

Dynamic Universal Accumulators for DDH Groups and Their Application to Attribute-Based Anonymous Credential Systems

Man Ho Au¹, Patrick P. Tsang^{2,*}, Willy Susilo¹, and Yi Mu¹

¹ Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
{mhaa456,wsusilo,ymu}@uow.edu.au

² Department of Computer Science
Dartmouth College, USA
patrick@cs.dartmouth.edu

Abstract. We present the first dynamic universal accumulator that allows (1) the accumulation of elements in a DDH-hard group \mathbb{G} and (2) one who knows x such that $y = g^x$ has — or has *not* — been accumulated, where g generates \mathbb{G} , to efficiently prove her knowledge of such x in zero knowledge, and hence without revealing, e.g., x or y .

We introduce the *Attribute-Based Anonymous Credential System*, which allows the verifier to authenticate anonymous users according to any access control policy expressible as a formula of *possibly negated* boolean user attributes. We construct the system from our accumulator.

1 Introduction

1.1 Background

Accumulators. Introduced by Benaloh and Mare [5], *accumulators* allow the representation of a set of *elements* $Y = \{y_1, y_2, \dots, y_n\}$ by a single *value* v of size independent of Y 's cardinality; using an initial value u , one can accumulate Y into v by invoking the *accumulating function* f as $v := f(u, Y)$. Accumulators should be *collision-resistant* [3]:

for any element y and any value v , there exists an efficiently computable *witness* w for y w.r.t. v *if and only if* y has been accumulated into v (often abbreviated as “ y is in v ”). To prove that y is in v , one can thus demonstrate the existence of a corresponding w by proving, potentially in zero-knowledge, the knowledge of w .

Several uses of accumulators, e.g., in anonymous credential systems [10], require them to be *dynamic* [12]: one can efficiently update an accumulator value

* Supported in part by the Institute for Security, Technology, and Society, under grant 2005-DD-BX-1091, and the National Science Foundation, under grant CNS-0524695. The views in this paper do not necessarily reflect those of the sponsors.

by adding elements to — and possibly later deleting them from — the value. Furthermore, when a value is updated, e.g., from v to v' , the witness w for some element y w.r.t. v can also be efficiently updated to the witness w' for the same element y w.r.t. the new value v' . Such accumulators are called *dynamic accumulators* (DA's).

Dynamic universal accumulators (DUA's) [20], on the other hand, are DA's with the additional property of *universality*: for any element set Y and any element \bar{y} , there exists an efficiently computable *non-membership witness* \bar{w} for \bar{y} w.r.t. value $v = f(u, Y)$ if and only if $\bar{y} \notin Y$. By demonstrating the existence of \bar{w} , one can prove that \bar{y} is *not* in v . Non-membership witnesses should allow efficient update.

Several existing DA/DUA constructions have $f : (u, \{y_1, y_2, \dots, y_n\}) \mapsto u^{y_1 y_2 \dots y_n} \bmod N$ as their accumulating function [3,12,20], where N is a safe-prime product and $u \in QR(N)^1$. They permit only primes (up to a certain size) to be accumulated. Their security relies on the Strong RSA (SRSA) assumption [3].

Nguyen [21] constructed a DA from bilinear pairings (to be defined later). It has $f : (u, \{y_1, y_2, \dots, y_n\}) \mapsto u^{(s+y_1)(s+y_2)\dots(s+y_n)}$ as the accumulating function, where s is the master secret of the accumulator instance and u is in some group equipped with a bilinear pairing. The construction allows elements in $\mathbb{Z}_p \setminus \{-s\}$ for some prime p to be accumulated. Its security relies on the q -Strong Diffie-Hellman (q -SDH) assumption [7]. Unlike the above “SRSA-based” constructions, dynamically adding an element to a value in Nguyen's construction requires the knowledge of s .

An accumulator would not be too useful (at least for building anonymous credential systems) without a suite of efficient zero-knowledge protocols for proving various facts about the accumulator values and elements. For instance, all the aforementioned constructions are equipped with a protocol for in zero-knowledge that a commitment c opens to some element in an accumulator value v .

Anonymous Credential Systems. In an anonymous credential system (ACS) [10], those and only those users who have registered to an organization O can authenticate their membership in O to any verifier (e.g., a server, another organization, etc.) anonymously and unlinkably among the set of all members in O . Camenisch and Lysyanskaya [10] constructed the first ACS using a *signature scheme with efficient protocols* [11] (commonly referred to as CL-signatures or P-signatures [4]) as a key building block. Many subsequent works have taken the same approach [11,12,13,4].

In this approach, to join an organization O , a user U first registers her pseudonym, which is simply a commitment of her pre-established private key x_U , e.g., in her PKI credential. Pseudonyms (even those of the same user) are hence unlinkable. O then issues a CL-signature σ_U on x_U according to the issuing protocol for CL-signatures, during which O learns nothing about x_U . U uses σ_U as her anonymous credential.

¹ $QR(N)$ denotes the group of quadratic residues modulo N .

To be able to revoke membership efficiently, O can maintain a DA as a “whitelist” of users whose membership has *not* yet been revoked [12], by adding each user U ’s credential σ_U (or its identifier) to its DA when U registers and, when desired, deleting σ_U from DA to revoke U ’s membership. Therefore, to demonstrate her non-revoked membership in O to a verifier V , U conducts a zero-knowledge proof that (1) she has O ’s signature on her private key, and that (2) the signature is a credential in O ’s current DA. Alternatively, O can maintain a DUA as a “blacklist” of users whose membership has been revoked [20]. In this case, to demonstrate her non-revoked membership in O , U instead proves in zero-knowledge that (1) she has O ’s signature on her private key, and that (2) the signature is *not* a credential in O ’s current DUA.

