MoniPoly—An Expressive q-SDH-Based Anonymous Attribute-Based Credential System

Syh-Yuan Tan and Thomas Groß

School of Computing, Newcastle University, UK
{syh-yuan.tan, thomas.gross}@newcastle.ac.uk

Abstract Modern attribute-based anonymous credential (ABC) systems benefit from special encodings that yield expressive and highly efficient show proofs on logical statements. The technique was first proposed by Camenisch and Groß, who constructed an SRSA-based ABC system with prime-encoded attributes that offers efficient AND, OR and NOT proofs. While other ABC frameworks have adopted constructions in the same vein, the Camenisch-Groß ABC has been the most expressive and asymptotically most efficient proof system to date, even if it was constrained by the requirement of a trusted message-space setup and an inherent restriction to finite-set attributes encoded as primes. In this paper, combining a new set commitment scheme and an SDH-based signature scheme, we present a provably secure ABC system that supports show proofs for complex statements. This construction is not only more expressive than existing approaches, but it is also highly efficient under unrestricted attribute space due to its ECC protocols only requiring a constant number of bilinear pairings by the verifier; none by the prover. Furthermore, we introduce strong security models for impersonation and unlinkability under adaptive active and concurrent attacks to allow for the expressiveness of our ABC as well as for a systematic comparison to existing schemes. Given this foundation, we are the first to comprehensively formally prove the security of an ABC with expressive show proofs. Specifically, building upon the the q-(co-)SDH assumption, we prove the security against impersonation with a tight reduction. Besides the set commitment scheme, which may be of independent interest, our security models can serve as a foundation for the design of future ABC systems.

1 Introduction

An anonymous attribute-based credential (ABC) system allows a user to obtain credentials, that is, certified attribute set A from issuers and to anonymously

This work was supported in part by the European Research Council Starting Grant "Confidentiality-Preserving Security Assurance (CASCAde)" under Grant GA n°716980.

prove the possession of these credentials as well as properties of A. Anonymous credentials were first proposed by Chaum [**?**] but it does draw much attention until Brands [**?**] constructed a pragmatic single-show ABC system and Camenisch and Lysyanskaya (CL) [**?**] presented a practical multi-show ABC system. CL-ABC system uses the signer's signature on a committed, and therefore blinded, attribute as the user credential. The proof of possession of a valid credential is a zero-knowledge proof of knowledge on the validity of the signature and the wellformedness of the commitment. This commit-and-sign technique has been employed by ABC systems from RSA-based signature scheme [**?**] and pairing-based signature schemes [**? ? ? ? ? ? ? ? ? ?**] on blocks of messages in which the i-th attribute is fixed as the exponent to the i-th base. Therefore, the show proofs have a computational complexity linear to the number of attributes in the credential, in terms of the modular exponentiations and scalar multiplications, respectively.

In contrast to the technique above which is termed as *traditional encoding* by Camenisch and Groß [**? ?**], they suggested a *prime encoding* for the SRSA-CL signature scheme [**?**] to offer show proofs on AND, OR and NOT statements with constant complexity for the prime-encoded attributes. Specifically, the Camenisch-Groß (CG) construction separates the unrestricted attribute space $\mathcal{S}$ into string attributes space and finite-set attributes space such that $\mathcal{S} = \mathcal{S}_S \cup \mathcal{S}_F$. The CG encoding uses a product of prime numbers to represent a finite-set attribute set $A_F \in \mathcal{S}_F$ in a single exponent, a technique subsequently applied to graphs as complex data structures [**? ?**]. Prime encoding results in highly efficient show proofs: each execution only requires a constant number of modular exponentiations. However, the construction constrains $\mathcal{S}_F$ to a set of pre-certified prime numbers and increases the public key size[1]. Furthermore, the security of the CG ABC system was only established on the properties of its show proofs and not formally on the overall properties of the ABC system. Despite these drawbacks, to the best of our knowledge, CG ABC system [**? ?**] is the only ABC system in the standard model that has show proof for AND, OR, and NOT statements with constant complexity.

