records and it does not matter which of these matches we consider. Assuming n calls to the Acc oracle, we denote the values to be accumulated during the calls by $v_1, \ldots, v_n$. Furthermore, let s_i and w, denote the associated values for the extracted commitments. Then the output of j-th call to Acc is associated with s_i and $w_j := w + v_j$. Hence, each w_j associated with s_i is of the form $w_j = u_i + v_{k_1} + \ldots + v_{k_\ell} + v_j$ for some $k_1, \ldots, k_\ell \in \{1, \ldots j-1\}$. There are a few other important observations to make about these accumulation sums w_j: (1) a specific v_t (considered as a variable) with $t \in \{1, \ldots, n\}$ may appear at most once in w_j, i.e., $k_1, \ldots, k_\ell$ are pairwise distinct values. (2) for any two sums w_j and $w_{j'}$ with $j \neq j'$ that are associated with distinct serial numbers $s_i \neq s_i'$, it holds that any variable v_t may not appear in both w_j and $w_{j'}$ as summand. (In other words, any v_t is uniquely associated with some s_i.) (3) the same holds for any variable u_i with $i \in \{1, \ldots, m\}$.

Now, let us consider the final output $(\rho_1^*, s_1^*, w_1^*), \ldots (\rho_\ell^*, s_\ell^*, w_\ell^*)$ of $\mathcal{A}$ assuming that no failure event occurred and the winning condition is satisfied. For any $\rho_i^* = (c_1', c_2', r_1, r_2, \pi)$ (omitting the index i in the tuple for readability), we consider the commitments c_1 and c_2 extracted from π. Now, we know that c_1' and c_2' are re-randomizations of c_1 and c_2, where the latter commitments have occurred previously as output of Issue or Acc. Thus, following our argument from before, also this c_1' and c_2' can be uniquely associated with some s_k and $w := u_k + \sum_{j \in M} v_j$ for some $M \subset \{1, \ldots, n\}$. Moreover, as the commitment scheme is perfectly binding, it holds for the claimed serial number $s_i^* \in \mathcal{S}$ and amount $w_i^* \in \mathcal{V}$ that indeed $s_i^* = s_k$ and $w_i^* = w$ provided that opening the commitments c_1' and c_2' with s_k^* and w_k^* succeeded. In particular, the case $s_i^* \notin \{s_1, \ldots, s_m\}$ cannot occur here.[5] So to satisfy the winning condition it must hold that

$$\sum_{i=1}^{\ell} w_i^* > \sum_{i=1}^{m} u_i + \sum_{i=1}^{n} v_i, \quad (9)$$

where the corresponding s_i^* are pairwise distinct. However, as observed before, each w_i^* is of the form $w_i^* = u_k + \sum_{j \in M} v_j$, where the u_k and v_j's will not occur in any other $w_{i'}^*$ with $i' \neq i$. Thus, Eq. 9 cannot be satisfied.

5 In this case, $\mathcal{A}$ would have forged a signature what we already excluded at this point.

Hence, we have

$$\mathsf{Adv}^{\mathrm{Game}_5}_{\mathsf{BBA},\mathcal{A}}(k) = \mathsf{Adv}^{\mathrm{Game}_4}_{\mathsf{BBA},\mathcal{A}}(k) = 0. \qquad (10)$$

Putting thing together. By considering Equations 3 to 10, we get

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{Game}_1}_{\mathsf{BBA},\mathcal{A}}(k) &\leq \left|\mathsf{Adv}^{\mathrm{Game}_1}_{\mathsf{BBA},\mathcal{A}}(k) - \mathsf{Adv}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k)\right| \\
&+ \mathsf{Adv}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) \\
&= \mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) + \mathsf{Adv}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) \\
&= \mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) \\
&+ \Pr[\mathsf{Exp}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) = 1 \wedge \neg(\mathfrak{F}_1 \vee \mathfrak{F}_2)] \\
&+ \Pr[\mathsf{Exp}^{\mathrm{Game}_2}_{\mathsf{BBA},\mathcal{A}}(k) = 1 \wedge (\mathfrak{F}_1 \vee \mathfrak{F}_2)] \\
&\leq \mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) + \mathsf{Adv}^{\mathrm{Game}_5}_{\mathsf{BBA},\mathcal{A}}(k) \\
&+ \Pr[\mathfrak{F}_1] + \Pr[\mathfrak{F}_2] \\
&= \mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) + \Pr[\mathfrak{F}_1] + \Pr[\mathfrak{F}_2] \\
&= \mathsf{Adv}^{\mathsf{pok\text{-}ext\text{-}setup}}_{\mathsf{POK},\mathcal{B}}(k) \\
&+ \mathsf{Adv}^{\mathsf{pok\text{-}ext}}_{\mathsf{POK},\mathcal{C}}(k) + \mathsf{Adv}^{\mathsf{euf\text{-}cma}}_{\mathsf{SIG},\mathcal{D}}(k)
\end{aligned}$$

As we assume that the three advantages in the last equation are negligible our theorem follows.