1.2 Attribute-Based Anonymous Credential Systems

As a major contribution of this paper, we present the *Attribute-Based Anonymous Credential System* (ABACS), which generalizes the conventional notion of anonymous credential system (ACS) [10], in a fashion analogous to how Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [6] generalizes public-key encryption, and how attribute certificates generalize identity certificates in X.509 PKIs [18].

Credentials in ABACS can be more precisely referred to as *anonymous attribute credentials* — they are issued to users to certify their possession of an attribute, allowing the users to prove various facts to any verifier about their credential ownership and hence attribute possession in some anonymous fashion. ABACS thus enables *privacy-preserving attribute-based access control*, in which a server is willing to grant a user access to an object such as a file or a service so long as the attributes possessed and/or lacked by the user satisfy the server’s access control policy on the object, while privacy-concerned users desire to access the object by revealing merely the fact that they satisfy the policy, and can thus conceal, e.g., their identity, how they satisfy the policy, and etc.

In this paper, we confine ourselves to *boolean attributes* only. (Some attributes such as age and weight may take a value from a wider range such as non-negative integers and real numbers, and are hence non-boolean.) Boolean attributes provide rich semantics for labeling objects for access control. For example, they can represent group membership, or “roles” in Role-Based Access Control (RBAC).

Features. ABACS is a credential system with the following features.

- Flexible attribute-based access control. The verifier can choose to enforce *any* access control policy expressible as a boolean attribute formula in disjunctive normal form (DNF), i.e., a disjunction of terms, where each term is a conjunction of *possibly negated* boolean attributes, e.g., “(Student \wedge Bio) \vee (\neg Bio)”.
- Multiple ACAs. To support an attribute, a corresponding *Attribute Certification Authority* (ACA) is created (during setup or dynamically when needed) to issue credentials to users to certify their possession of that attribute. These

ACAs are mutually independent; an ACA can only certify the possession of attributes for which it was created. This allows them to have different certification procedures with different trust levels, and confines the damages of their compromises.

- **Robust accountability.** The verifier accepts in the authentication only if the authenticating user satisfies the access control policy being enforced, i.e., the corresponding boolean formula evaluates to **true** on input the set of attribute for which the user has acquired a credential². Hence, a user who has acquired a credential for an attribute can't pretend that she hasn't, and colluding users, none of which alone satisfy the policy, can't satisfy it by pooling together their credentials.
- **Anonymous authentication.** The verifier knows only whether an authenticating user satisfies the access control policy he is enforcing. More precisely, authentication attempts by honest users who (resp. do not) satisfy the verifier's policy are anonymous and unlinkable among the set of all users who also (resp. do not) satisfy the policy.
- **Anonymous certification.** While ACAs must make public some data related to the certification status of users' attribute possession for authentication to be possible, some applications may require that such data reveals no (computational) information about the identity of the certified users, or more generally, no one can tell if two ACAs have issued a credential a common user.
- **Efficiency and practical negation support.** The authentication can be done in $O(|P|)$ time, where $|P|$ is the size of the verifier's policy measured in the number of (negated) attributes in it, and hence regardless of, e.g., the number of users, verifiers, ACAs, or attributes that the authenticating user possesses/lacks. Also, a user who lacks an attribute never has to contact anyone (e.g., the corresponding ACA) before she can prove her lack of the attribute.

Applications. The two scenarios below can benefit from ABACS.

The Biology department provides free parking to its students and any visitor from outside the department. The parking lot entrance hence enforces an access control policy of " $(\text{Bio} \wedge \text{Student}) \vee (\neg \text{Bio})$ ". Identifiable authentication solutions³ would violate the privacy desired by some users. A solution should allow different departments to locally manage their own "membership". Also, a visitor shouldn't have to show up at the Biology Department to get a " $\neg \text{Bio}$ " credential before he or she can park.

A pharmacist must check that " $\text{Fever} \wedge \neg \text{Asthma}$ " holds for a patient before dispensing Aspirin (as many asthma sufferers are allergic to Aspirin), while the patient may not want to disclose her entire medical record, e.g., when she has an unrelated genetic disorder. Also, a fever patient with asthma with the "help" from someone without fever or asthma must still be unable to obtain Aspirin.

² A (resp. negated) attribute in a formula evaluates to **true** if and only if it is (resp. not) contained in the user's attribute set.

³ E.g., waving an RFID card, or an e-token installed with X.509 attribute certificates.

2 Solution Overview

We start with some preliminaries. We then briefly describe how we construct a DUA that allows the accumulation of elements in a DDH-hard group, which we call DUA-DDH. Finally, we highlight how we build ABACS from it.

2.1 Preliminaries

Bilinear pairings. A bilinear pairing is a mapping from a pair of group elements to a group element. Specifically, let \mathbb{G}_1 and \mathbb{G}_2 be some cyclic groups of prime order p . Let g be a generator of \mathbb{G}_1 . A function $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is a bilinear pairing if the following holds:

- *Unique Representation.* Each element in $\mathbb{G}_1, \mathbb{G}_2$ has unique binary representation.
- *Bilinearity.* $e(A^x, g^y) = e(A, B)^{xy}$ for all $A, B \in \mathbb{G}_1$ and $x, y \in \mathbb{Z}_p$.
- *Non-degeneracy.* $e(g, g) \neq 1$, where 1 is the identity element in \mathbb{G}_2 .
- *Efficient Computability.* $e(A, B)$ can be computed efficiently (i.e. in polynomial time) for all $A, B \in \mathbb{G}_1$.