Related Works. The SDH-CL signature scheme [**? ? ?**] is a popular candidate for the ABC system based on the traditional encoding. It is also referred as the BBS+ signature scheme [**? ? ? ? ? ?**] or the Okamoto signature scheme [**? ?**]. Au et al. [**?**] and Akagi et al. [**?**] constructed provably secure ABC systems on this foundation while Camenisch et al. [**?**] integrated a pairing-based accumulator to yield an ABC system that supports revocation. Later, Sudarsono et al. [**?**] applied the accumulator on $\mathcal{S}_F$ as in prime encoding and showed that the resulting ABC system can support show proofs for AND and OR statements with constant complexity. Yet, the accumulator requires a large public key size: $|\mathcal{S}_F|$ finite-set attributes plus the corresponding $|\mathcal{S}_F|$ signatures. Inspired by the concept of attribute-based signature, Zhang and Feng [**?**] solved the large

[1] If the prime numbers are not pre-certified by a signature each, the show proofs have to include expensive interval proofs.

public key problem, while additionally supporting threshold statements (ANY) in show proofs, at the cost of having the credential size linear to $|A_F|$. Comparing the traditional encoding-based ABC systems to the accumulator-based ABC systems, the latter require more bilinear pairing operations in the show proofs, while having either large public key or credential sizes.

There were some attempts to apply Camenisch et al.'s accumulator [**?**] and its variants on P-signatures [**?**], LRSW-CL signature [**?**] and structure preserving signatures [**? ? ?**] to support complex non-interactive zero-knowledge (NIZK) show proofs. Among all, Sadiah et al.'s ABC system [**?**] offers the most expressive show proofs. Considering only $\mathcal{S} = \mathcal{S}_F$, their ABC system allows constant-size and constant-complexity NIZK show proofs for monotone formulas at the cost of issuing $|\mathcal{P}(A_F)|$ credentials to every user where $\mathcal{P}(A_F)$ is the power set of the user attribute set A_F. Instead of performing this expensive process during the issuing protocol, Okishima and Nakanishi's ABC system [**?**] generates $\mathcal{P}(S_F)$ during key generation and inflates the public key size with $|\mathcal{P}(S_F)|$ signatures to enable constant-size non-interactive witness-indistinguishable (NIWI) show proofs for conjunctive composite formulas. There are also ABC systems [**? ?**] that were built on Pointcheval and Sanders' signature [**?**]. The ABC system proposed by Bemmann et al. [**?**] combines both traditional encoding and accumulator [**?**] to support monotone formulas under the non-interactive proof of partial knowledge protocol [**?**]. Although it has significantly shorter credential and supports unrestricted attribute space compared to that of Sadiah et al.'s [**?**], its show proofs complexity is linear to the number of literals in the monotone formula.

The findings on the use of accumulator in constructing ABC system correspond to the observations in the ABC transformation framework proposed by Camenisch et al. [**?**]. They discovered that the CL signatures are not able to achieve constant-size NIZK show proofs without random oracle. The framework takes in a structure-preserving signature scheme and a vector commitment scheme to produce a UC-secure ABC system. Their instantiation supports constant-size NIZK show proofs on subset statements and provably secure under the common reference string model. Using the similar ingredients, Fuchsbauer et al. [**?**] constructed an ABC system that offers constant-size NIZK show proofs on subset statement. The security models in the two works, however, are not designed to cover expressive show proofs. Other frameworks [**? ?**] that formalized the commit-and-sign technique and even those [**? ? ?**] support show proofs on complex statements also fall short in this aspect.