C Proof of Privacy

In this section we prove Theorem 3.2 from Section 3.2.

We prove our theorem by showing that there exist PPT algorithms $(\mathsf{SimSetup}, \mathsf{SimBlind}, \mathsf{SimPrepVer})$ such that from each privacy-adversary $\mathcal{A}$ we can construct adversaries $\mathcal{B}$ and $\mathcal{C}$ such that

$$\mathsf{Adv}^{\mathsf{priv}}_{\mathsf{BBA},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathsf{zk}}_{\mathsf{POK},\mathcal{B}}(k) + 2n \cdot \mathsf{Adv}^{\mathsf{hide}}_{\mathcal{A}}(1^k)$$

We proceed in a sequence of games. We start with the real experiment $\mathsf{Exp}^{\mathsf{priv\text{-}real}}_{\mathsf{BBA},\mathcal{A}}$, and then gradually modify this experiment, until we reach an experiment which is identical to $\mathsf{Exp}^{\mathsf{priv\text{-}ideal}}_{\mathsf{BBA},\mathcal{A}}$. The theorem then follows by showing that any PPT adversary has only an at most negligible advantage in distinguishing any two consecutive games. We write Setup_i, Blind_i, and $\mathsf{PrepVer}_i$ do denote the implementations of Setup, Blind, and $\mathsf{PrepVer}$ in Game i. Moreover, we write X_i to denote the event that $\mathcal{A}$ outputs 1 in Game i.

We define **Game** 1 identical to $\mathsf{Exp}^{\mathsf{priv\text{-}real}}_{\mathsf{BBA},\mathcal{A}}$. Thus we have $\mathsf{Setup}_1 = \mathsf{Setup}$, $\mathsf{Blind}_1 = \mathsf{Blind}$, and $\mathsf{PrepVer}_1 = \mathsf{PrepVer}$.

In **Game** 2 we modify algorithm $\mathsf{PrepVer}_1$. Note that $\mathsf{PrepVer}_1$ runs algorithm Blind_1 as a subroutine. Now we change this, by including all these operations directly in $\mathsf{PrepVer}_1$. This is a preparation for our modifications to algorithm Blind_1 in subsequent games.

More precisely, we define $\mathsf{Setup}_2 := \mathsf{Setup}_1$ and $\mathsf{Blind}_2 := \mathsf{Blind}_1$, and we let $\mathsf{PrepVer}_2$ be the following algorithm.

$\mathsf{PrepVer}_2(pk, \tau, s, w)$

If $\tau = \bot$ **return** $\bot$
parse $(c_1, c_2, \sigma, r_1, r_2) := \tau$
$t'_1, t'_2 \leftarrow \mathcal{R}$
$c'_1 := \mathsf{CAdd}(c_1, \mathsf{Com}(CRS_{\mathsf{com}}, 0, t'_1))$
$c'_2 := \mathsf{CAdd}(c_2, \mathsf{Com}(CRS_{\mathsf{com}}, 0, t'_2))$
$\pi := \mathsf{Prove}(CRS_{\mathsf{pok}}, (c'_1, c'_2), (c_1, c_2, \sigma, \phi_1(t'_1), \phi_2(t'_2)))$
$\{\tau\} := (c'_1, c'_2, \pi)$, $(r_1, r_2) := (t_1 + t'_1, t_2 + t'_2)$
return $\rho := (c'_1, c'_2, r_1, r_2, \pi)$

This change is purely conceptual, so we have

$$\Pr[X_2] = \Pr[X_1]$$

Game 3 is identical to Game 2, except for the following. Recall that in Setup_2 we compute $CRS = (CRS_{\mathsf{com}}, CRS_{\mathsf{pok}})$ by computing $CRS_{\mathsf{com}} \leftarrow \mathsf{Setup}_{\mathsf{com}}(1^k)$ and $CRS_{\mathsf{pok}} \leftarrow \mathsf{Setup}_{\mathsf{pok}}(1^k)$. In Game 3 we replace Setup_2 with algorithm Setup_3, which works identical, except that it computes CRS_{pok} by running the simulation algorithm S_1 of the NIZK proof system to compute $(CRS_{\mathsf{pok}}, \tau_{\mathsf{sim}}) \leftarrow S_1(1^k)$.