Complexity assumptions. The *Decisional Diffie-Hellman* (DDH) problem in \mathbb{G} is defined as follows: On input a quadruple $(h_0, h_1, h_0^x, y^*) \in \mathbb{G}^4$, output 1 if $y^* = h_1^x$ and 0 otherwise. We say that the DDH assumption holds in \mathbb{G} if no PPT algorithm has non-negligible advantage over random guessing in solving the DDH problem in \mathbb{G} . We call a group *DDH-hard* if the DDH assumption holds in the group.

The *q-Strong Diffie-Hellman* (*q*-SDH) problem in $\mathbb{G} = \langle g_0 \rangle$ is defined as follows: On input a $(q + 1)$ -tuple $(g_0, g_0^\alpha, g_0^{\alpha^2}, \dots, g_0^{\alpha^q}) \in \mathbb{G}^{q+1}$, output a pair $(w, y) \in \mathbb{G} \times \mathbb{Z}_p^*$, where p is the order of \mathbb{G} , such that $w^{(\alpha+y)} = g_0$. We say that the *q*-SDH assumption holds in \mathbb{G} if no PPT algorithm has non-negligible advantage in solving the *q*-SDH problem in \mathbb{G} .

Zero-knowledge proof-of-knowledge. In a zero-knowledge proof of knowledge protocol [19], a prover convinces a verifier that some statement is true without the verifier learning anything except the validity of the statement. We use Camenisch and Stadler’s notation [14]. For example, $PK\{(x) : y = g^x\}$ denotes a zero-knowledge proof-of-knowledge protocol that proves the knowledge of the discrete logarithm of y to the base g .

2.2 Our Dynamic Universal Accumulators for DDH Groups

To construct DUA-DDH, we take Nguyen’s DA construction as the point of departure; we augment *universality* to it. Li et al. [20] presented a technique to augment *universality* to Camenisch and Lysyanskaya’s DA construction [12]. The technique, however, requires the unique factorization of integers and relies on the SRSa assumption, and hence is not immediately applicable to Nguyen’s

DA. Fortunately, we make the observation that the technique works as long as the domain of accumulatable elements is (a subset of) a Euclidean domain. (In the case of Li et al.'s, the domain is the ring of integers.) Consequently, to augment *universality* to Nguyen's construction, we adapt the technique to work on a different Euclidean domain, namely the ring of polynomials over a finite field.

We also equip our accumulator construction with a few useful zero-knowledge protocols. Of particular importance is the following pair:

$$\begin{cases} PK \{(x, y) : C = \text{Com}_1(x) \wedge D = \text{Com}_2(y) \wedge y = g^x \wedge y \text{ is in } v\} \\ PK \{(x, y) : C = \text{Com}_1(x) \wedge D = \text{Com}_2(y) \wedge y = g^x \wedge y \text{ is not in } v\} \end{cases}$$

where Com_1 and Com_2 are commitment schemes and g generates a DDH group, the elements in which can be accumulated in our accumulator. We construct the protocol using Pedersen's commitment scheme [22] and Camenisch's technique for proving double discrete logarithms [14]. The construction has a complexity of $O(\lambda)$ for a cheating probability of $2^{-\lambda}$.

This protocol is the cornerstone of our ABACS construction.

2.3 Our Attribute-Based Anonymous Credential System

Let \mathbb{G} be a DDH group. Let ACA i be the ACA that certifies users' possession of attribute i . Each ACA i instantiates and maintains a DUA-DDH A_i of its own, but for the same \mathbb{G} , and independently picks a generator g_i of \mathbb{G} at random.

Let U be a user with a pre-established private key x . For each attribute i she possesses, she can get certified by ACA i by providing her pseudonym $y_i = g_i^x$ w.r.t. ACA i . ACA i then adds y_i to its A_i . To later revoke the certification, ACA i can simply delete y_i from A_i . Finally, for each attribute j U lacks, she need not do anything (such as contacting ACA j); her pseudonym w.r.t. ACA j is by default *not* in ACA j 's A_j .

Each ACA i publishes A_i, g_i (with a proof of their correct generation) and the list of pseudonyms that have been added in A_i . Thanks to the DDH assumption, no one — not even to the ACAs — can tell which user a pseudonym belongs to, or whether two ACAs' pseudonym lists contain a common user (non-negligibly better than random guessing).

From the published information, a user can compute a (resp. non-) membership witness for each attribute i she has (resp. not) been certified. The first-time computation takes $O(|L_i|)$ time when ACA i has certified $|L_i|$ users. This computation can be further reduced to $O(1)$ by moving the computation to ACA which is in possession of the auxiliary information of the accumulator. Updating the witness in the future take constant time per each change in the list of certified users.

User U who possesses attribute i and has been certified by ACA i can prove such fact to any verifier during authentication by proving that she has the knowledge of some x such that $y_i = g_i^x$ is in A_i . Similarly, if U lacks attribute j , she can prove the fact by proving that $y_j = g_j^x$ is not in A_j . These proofs can be

accomplished in constant time. Generalizing the proof using a standard technique [16], a user can prove the validity of any DNF boolean attribute formula in time linear in the size of the formula.

Due to page limitation, the presentation of our ABACS ends here. More details about its syntax, construction, security formalism and proofs can be found in the full version of this paper [2].

3 Our Dynamic Universal Accumulators for DDH Groups

3.1 Definitions

We incrementally define *Dynamic Universal Accumulators for DDH Groups* (DUA-DDH's). We start by adapting Li et al.'s definition of *universality* to pairing-based accumulators.