Research Gap. Existing constructions yield considerable restrictions when expressive show proofs are concerned: The SRSA-based CG scheme [**?**] as well as accumulator-based schemes [**? ? ? ? ? ?**] constrain the attribute space to finite-set attributes ($A_F \in \mathcal{S}_F$) and require a trusted setup that inflates either the public-key size or the credential size. Their expressiveness and the computational complexity are no better than the pairing-based constructions [**? ? ? ? ?**] and the general ABC frameworks [**? ? ?**] alike, when only string attributes ($A_S \in \mathcal{S}_S$) are considered. Expressive proofs for large attribute set are desirable

in privacy-preserving applications such as direct anonymous attestation [**?** **?** **?** **?** **?** **?**]. Also, we observe a need for a systematic canonicalization of security models for all mentioned schemes. In short, an ideal ABC system should have:

1. strong security assurance, and
2. appropriate public key size, and
3. expressive show proofs with low complexity regardless of the attribute space.

Our Contribution. We present a perfectly hiding and computationally binding set commitment scheme, called MoniPoly, which supports set membership proofs and disjointness proofs on the committed messages. Following the commit-and-sign methodology, we combine the MoniPoly commitment scheme tracing back to Kate et. al.'s work [**?**] with SDH-based Camenisch-Lysyanskaya signature scheme [**?** **?**] to present an efficient ABC system that support expressive show proofs for AND, OR and k-out-of-n threshold (ANY) clauses as well as their respective complements (NAND, NOR and NANY). Our ABC system is the most efficient construction for the unrestricted attribute space to-date. And it is at least as expressive as the existing constructions specially crafted for the restricted attribute space.

To the best of our knowledge, neither the constructions nor security models of existing ABC systems allow for complex interactive show proofs. As an immediate contribution, we rigorously define the necessary and stronger security notions for ABC systems. Our notions for security of impersonation resilience and unlinkability under adaptive active and concurrent attacks are stronger than those of the state-of-the-art ABC systems [**?** **?** **?** **?**]. We prove the security of our construction with respect to the security against impersonation and linkability in the standard model, especially offering a tight reduction for impersonation resilience under the q-(co-)SDH assumption.

Organization. We organize the paper as follows. In Section 2, we briefly introduce the underlying SDH-based CL signature scheme. In Section 3, we present the MoniPoly commitment scheme. We present our ABC system which is a combination of the MoniPoly commitment scheme with SDH-based CL signatures [**?** **?**] in Section 4. Section 5 offers an evaluation of the MoniPoly ABC in terms of security properties, expressivity as well as computational complexity in comparison to other schemes in the field.

2 Preliminaries

The MoniPoly commitment and ABC schemes are based on standard mathematical foundations in elliptic curves and bilinear maps as well as notions on signature schemes and proof systems. Readers may refer to the full version [**?**] for this information.

2.1 The SDH-based CL Signature Scheme

Camenisch and Lysyanskaya [**?**] introduced a technique to construct secure pairing-based signature schemes which support signing on committed messages. They also showed that their technique can extract an efficient SDH-based signature scheme from Boneh et al.'s group signature [**?**] scheme but no security proof was provided. This scheme was later proven to be seuf-cma-secure with a tight reduction [**?**] to the SDH assumption in the standard model. We describe the SDH-CL signature scheme [**? ? ?**] as follows:

$\mathsf{KeyGen}(1^k)$: Construct three cyclic groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of order p based on an elliptic curve whose bilinear pairing is $\mathsf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. Select random generators $a, b, c \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and a secret value $x \in \mathbb{Z}_p^*$. Output the public key $pk = (\mathsf{e}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, a, b, c, g_2, X = g_2^x)$ and the secret key $sk = x$.

$\mathsf{Sign}(m, pk, sk)$: On input m, choose the random values $s, t \in \mathbb{Z}_p^*$ to compute $v = (a^m b^s c)^{\frac{1}{x+t}}$. In the unlikely case in which $x + t = 0 \mod p$ occurs, reselect a random t. Output the signature as $sig = (t, s, v)$.