Moreover, recall that in Blind_2 and $\mathsf{PrepVer}_2$ we compute proofs as $\pi \leftarrow \mathsf{Prove}(CRS_{\mathsf{pok}}, (c'_1, c'_2), (c_1, c_2, \sigma, \phi_1(t'_1), \phi_2(t'_2)))$. We replace these algorithms with Blind_3 and $\mathsf{PrepVer}_3$, which instead run the simulation algorithm S_2 of the NIZK proof system to compute simulated proofs as $\pi \leftarrow S_2(\tau_{\mathsf{sim}}, (c'_1, c'_2))$.

We have $|\Pr[X_3] - \Pr[X_2]| = \epsilon_{\mathsf{nizk}}$ for some $\epsilon_{\mathsf{nizk}} \in [0, 1]$. We show that any adversary $\mathcal{A}$ distinguishing Game 3 from Game 2 implies an adversary $\mathcal{B}$ breaking the zero-knowledge property of the NIZK proof system with advantage $\mathsf{Adv}^{\mathsf{zk}}_{\mathsf{POK},\mathcal{B}}(k) = \epsilon_{\mathsf{nizk}}$. Adversary $\mathcal{B}$ works as follows. It runs adversary $\mathcal{A}$ as a subroutine by simulating the security experiment exactly as in Game 2, except for the following two differences. First, $\mathcal{B}$ does not generate a common reference string for the proof of knowledge on its own, but instead uses the CRS CRS_{pok} that it receives from the NIZK security experiment. Second, instead of computing proofs in Blind_2 by running $\pi \leftarrow \mathsf{Prove}(CRS_{\mathsf{pok}}, (c'_1, c'_2), (c_1, c_2, \sigma, \phi_1(t'_1), \phi_2(t'_2)))$, $\mathcal{B}$ obtains the proof by querying its S'_2-oracle on statement (c'_1, c'_2) with corresponding witness $(c_1, c_2, \sigma, \phi_1(t'_1), \phi_2(t'_2))$.

If CRS_{pok} is computed using $\mathsf{Setup}_{\mathsf{pok}}$ and S'_2 internally uses algorithm Prove to compute proofs, then the view provided by $\mathcal{B}$ to $\mathcal{A}$ is identical to Game 2. If CRS_{pok} is computed using S_1 and S'_2 internally uses algorithm S_2 to compute proofs, then the view provided

by $\mathcal{B}$ to $\mathcal{A}$ is identical to Game 3. Thus, we have

$$|\Pr[X_3] - \Pr[X_2]| \leq \mathsf{Adv}^{\mathsf{zk}}_{\mathsf{POK},\mathcal{B}}(k)$$

Game 4 is identical to Game 3, in particular we have $\mathsf{Setup}_4 = \mathsf{Setup}_3$ and $\mathsf{Blind}_4 = \mathsf{Blind}_3$, but we change the way how commitments are computed in $\mathsf{PrepVer}_4$. Recall that the commitments c_1' and c_2' in $\mathsf{PrepVer}_3$ are computed by re-randomizing c_1 and c_2 using the fact that the commitments are additively homomorphic in both the message space and the randomness space. Thus, c_1' and c_2' are uniformly random commitments to the messages committed in c_1 and c_2.

In Game 4 we now replace algorithm $\mathsf{PrepVer}_3$ with algorithm $\mathsf{PrepVer}_4$, which proceeds exactly as $\mathsf{PrepVer}_3$, except that c_1' and c_2' are computed as $c_1' \leftarrow \mathsf{Com}(CRS_{\mathsf{com}}, s, t_1')$ and $c_2' \leftarrow \mathsf{Com}(CRS_{\mathsf{com}}, w, t_2')$ for uniformly random t_1', t_2'. Thus, in $\mathsf{PrepVer}_4$ the commitments c_1' and c_2' are uniformly random commitments to s and w, independent of c_1 and c_2.

Note if $\mathsf{PrepVer}_3$ and $\mathsf{PrepVer}_4$ are called on input a token $\tau \neq \bot$, then it is guaranteed that $\tau = (c_1, c_2, \sigma, r_1, r_2) \neq \bot$ where c_1 is a commitment to s and c_2 is a commitment to w. This is because correctness of the commitments is checked after $\mathcal{A}_0$ has output the first token and after each invocation of $\mathsf{Unblind}$. Thus, we either have $\tau = \bot$, or the commitments contained in τ are commitments to s and w, respectively. Therefore again this modification is purely conceptual, so we have

$$\Pr[X_4] = \Pr[X_3]$$

Note that $\mathsf{PrepVer}_4$ now depends only on (pk, s, w), but not on token τ anymore.