Definition 1 (Universal Accumulators (UAs)). A universal accumulator is a scheme with the following properties:

- **Efficient generation** There exists a *Probabilistic Polynomial-Time (PPT)* algorithm Gen that, on input security parameter 1^λ , outputs a tuple $(f, g, \mathcal{Y}_f, u, t_f)$, where f is a function $\mathcal{U}_f \times \mathcal{Y}'_f \rightarrow \mathcal{U}_f$ and g is another function $\mathcal{U}_f \rightarrow \mathcal{U}_g$ for some domains $\mathcal{Y}_f, \mathcal{U}_f, \mathcal{U}_g$; $\mathcal{Y}_f \subseteq \mathcal{Y}'_f$ is the domain for accumulatable elements; t_f is some optional auxiliary information about f ; and u is a element in \mathcal{U}_f . We assume the tuple (f, g) is drawn uniformly at random from its domain.
- **Quasi-commutativity** For all $(f, g, \mathcal{Y}_f, \cdot) \leftarrow \text{Gen}(1^\lambda)$, $v \in \mathcal{U}_f$ and $y_1, y_2 \in \mathcal{Y}'_f$, we have $f(f(v, y_1), y_2) = f(f(v, y_2), y_1)$. Hence, if $Y = \{y_1, \dots, y_k\} \subset \mathcal{Y}'_f$, then we can denote $f(\dots f(f(v, y_1), y_2) \dots, y_k)$ by $f(v, Y)$ unambiguously.
- **Efficient evaluation** For all $(f, g, \mathcal{Y}_f, t_f, u) \leftarrow \text{Gen}(1^\lambda)$, $v \in \mathcal{U}_f$, and $Y \subset \mathcal{Y}_f$ so that $|Y|$ is polynomial in λ , the function $g \circ f(v, Y)$ is computable in time polynomial in λ . $v = g \circ f(u, Y)$ represents the set Y . We call v the accumulator value for Y and say that y has been accumulated into v (or y is “in” v), for all $y \in Y$.
- **Membership (resp. non-membership) witnesses** For all $(f, g, \mathcal{Y}_f, \cdot) \leftarrow \text{Gen}(1^\lambda)$, there exists a relation Ω (resp. $\bar{\Omega}$) that defines membership (resp. non-membership) witnesses: w (resp. \bar{w}) is a valid membership (resp. non-membership) witness for element $y \in \mathcal{Y}_f$ w.r.t. accumulator value $v \in \mathcal{U}_f$ if and only if $\Omega(w, y, v) = 1$ (resp. $\bar{\Omega}(\bar{w}, y, v) = 1$). Membership witness (resp. non-membership witness) should be efficiently computable (in polynomial-time in λ) with t_f . □

The security of universal accumulators requires that it is hard to find a valid membership (resp. non-membership) witness for an element that is *not* in (resp. is *indeed* in) an accumulator value w.r.t. that accumulator value. We employ a strong definition in which the adversary is considered successful even if he present an element that is outside the intended domain of the accumulator (\mathcal{Y}'_f instead

of \mathcal{Y}_f). Accumulators with this stronger sense of security improves efficiency of systems on which it is based because users within this system needs not conduct proof to demonstrate the elements presented is inside the intended domain of the accumulator. Below we give a precise definition.

Definition 2 (Security of Universal Accumulators (UAs)). A universal accumulator is secure if, for any PPT algorithm \mathcal{A} , both P_1 and P_2 are negligible in λ , where:

$$\left\{ \begin{array}{l} P_1 = \Pr \left[(f, g, \mathcal{Y}_f, u, \cdot) \leftarrow \text{Gen}(1^\lambda); (y, w, Y) \leftarrow \mathcal{A}(g \circ f, g, \mathcal{Y}_f, u) : \right. \\ \left. Y \subset \mathcal{Y}'_f \wedge y \in \mathcal{Y}'_f \setminus Y \wedge \Omega(w, y, g \circ f(u, Y)) = 1 \right], \\ P_2 = \Pr \left[(f, g, \mathcal{Y}_f, u) \leftarrow \text{Gen}(1^\lambda); (y, \bar{w}, Y) \leftarrow \mathcal{A}(g \circ f, g, \mathcal{Y}_f, u) : \right. \\ \left. Y \subset \mathcal{Y}'_f \wedge y \in Y \wedge \bar{\Omega}(\bar{w}, y, g \circ f(u, Y)) = 1 \right]. \quad \square \end{array} \right.$$

Definition 3 (Dynamic Universal Accumulators (DUAs)). A DUA is an UA with the following additional properties:

- **Efficient update of accumulator.** There exists an efficient algorithm D_1 such that for all $v = g \circ f(u, Y)$, $y \notin Y$ and $\hat{v} \leftarrow D_1(t_f, v, y)$, we have $\hat{v} = g \circ f(u, Y \cup \{y\})$. If $y \in Y$ instead, then we have $\hat{v} = g \circ f(1, Y \setminus \{y\})$ instead.
- **Efficient update of membership witnesses.** Let v and \hat{v} be the original and updated accumulator values respectively and \hat{y} be the newly added (or deleted) element. There exists an efficient algorithm D_2 that, on input y, w, v, \hat{v} with $y \neq \hat{y}$ and $\Omega(w, y, v) = 1$, outputs \hat{w} such that $\Omega(\hat{w}, y, \hat{v}) = 1$.
- **Efficient update of non-membership witnesses.** Let v and \hat{v} be the original and updated accumulator values respectively and \hat{y} be the newly added (or deleted) element. There exists an efficient algorithm D_3 that, on input y, \bar{w}, v, \hat{v} with $y \neq \hat{y}$ and $\bar{\Omega}(\bar{w}, y, v) = 1$, outputs $\bar{\hat{w}}$ such that $\bar{\Omega}(\bar{\hat{w}}, y, \hat{v}) = 1$. □

In the above, we call an algorithm “efficient” if its time complexity is independent of the cardinality of the accumulated element set Y . Security of DUA is defined as follows. Capabilities of an adversary is defined through queries to oracle O_D which models a working DUA. O_D is initialized with the tuple $(f, g, \mathcal{Y}_f, u, t_f)$ and maintains a list of elements Y , which is initially empty. O_D responds to two types of queries, namely “add y ” and “delete y .” It responds to an “add y ” query by adding y to the set Y , modifying the accumulator value v using algorithm D_1 and sending back the updated accumulator value \hat{v} . It responds to a “delete y ” query by deleting it from set Y , modifying the accumulator value v using algorithm D_1 and sending back the updated accumulator value \hat{v} . In the end, O_D outputs the current set Y and accumulator value v . The following is the definition of secure DUA.