$\mathsf{Verify}(m, sig, pk)$: Given $sig = (t, s, v)$, output 1 if the equation:

$$\begin{aligned} \mathsf{e}(v, Xg_2^t) &= \mathsf{e}((a^m b^s c)^{\frac{1}{x+t}}, g_2^{x+t}) \\ &= \mathsf{e}(a^m b^s c, g_2). \end{aligned}$$

holds and output 0 otherwise.

Theorem 1. *[**?**] SDH-based CL signature scheme is seuf-cma-secure in the standard model if the Strong Diffie-Hellman problem is $(t_{\mathsf{sdh}}, \varepsilon_{\mathsf{sdh}})$-hard.*

3 MoniPoly Set Commitment Scheme

The key idea of set commitment scheme traces back to the polynomial commitment scheme [**?**] which can commit to a polynomial and support opening at indexes of the polynomial. Inheriting this nature, our MoniPoly set commitment scheme and similar ones [**? ?**] transform a message $m \in \mathbb{Z}_p$ into $(x' + m)$ where $x' \in \mathbb{Z}_p$ is not known to the user and multiple messages form a *monic polynomial* $f(x') = \prod_{i=1}^{n}(x' + m_i)$. This monic polynomial, in turn, can be rewritten as $f(x') = \sum_{i=0}^{n} \mathsf{m}_i x'^i$. Its coefficients $\mathsf{m}_i \in \mathbb{Z}_p^*$ can be efficiently computed, for instance, using the encoding algorithm $\mathsf{MPEncode}() : \mathbb{Z}_p^n \rightarrow \mathbb{Z}_p^{n+1}$ described in the full version [**?**].

Our commitment scheme's unique property is that it treats the opening value as one of the roots in the monic polynomial. Hence, the name *MoniPoly*. Folding the opening value into the monic polynomial yields compelling advantages, especially, enabling a greater design space for presentation proofs.

While related schemes [**? ? ?**] realize subset opening, our scheme supports the opening of intersection sets and difference sets, in addition. Thus, MoniPoly is

more expressive. Furthermore, the presentation proofs created on MoniPoly are more efficient than other commitment-based frameworks. Finally, treating the opening value as a root of the monic polynomial yields a scheme that is closely aligned with well-established commitment scheme paradigms, which, in turn, fits into a range of popular signature schemes and enables signing committed messages.

3.1 Interface

We define the MoniPoly set commitment scheme as the following algorithms:

$$\mathsf{MoniPoly} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{OpenIntersection}, \mathsf{VerifyIntersection}, \mathsf{OpenDifference}, \mathsf{VerifyDifference})$$

1. $\mathsf{Setup}(1^k, n) \rightarrow (pk, sk)$. A pair of public and secret keys (pk, sk) are generated by a trusted authority based on the security parameter input 1^k. The message domain $\mathcal{D}$ is defined and $n-1$ is the maximum messages allowed. If n is fixed, sk is not required in the rest of the scheme.
2. $\mathsf{Commit}(pk, A, o) \rightarrow (C)$. On the input of pk, a message set $A \in \mathcal{D}^{n-1}$ and a random opening value $o \in \mathcal{D}$, output the commitment C.
3. $\mathsf{Open}(pk, C, A, o) \rightarrow b$. Return $b = 1$ if C is a valid commitment to A with the opening value o under pk, and return $b = 0$ otherwise.
4. $\mathsf{OpenIntersection}(pk, C, A, o, (A', l)) \rightarrow (I, W)$ or $\perp$. If $|A' \cap A| \geq l$ holds, return an intersection set $I = A' \cap A$ of length l with the corresponding witness W, and return an error $\perp$ otherwise.
5. $\mathsf{VerifyIntersection}(pk, C, (I, W), (A', l)) \rightarrow b$. Return $b = 1$ if W is a witness for S being the intersection set of length l for A' and the set committed to in C, and return $b = 0$ otherwise.
6. $\mathsf{OpenDifference}(pk, C, A, o, (A', \bar{l})) \rightarrow (D, W)$. If $|A' - A| \geq \bar{l}$ holds, return the difference set $D = A' - A$ of length $\bar{l}$ with the corresponding witness W, and return $\perp$ otherwise.
7. $\mathsf{VerifyDifference}(pk, C, (D, W), (A', \bar{l})) \rightarrow b$. Return $b = 1$ if W is the witness for D being the difference set of length $\bar{l}$ for A' and the set committed to in C, and return $b = 0$ otherwise.