In **Game** 5 we replace all commitments c_1' and c_2' computed in algorithm Blind_4 with commitments to zero, using the hiding property of the commitment scheme and the fact that proofs are simulated. More precisely, recall that the commitments c_1' and c_2' in Blind_4 are computed by re-randomizing c_1 and c_2 using the fact that the commitments are additively homomorphic in both the message space and the randomness space. Thus, c_1' and c_2' are uniformly random commitments to the messages committed in c_1 and c_2.

In Game 5 we now replace algorithm Blind_4 with algorithm Blind_5, which proceeds exactly as Blind_4, except that c_1' and c_2' are computed as $c_1' \leftarrow \mathsf{Com}(CRS_{\mathsf{com}}, 0, t_1')$ and $c_2' \leftarrow \mathsf{Com}(CRS_{\mathsf{com}}, 0, t_2')$ for uniformly random t_1', t_2'. Thus, in Blind_5 the commitments c_1' and c_2' are uniformly random commitments to 0, independent of c_1 and c_2.

We show that any adversary $\mathcal{A}$ distinguishing Game 5 from Game 4 implies an adversary $\mathcal{C}$ breaking the hiding-property of Definition A.1 of the commitment scheme.

Since $\mathcal{A}$ interacts exactly n times with its oracle $\mathcal{U}$, and thus in total $2n$ commitments are replaced with zero-commitments, we use a hybrid argument. In the sequel let experiment H_0 be identical to Game 4. In hybrid H_i we compute the first i commitments as in Blind_5, and the remaining $2n - i$ as in Blind_4. Thus, the first i commitments in H_i are commitments to zero, while the remaining commitments are commitments to s and w, respectively. Clearly, hybrid H_{2n} is identical to Game 4.

Suppose there exists an algorithm $\mathcal{A}$ distinguishing H_{i-1} from H_i for some $i \in \{1, \ldots, 2n\}$. We construct algorithm $\mathcal{C}$ as follows. $\mathcal{C}$ runs $\mathcal{A}$ as a subroutine, by simulating hybrid H_{i-1}, with the following two exceptions. First, instead of generating a commitment key by running $\mathsf{Setup}_{\mathsf{com}}$, $\mathcal{C}$ instead uses the commitment key CRS_{com} that it receives from the commitment security experiment. Second, when the BBA privacy experiment has to compute the i-th commitment to some value γ, $\mathcal{C}$ does not compute this commitment itself. Instead it defines $m_0 := \gamma$ and $m_1 := 0$, outputs (m_0, m_1) to the commitment security experiment, and uses the commitment to message m_b it receives back. If $b = 0$, then $\mathcal{A}$'s view is identical to Game 4, while if $b = 1$ then it is identical to Game 5. Since there are $2n$ hybrid experiments, we have $\epsilon_{\mathsf{hiding}} \leq 2n \cdot \mathsf{Adv}^{\mathsf{hide}}_{\mathcal{C}}(1^k)$ and thus

$$|\Pr[X_5] - \Pr[X_4]| \leq 2n \cdot \mathsf{Adv}^{\mathsf{hide}}_{\mathcal{C}}(1^k)$$

Note that algorithm Blind_5 now depends only on pk, but not on (τ, s, w) anymore.

In **Game** 6 we make a conceptual modification to algorithms Blind_5 and $\mathsf{PrepVer}_5$. We remove all arguments from algorithms which are not used inside the algorithms. Thus, Blind_6 receives as input only pk, and $\mathsf{PrepVer}_6$ receives only (pk, s, w).

Note that we changed only the concrete implementation of the security experiment, but this is completely oblivious to the adversary, thus we have

$$\Pr[X_6] = \Pr[X_5]$$

Now we are in a game where Blind_6 and $\mathsf{PrepVer}_6$ have the correct syntax and functionality such that we can define $\mathsf{SimSetup} := \mathsf{Setup}_6$, $\mathsf{SimBlind} := \mathsf{Blind}_6$, and $\mathsf{SimPrepVer} := \mathsf{PrepVer}_6$. Note that Game 6 with these algorithms is identical to $\mathsf{Exp}^{\mathsf{priv\text{-}ideal}}_{\mathsf{BBA},\mathcal{A}}$. The theorem follows.