Definition 4 (Security of Dynamic Universal Accumulators (DUAs)). An universal accumulator is secure if, for any PPT algorithm \mathcal{A} , P_3 and P_4 are negligible in λ , where:

$$\left\{ \begin{array}{l} P_3 = \Pr \left[\begin{array}{l} (f, g, \mathcal{Y}_f, u, t_f) \leftarrow \text{Gen}(1^\lambda); (y, w, Y) \leftarrow \mathcal{A}^{OD(f, g, \mathcal{Y}_f, t_f)}(g \circ f, g, \mathcal{Y}_f) : \\ Y \subset \mathcal{Y}'_f \wedge y \in \mathcal{Y}'_f \setminus Y \wedge v = g \circ f(u, Y) \wedge \Omega(w, y, v) = 1 \end{array} \right] \\ P_4 = \Pr \left[\begin{array}{l} (f, g, \mathcal{Y}_f, u, t_f) \leftarrow \text{Gen}(1^\lambda); (y, \bar{w}, Y) \leftarrow \mathcal{A}^{OD(f, g, \mathcal{Y}_f, t_f)}(g \circ f, g, \mathcal{Y}_f) : \\ Y \subset \mathcal{Y}'_f \wedge y \in Y \wedge v = g \circ f(u, Y) \wedge \bar{\Omega}(\bar{w}, y, v) = 1 \end{array} \right] \end{array} \right.$$

□

We state the following theorem. Its proof can be found in the full version of this paper [2].

Theorem 1. A DUA is secure if the underlying UA is secure. □

Finally, a DUA-DDH is a DUA such that there exists a cyclic group $\mathbb{G} \subset \mathcal{Y}_f$ in which the DDH assumption holds.

3.2 Constructions

We construct our DUA-DDH in stages. We first give a construction of UA for DDH groups. We then adds the necessary algorithms for enabling dynamism.

Our UA construction. This construction can be thought as the extension of Nguyen’s accumulator to support *universality*. Our computation of non-membership witnesses involves operations on polynomials over finite fields.

- **Generation.** Let λ be a security parameter. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a bilinear pairing such that $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some λ -bit prime p . Let g_0 be a generator of \mathbb{G}_1 and $\mathbb{G}_q = \langle h \rangle$ be a cyclic group of prime order q such that $\mathbb{G}_q \subset \mathbb{Z}_p^*$.⁴ The generation algorithm Gen randomly chooses $\alpha \in_R \mathbb{Z}_p^*$. For simplicity, we always take the initial element $u = 1$, the identity element in \mathbb{Z}_p^* . The function f is defined as $f : \mathbb{Z}_p^* \times \mathbb{Z}_p^* \rightarrow \mathbb{Z}_p^*$ such that $f : u, y \mapsto u(y + \alpha)$. The function g is defined as $g : \mathbb{Z}_p^* \times \mathbb{G}_1$ such that $g : y \mapsto g_0^y$. The domain \mathcal{Y}_f of accumulatable elements is \mathbb{G}_q .⁵ The auxiliary information t_f is α .
- **Evaluation.** Computing $g \circ f(1, Y)$ efficiently is straightforward with the auxiliary information α . In case one wishes to allow computation of $g \circ f$ without α , one can publish $g_0^{\alpha^i}$ for $i = 0$ to k , where k is the maximum number of elements to be accumulated. If we denote the polynomial $\prod_{y \in Y} (y + \alpha) = \sum_{i=0}^{i=k} (u_i \alpha^i)$ of maximum degree k as $v(\alpha)$, one can efficiently compute $g \circ f(1, Y)$ as $g \circ f(1, Y) = g_0^{v(\alpha)} = \prod_{i=0}^{i=k} g_0^{u_i} \in \mathbb{G}_1$, without the knowledge of α .
- **Membership witnesses.** The relation Ω is defined as $\Omega(w, y, v) = 1$ if and only if $\hat{e}(w, g_0^y g_0^\alpha) = \hat{e}(v, g_0)$. For a set of elements $Y := \{y_1, \dots, y_k\} \in \mathbb{G}_q$, a membership witness for the element $y \in Y$ can be computed in either one of the following ways, depending on whether one knows the auxiliary information.

⁴ If $p = 2q + 1$, one can choose a random element in $h \in_R \mathbb{Z}_p^*$ with order q and set $\mathbb{G}_q = \langle h \rangle$.

⁵ Formally, it is $\mathbb{G}_q \setminus \{-\alpha\}$.