3.2 Security Requirements

Definition 1. *A set commitment scheme is perfectly hiding if every commitment* $C = \mathsf{Commit}(pk, A, o)$ *is uniformly distributed such that there exists an* $o' \neq o$ *for all* $A' \neq A$ *where* $\mathsf{Open}(pk, C, A', o') = 1$.

Definition 2. *An adversary* $\mathcal{A}$ *is said to* $(t_{\mathsf{bind}}, \varepsilon_{\mathsf{bind}})$*-break the binding security of a set commitment scheme if* $\mathcal{A}$ *runs in time at most* t_{bind} *and furthermore:*

$$\Pr[\mathsf{Open}(pk, C, A_1, o_1) = \mathsf{Open}(pk, C, A_2, o_2) = 1] \geq \varepsilon_{\mathsf{bind}}.$$

for a negligible probability $\varepsilon_{\mathsf{bind}}$ *and any two pairs* $(A_1, o_1), (A_2, o_2)$ *output by* $\mathcal{A}$. *We say that a set commitment scheme is* $(t_{\mathsf{bind}}, \varepsilon_{\mathsf{bind}})$*-secure wrt. binding if no adversary* $(t_{\mathsf{bind}}, \varepsilon_{\mathsf{bind}})$*-breaks the binding security of the set commitment scheme.*

3.3 Construction

We describe the MoniPoly commitment scheme as follows:

$\mathsf{Setup}(1^k)$. Construct three cyclic groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of order p based on an elliptic curve whose bilinear pairing is $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Select random generators $a \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and a secret values $x' \in \mathbb{Z}_p^*$. Compute the values $a_0 = a, a_1 = a^{x'}, \ldots, a_n = a^{x'^n}, X_0 = g_2, X_1 = g_2^{x'}, \ldots, X_n = g_2^{x'^n}$ to output the public key $pk = (\mathsf{e}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, \{a_i, X_i\}_{0 \le i \le n})$ and the secret key $sk = (x')$. Note that sk can be discarded by the authority if the parameter n is fixed.

$\mathsf{Commit}(pk, A, o)$. Taking as input a message set $A = \{m_1, \ldots, m_{n-1}\} \in \mathbb{Z}_p^*$ and the random opening value $o \in \mathbb{Z}_p^*$, output the commitment as

$$C = a_0^{(x'+o)\prod_{j=1}^{n-1}(x'+m_j)} = \prod_{j=0}^{n} a_j^{\mathsf{m}_j}$$

where $\{\mathsf{m}_j\} = \mathsf{MPEncode}(A \cup \{o\})$.

$\mathsf{Open}(pk, C, A, o)$. Return 1 if $C = \prod_{j=0}^{n} a_j^{\mathsf{m}_j}$ holds where $\{\mathsf{m}_j\} = \mathsf{MPEncode}(A \cup \{o\})$ and return 0 otherwise.