- (With auxiliary information.) Compute the witness as $w = [g_0^{\prod_{i=1}^k (y_i + \alpha)}]_{\alpha + \tilde{y}}$.
 - (Without auxiliary information.) Let $w(\alpha)$ be the polynomial $\prod_{i=1, i \neq j}^k (y_i + \alpha)$. Expand w and write it as $w(\alpha) = \sum_{i=0}^{i=k-1} (u_i \alpha^i)$. Compute the witness as $w = g_0^{w(\alpha)} = \prod_{i=0}^{i=k-1} g_i^{u_i} \in \mathbb{G}_1$.
- **Non-membership witnesses.** The relation $\overline{\mathcal{Q}}$ for non-membership witnesses is defined as $\overline{\mathcal{Q}}(\overline{w}, y, v) = 1$ if and only if $\overline{w} = (c, d)$ and $\hat{e}(c, g_0^y g_0^\alpha) \hat{e}(g_0, g_0)^d = \hat{e}(v, g_0)$. For a set of elements $Y := \{y_1, \dots, y_k\} \in \mathbb{G}_q$, a non-membership witness for $\tilde{y} \notin Y$ can be computed in either one of the following ways, depending on whether one knows the auxiliary information:
- (With auxiliary information.) Compute $\overline{w} = (c, d)$ according to $d = \prod_{i=1}^k (y_i + \alpha) \bmod (\alpha + \tilde{y}) \in \mathbb{Z}_p$ and $c = g_0^{\frac{\prod_{i=1}^k (y_i + \alpha) - d}{\tilde{y} + \alpha}} \in \mathbb{G}_1$.
 - (Without auxiliary information.) Denote the polynomial $v(\alpha)$ as $\prod_{i=1}^k (y_i + \alpha)$. Compute a polynomial division of $v(\alpha)$ by $(\alpha + \tilde{y})$. Since $(\alpha + \tilde{y})$ is a degree one polynomial and $\tilde{y} \neq y_i$ for all i , there exists a degree $k - 1$ polynomial $c(\alpha)$ and a constant d such that $v(\alpha) = c(\alpha)(\alpha + \tilde{y}) + d$. Expand c and write it as $c(\alpha) = \sum_{i=0}^{i=k-1} (u_i \alpha^i)$. Compute $c = g_0^{c(\alpha)} = \prod_{i=0}^{i=k-1} g_i^{u_i} \in \mathbb{G}_1$. The non-membership witness of \tilde{y} is $\overline{w} = (c, d)$.

The theorem below states the security of our UA. Its proof can be found in the full version of this paper [2].

Theorem 2 (Security of our UA construction). Under the k -SDH assumption in \mathbb{G}_1 , the above construction is a secure universal accumulator. \square

Our DUA-DDH construction. We present our construction of DUA-DDH by adding the various dynamism algorithms D_1, D_2, D_3 to our UA construction above. Due to Theorem 1 and 2, our construction is secure under the k -SDH assumption.

- **Update of accumulator (algorithm D_1).** Adding an element \hat{y} to the accumulator value v can be done by computing $\hat{v} = v^{\hat{y} + \alpha}$. Similarly, deleting an element \hat{y} in the accumulator v can be done by computing $\hat{v} = v^{\frac{1}{\hat{y} + \alpha}}$. Both cases require the auxiliary information α .
- **Update of membership witnesses (algorithm D_2).** Let w be the original membership witness of y w.r.t the accumulator value v . Let \hat{v} and \hat{y} be the new accumulator value and the element added (resp. deleted) respectively. Suppose \hat{y} has been added, the new membership witness \hat{w} for y can be computed as $v w^{\hat{y} - y}$. Suppose $\hat{y} \neq y$ has been deleted, the new non-membership witness \hat{w} for y can be computed as $w^{\frac{1}{\hat{y} - y} \hat{v}^{\frac{1}{y - \hat{y}}}}$.
- **Update of non-membership witnesses (algorithm D_3).** Let c, d be the original non-membership witness of y w.r.t. accumulator value v . Let \hat{v} and \hat{y} be the new accumulator value and the element added (resp. deleted) respectively.

- (*Addition.*) Suppose $\hat{y} \neq y$ has been added, the new non-membership witness \hat{c}, \hat{d} of y can be computed as $\hat{c} = v c^{\hat{y}-y} \in \mathbb{G}_1$ and $\hat{d} = d(\hat{y} - y) \in \mathbb{Z}_p^*$. This can be verified as follows:

$$\begin{aligned} \hat{v} &= v^{\alpha+\hat{y}} = v^{(\alpha+y)+(\hat{y}-y)} = v^{\alpha+y} v^{\hat{y}-y} = v^{\alpha+y} (c^{\alpha+y} g_0^d)^{\hat{y}-y} \\ &= [v c^{\hat{y}-y}]^{\alpha+y} g_0^{d(\hat{y}-y)} = \hat{c}^{\alpha+y} g_0^{\hat{d}} \end{aligned}$$

- (*Deletion.*) Suppose \hat{y} has been deleted, the new non-membership witness \hat{c}, \hat{d} of y can be computed as $\hat{c} = (c\hat{v}^{-1})^{\frac{1}{\hat{y}-y}} \in \mathbb{G}_1$ and $\hat{d} = \frac{d}{\hat{y}-y} \in \mathbb{Z}_p^*$. Indeed,

$$\begin{aligned} \hat{v} &= \hat{v}^{\frac{(\alpha+\hat{y})-(\alpha+y)}{\hat{y}-y}} = v^{\frac{1}{\hat{y}-y}} \hat{v}^{\frac{\alpha+y}{\hat{y}-y}} = [c^{\alpha+y} g_0^d]^{\frac{1}{\hat{y}-y}} \hat{v}^{\frac{\alpha+y}{\hat{y}-y}} \\ &= [(c\hat{v}^{-1})^{\alpha+y} g_0^d]^{\frac{1}{\hat{y}-y}} = [(c\hat{v}^{-1})^{\frac{1}{\hat{y}-y}}]^{\alpha+y} g_0^{\frac{d}{\hat{y}-y}} = \hat{c}^{\alpha+y} g_0^{\hat{d}} \end{aligned}$$

4 Zero-Knowledge Protocols for Our DUA-DDH

We present several efficient zero-knowledge protocols for our DUA-DDH construction. In the presentation, we give priority to clarity over efficiency; the protocols may be optimized for better performance.

Let $\mathbb{G}_1 = \langle \mathbf{g} \rangle$ and $\mathbb{G}_q = \langle \mathbf{h} \rangle$ be cyclic groups of prime order p and q respectively, such that $\mathbb{G}_q \subset \mathbb{Z}_p^*$ is the domain of our DUA-DDH construction. Let $\mathbf{g}_0, \mathbf{g}_1$ and $\mathbf{h}_0, \mathbf{h}_1, \mathbf{h}_2$ be independent generators of \mathbb{G}_1 and \mathbb{G}_q respectively. Let $y = \mathbf{h}_0^x \in \mathbb{G}_q$ and let $\mathfrak{C} = \mathbf{g}_0^y \mathbf{g}_1^r \in \mathbb{G}_1$ be the commitment of y using random number r . Let v be an accumulator value.

4.1 Proof of Knowledge of the Discrete Logarithm of a Committed Element

This protocol is the main building block of the protocols used in our DUA-DDH construction. We call it PK_1 . Let $\mathfrak{D} = \mathbf{h}_1^x \mathbf{h}_2^s \in \mathbb{G}_q$ be the commitment of x using some random number s . The goal of PK_1 is to prove the knowledge of x and y such that $y = \mathbf{h}_0^x$ in zero-knowledge, thus without revealing, e.g., x or y . In other words, we have:

$$\text{PK}_1 \left\{ (y, r, x, s) : \mathfrak{C} = \mathbf{g}_0^y \mathbf{g}_1^r \wedge \mathfrak{D} = \mathbf{h}_1^x \mathbf{h}_2^s \wedge y = \mathbf{h}_0^x \right\}$$

The protocol can be used with the common discrete logarithm relationship proofs [9] to demonstrate relationships of discrete logarithms in \mathbb{G}_1 or \mathbb{G}_q . Instantiation of PK_1 makes use of the zero-knowledge proof-of-knowledge of double discrete logarithms [14], as we now describe. Let λ_k be a security parameter that determines the cheating probability of the protocol. (The cheating probability is $2^{-\lambda_k}$, we hence suggest $\lambda_k = 80$.) PK_1 consists of PK_{1A} and PK_{1B} as follows.

$$\text{PK}_1 \begin{cases} \text{PK}_{1A} \left\{ (y, r) : \mathfrak{C} = \mathfrak{g}_0^y \mathfrak{g}_1^r \right\} \\ \text{PK}_{1B} \left\{ (x, r, s) : \mathfrak{C} = \mathfrak{g}_0^{\mathfrak{h}_0^x} \mathfrak{g}_1^r \wedge \mathfrak{D} = \mathfrak{h}_1^x \mathfrak{h}_2^s \right\} \end{cases}$$

Instantiating PK_{1A} is straightforward. Below we only show how to instantiate PK_{1B} .

(Commitment.) For $i = 1$ to λ_k , the prover randomly generates $\rho_{x,i}, \rho_{s,i} \in_R \mathbb{Z}_q$ and $\rho_{r,i} \in_R \mathbb{Z}_p$, computes $T_{1,i} = \mathfrak{g}_0^{\rho_{x,i}} \mathfrak{g}_1^{\rho_{r,i}} \in \mathbb{G}_1$ and $T_{2,i} = \mathfrak{h}_1^{\rho_{x,i}} \mathfrak{h}_2^{\rho_{s,i}} \in \mathbb{G}_q$, and sends $T_{1,i}, T_{2,i}$ to the verifier.

(Challenge.) The verifier randomly generates a λ_k -bit challenge m and sends it to the prover.

(Response.) Denote by $m[i]$ the i -th bit of m , starting from $i = 1$. For $i = 1$ to λ_k , the prover computes $z_{x,i} = \rho_{x,i} - m[i]x \in \mathbb{Z}_q$, $z_{s,i} = \rho_{s,i} - m[i]s \in \mathbb{Z}_q$ and $z_{r,i} = \rho_{r,i} - m[i]\mathfrak{h}_0^x r \in \mathbb{Z}_p$. She sends $(z_{x,i}, z_{s,i}, z_{r,i})_{i=1}^{\lambda_k}$ to the verifier.

(Verify.) The verifier outputs 1 if the following holds for all $i = 1$ to λ_k . He outputs 0 otherwise.

$$T_{2,i} \stackrel{?}{=} \mathfrak{D}^{m[i]} \mathfrak{h}_1^{z_{x,i}} \mathfrak{h}_2^{z_{s,i}} \quad \text{and} \quad T_{1,i} \stackrel{?}{=} \begin{cases} \mathfrak{g}_0^{z_{x,i}} \mathfrak{g}_1^{z_{r,i}}, & \text{if } m[i] = 0, \\ \mathfrak{C}^{\mathfrak{h}_0^{z_{x,i}}} \mathfrak{g}_1^{z_{r,i}}, & \text{otherwise.} \end{cases}$$

It is straightforward to show that PK_1 is Honest-Verifier Zero-Knowledge. It can be converted into a 4-round perfect zero-knowledge protocol using the technique due to Cramer et al. [15] or 3-move concurrent zero-knowledge protocol in the auxiliary string model based on trapdoor commitment schemes [17]. Note that the prover does not need to explicitly prove that the r in PK_{1A} and PK_{1B} are the same; they are bounded to be the same under the discrete logarithm assumption.