$\mathsf{OpenIntersection}(pk, C, A, o, (A', l))$. If $|A' \cap A| \ge l$ holds, return an intersection set $I = A' \cap A$ of length l and a witness such that:

$$\begin{aligned} W &= a_0^{(x'+o)\prod_{m_j \in (A-I)}(x'+m_j)} \\ &= \prod_{j=0}^{n-l} a_j^{\mathsf{w}_j} \end{aligned}$$

where $\{\mathsf{w}_j\} = \mathsf{MPEncode}((A \cup \{o\}) - I)$. Otherwise, return a null value $\perp$. The correctness can be verified as follows:

$$\begin{aligned} C &= W^{\prod_{m_j \in I}(x'+m_j)} \\ &= \left(a_0^{(x'+o)\prod_{m_j \in (A-I)}(x'+m_j)}\right)^{\prod_{m_j \in I}(x'+m_j)} \\ &= a_0^{(x'+o)\prod_{m_j \in A}(x'+m_j)}. \end{aligned}$$

$\mathsf{VerifyIntersection}(pk, C, I, W, (A', l))$. Return 1 if

$$\mathsf{e}\left(C \prod_{j=0}^{|A'|} a_j^{\mathsf{m}_{1,j}}, X_0\right) = \mathsf{e}\left(W \prod_{j=0}^{|A'|-l} a_j^{\mathsf{m}_{2,j}}, \prod_{j=0}^{l} X_j^{\mathsf{i}_j}\right)$$

holds and return 0 otherwise, where $\{\mathsf{i}_j\} = \mathsf{MPEncode}(I)$, $\{\mathsf{m}_{1,j}\} = \mathsf{MPEncode}(A')$ and $\{\mathsf{m}_{2,j}\} = \mathsf{MPEncode}(A' - I)$. The correctness is as follows:

$$
\begin{aligned}
&\mathsf{e}\left(C\prod_{j=0}^{|A'|} a_j^{\mathsf{m}_{1,j}}, X_0\right) \\
&= \mathsf{e}\left(C, X_0\right)\mathsf{e}\left(\prod_{j=0}^{|A'|} a_j^{\mathsf{m}_{1,j}}, X_0\right) \\
&= \mathsf{e}\left(a_0^{(x'+o)\prod_{m_j\in A}(x'+m_j)}, X_0\right)\mathsf{e}\left(a_0^{\prod_{m_j\in A'}(x'+m_j)}, X_0\right) \\
&= \mathsf{e}\left(a_0^{(x'+o)\prod_{m_j\in (A-I)}(x'+m_j)}, X_0^{\prod_{m_j\in I}(x'+m_j)}\right)\mathsf{e}\left(a_0^{\prod_{m_j\in (A'-I)}(x'+m_j)}, X_0^{\prod_{m_j\in I}(x'+m_j)}\right) \\
&= \mathsf{e}\left(W, \prod_{j=0}^{l} X_j^{\mathsf{i}_j}\right)\mathsf{e}\left(\prod_{j=0}^{|A'|-l} a_j^{\mathsf{m}_{2,j}}, \prod_{j=0}^{l} X_j^{\mathsf{i}_j}\right) \\
&= \mathsf{e}\left(W\prod_{j=0}^{|A'|-l} a_j^{\mathsf{m}_{2,j}}, \prod_{j=0}^{l} X_j^{\mathsf{i}_j}\right)
\end{aligned}
$$

$\mathsf{OpenDifference}(pk, C, A, o, (A', \bar{l}))$. If $|A' \cap A| \geq \bar{l}$ holds, return a difference set $D = A' - A$ of length $\bar{l}$ and the witness $(W = \prod_{j=0}^{n-\bar{l}} a_j^{\mathsf{w}_j}, \{\mathsf{r}_j\}_{j=0}^{\bar{l}-1})$. The values $(\{\mathsf{w}_j\}, \{\mathsf{r}_j\}) = \mathsf{MPEncode}(A)/\mathsf{MPEncode}(D)$ are computed using expanded synthetic division such that $\{\mathsf{w}_j\}$ are the coefficients of quotient $q(x')$ and $\{\mathsf{r}_j\}$ are the coefficients of remainder $r(x')$. Specifically, let the polynomial divisor be $d(x') = \sum_j^{\bar{l}} \mathsf{d}_j x'^j$ where $\{\mathsf{d}_j\} = \mathsf{MPEncode}(D)$, the monic polynomial $f(x')$ in the commitment $C = a_0^{f(x')}$ can be rewritten as $f(x') = d(x')q(x') + r(x')$. Note that $\prod_{j=0}^{\bar{l}-1} a_j^{\mathsf{r}_j} \neq 1_{\mathbb{G}_1}$ whenever $d(x')$ cannot divide $f(x')$, i.e., the sets A and D are disjoint. The correctness can be verified from the following:

$$
\begin{aligned}
C &= a_0^{(x'+o)\prod_{m_j\in A}(x'+m_j)} \\
&= a_0^{q(x')\prod_{m_j\in D}(x'+m_j)} a_0^{r(x')} \\
&= \left(\prod_{j=0}^{n-\bar{l}} a_j^{\mathsf{w}_j}\right)^{d(x')} a_0^{\prod_{m_j\in D}(x'+m_j)} \\
&= W^{d(x')}\prod_{j=0}^{\bar{l}-1} a_j^{\mathsf{r}_j}.
\end{aligned}
$$

$\mathsf{VerifyDifference}(pk, C, D, (W, \{\mathsf{r}_j\}_{j=0}^{\bar{l}-1}), (A', \bar{l}))$. Return 1, if the following holds:

$$\mathsf{e}\left(C\prod_{j=0}^{\bar{l}-1} a_j^{-\mathsf{r}_j}\prod_{j=0}^{|A'|} a_j^{\mathsf{m}_{1,j}}, X_0\right) = \mathsf{e}\left(W\prod_{j=0}^{|A'|-\bar{l}} a_j^{\mathsf{m}_{2,j}}, \prod_{j=0}^{\bar{l}} X_j^{\mathsf{d}_j}\right), \prod_{j=0}^{\bar{l}-1} a_j^{\mathsf{r}_j} \neq 1_{\mathbb{G}_1}$$

and return 0 otherwise, where $\{\mathsf{d}_j\} = \mathsf{MPEncode}(D)$, $\{\mathsf{m}_{1,j}\} = \mathsf{MPEncode}(A')$ and $\{\mathsf{m}_{2,j}\} = \mathsf{MPEncode}(A' - D)$. The correctness is as follows:

$$\begin{aligned}
&\mathsf{e}\left(C\prod_{j=0}^{\bar{l}-1} a_j^{-\mathsf{r}_j}\prod_{j=0}^{|A'|} a_j^{\mathsf{m}_{1,j}}, X_0\right)\\
&= \mathsf{e}\left(C\prod_{j=0}^{\bar{l}-1} a_j^{-\mathsf{r}_j}, X_0\right)\mathsf{e}\left(\prod_{j=0}^{|A'|} a_j^{\mathsf{m}_{1,j}}, X_0\right)\\
&= \mathsf{e}\left(a_0^{d(x')q(x')+r(x')}a_0^{-r(x')}, X_0\right)\mathsf{e}\left(a_0^{\prod_{m_j\in A'}(x'+m_j)}, X_0\right)\\
&= \mathsf{e}\left(a_0^{d(x')q(x')}, X_0\right)\mathsf{e}\left(a_0^{\prod_{m_j\in (A'-D)}(x'+m_j)}, X_0^{\prod_{m_j\in D}(x'+m_j)}\right)\\
&= \mathsf{e}\left(a_0^{\sum_{j=0}^{n-\bar{l}}\mathsf{w}_{1,j}x'^j}, X_0^{d(x')}\right)\mathsf{e}\left(\prod_{j=0}^{|A'|-\bar{l}} a_j^{\mathsf{m}_{2,j}}, X_0^{d(x')}\right)\\
&= \mathsf{e}\left(W\prod_{j=0}^{|A'|-\bar{l}} a_j^{\mathsf{m}_{2,j}}, \prod_{j=0}^{\bar{l}} X_j^{\mathsf{d}_j}\right).
\end{aligned}$$

Remark 1. In the security analysis of MoniPoly, we will take a different approach compared to the previous constructions [**? ? ?**]. We consider the perfectly hiding property and the conventional computational binding property [**?**] that only requires an adversary cannot present two pairs (A_1, o_1) and (A_2, o_2) such that $\mathsf{Commit}(pk, A_1, o_1) = \mathsf{Commit}(pk, A_2, o_2)$. We will show in Section 3.4 that this conventional binding property is a superset of formers' subset binding properties.