4.2 Proof of Knowledge of a Committed Element in an Accumulator Value

Suppose y is in the accumulator value v . That is, there exists witness w such that $\Omega(w, y, v) = 1$. The following protocol demonstrates that the element y , committed as \mathfrak{C} , is in the accumulator value v .

$$\text{PK}_2 \left\{ (w, y, r) : \hat{e}(w, \mathfrak{g}_0^y \mathfrak{g}_0^\alpha) = \hat{e}(v, \mathfrak{g}_0) \wedge \mathfrak{C} = \mathfrak{g}_0^y \mathfrak{g}_1^r \right\}$$

PK_2 can be instantiated using the standard proof-of-knowledge of an SDH-tuple [8,1].

Combining PK_1 and PK_2 , we have a protocol, denoted as PK_3 , that proves the knowledge of the discrete logarithm of an element in an accumulator value:

$$\text{PK}_3 \left\{ (w, y, x) : \hat{e}(w, \mathfrak{g}_0^y \mathfrak{g}_0^\alpha) = \hat{e}(v, \mathfrak{g}_0) \wedge y = \mathfrak{h}_0^x \right\}$$

4.3 Proof of Knowledge of a Committed Element Not in an Accumulator Value

Suppose y is *not* in the accumulator value v . Then there exists witness $\bar{w} = (c, d)$ such that $d \neq 0$ and $\overline{\mathcal{Q}}(\bar{w}, y, v) = 1$. The following protocol demonstrates that the element y , committed as \mathfrak{C} , is *not* in the accumulator value v .

$$\text{PK}_4 \left\{ (c, d, y, r) : \hat{e}(c, g_0^y g_0^r) = \hat{e}(v, g_0) \hat{e}(g_0, g_0)^d \wedge d \neq 0 \wedge \mathfrak{C} = \mathfrak{g}_0^y \mathfrak{g}_1^r \right\}$$

PK_4 can be instantiated using standard techniques, which we describe in the full version of this paper [2].

Combining PK_1 and PK_4 , we have a protocol, denoted as PK_5 , that proves the knowledge of the discrete logarithm of an element not in an accumulator value:

$$\text{PK}_5 \left\{ (c, d, y, x) : \hat{e}(c, g_0^y g_0^x) \hat{e}(g_0, g_0)^d = \hat{e}(v, g_0) \wedge d \neq 0 \wedge y = \mathfrak{h}_0^x \right\}$$

5 Concluding Remarks

We have presented the first dynamic universal accumulator construction for accumulating elements in DDH-hard groups and a number of useful zero-knowledge protocols for it. Using this accumulator, we have built an Attribute-Based Anonymous Credential System, which allows the verifier to authenticate anonymous users according to any access control policy expressible as formula of boolean user attributes in the DNF form. Our system features many practicality and scalability properties for a large-scale deployment of privacy-preserving access control in a heterogeneous and decentralized environment.

We end the paper with two research questions that we believe to be worth exploring in the future. The first one is how one can construct ABACS that also efficiently supports numeric attributes. (While one could certainly encode a numerical attribute by a bunch of boolean attributes, that wouldn't be very efficient.) The second question is how one can construct ABACS that avoids the need to prove double discrete logarithms, and hence achieves better efficiency.

References

1. Au, M.H., Susilo, W., Mu, Y.: Constant-Size Dynamic k -TAA. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 111–125. Springer, Heidelberg (2006)
2. Au, M.H., Tsang, P.P., Susilo, W., Mu, Y.: Dynamic Universal Accumulators for DDH Groups and Their Application to Attribute-Based Anonymous Credential Systems. Cryptology ePrint Archive, Report 2009/044 (2009)
3. Barić, N., Pfizmann, B.: Collision-Free Accumulators and Fail-Stop Signature Schemes without Trees. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 480–494. Springer, Heidelberg (1997)

4. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and Noninteractive Anonymous Credentials. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 356–374. Springer, Heidelberg (2008)
5. Benaloh, J.C., de Mare, M.: One-Way Accumulators: A Decentralized Alternative to Digital Signatures. In: Hellesest, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 274–285. Springer, Heidelberg (1994)
6. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334. IEEE Computer Society, Los Alamitos (2007)
7. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
8. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
9. Camenisch, J.: Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem. PhD Thesis, ETH Zürich, 1998. Diss. ETH No. 12520, Hartung Gorre Verlag, Konstanz (1998)
10. Camenisch, J., Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)
11. Camenisch, J.L., Lysyanskaya, A.: A Signature Scheme with Efficient Protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 268–289. Springer, Heidelberg (2003)
12. Camenisch, J., Lysyanskaya, A.: Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)
13. Camenisch, J., Lysyanskaya, A.: Signature Schemes and Anonymous Credentials from Bilinear Maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
14. Camenisch, J., Stadler, M.: Efficient Group Signature Schemes for Large Groups. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
15. Cramer, R., Damgård, I.B., MacKenzie, P.D.: Efficient Zero-Knowledge Proofs of Knowledge without Intractability Assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–373. Springer, Heidelberg (2000)
16. Cramer, R., Damgård, I.B., Schoenmakers, B.: Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
17. Damgård, I.: Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 418–430. Springer, Heidelberg (2000)
18. Farrell, S., Housley, R.: An Internet Attribute Certificate Profile for Authorization (2002)
19. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof Systems. SIAM J. Comput. 18(1), 186–208 (1989)
20. Li, J., Li, N., Xue, R.: Universal Accumulators with Efficient Nonmembership Proofs. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 253–269. Springer, Heidelberg (2007)
21. Nguyen, L.: Accumulators from Bilinear Pairings and Applications. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 275–292. Springer, Heidelberg (2005)
22. Pedersen, T.P.: Non-interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)