3.4 Security Analysis

Theorem 2. *The MoniPoly commitment scheme is perfectly hiding.*

Proof. Given a commitment $C = a_0^{(x'+o)\prod_{j=1}^{n-1}(x'+m_j)}$, there are $|\mathbb{Z}_p^*| - 1$ possible pairs of $((m'_1, \ldots, m'_{n-1}), o') \neq ((m_1, \ldots, m_{n-1}), o)$ which can result in the same C. Furthermore, for every committed message set $\{m_1, \ldots, m_{n-1}\}$, there is a unique o such that:

$$\begin{aligned}
\mathsf{dlog}_{a_0}(C) &= (x' + o)\prod_{j=1}^{n-1}(x' + m_j) \mod p\\
o &= \frac{\mathsf{dlog}_{a_0}(C)}{\prod_{j=1}^{n-1}(x' + m_j)} - x' \mod p
\end{aligned}$$

Since o is chosen independently of the committed messages $\{m_1, \ldots, m_{n-1}\}$, the latter are perfectly hidden. □

The following theorem considers an adversary which breaks the binding property by finding two different message sets A and A^* which can be of different lengths such that $|A| \geq |A^*|$. The proof is in the full version [**?**].

Theorem 3. *The MoniPoly commitment scheme is* $(t_{\mathsf{bind}}, \varepsilon_{\mathsf{bind}})$*-secure wrt. the binding security if the co-SDH problem is* $(t_{\mathsf{cosdh}}, \varepsilon_{\mathsf{cosdh}})$*-hard such that:*

$$\varepsilon_{\mathsf{bind}} = \varepsilon_{\mathsf{cosdh}}, t_{\mathsf{bind}} = t_{\mathsf{cosdh}} + T(n)$$

where $T(n)$ *is the time for dominant group operations in* $\mathbb{G}_1$ *to extract a co-SDH solution where* n *is the total of committed messages plus the opening value.*

4 Attribute-Based Anonymous Credential System

Table 1: Syntax and semantics for an access policy ϕ.

(a) BNF grammar

BNF
attr ::= <attribute>=<value>
set ::= attr,set | attr
con ::= AND | NAND | OR | NOR
cont ::= ANY | NANY
clause ::= con(set) | cont(l,set)
stmt ::= clause $\wedge$ stmt | clause
policy ::= stmt(set) | $\perp$

(b) Truth table with respect to input A

Clause	Truth Condition
OR(A')	$\lvert A' \cap A\rvert > 0$
ANY($1 < l < \lvert A'\rvert, A'$)	$\lvert A' \cap A\rvert \geq l$
AND(A')	$\lvert A' \cap A\rvert = \lvert A'\rvert$
NOR(A')	$\lvert A' \cap \bar{A}\rvert > 0$
NANY($1 < l < \lvert A'\rvert, A'$)	$\lvert A' \cap \bar{A}\rvert \geq l$
NAND(A')	$\lvert A' \cap \bar{A}\rvert = \lvert A'\rvert$

Note: con = connective, cont = connective with threshold

Before presenting the formal definition of ABC system, we briefly define the attribute set A and the access policy ϕ in our proposed ABC system which are closely related to MoniPoly's opening algorithms. Informally, we view a relation between two attribute sets as a clause. Clauses can be accumulated using the logical $\wedge$ operator in building the composite statement for an access policy.

Attribute We view a descriptive attribute set $A = \{m_1, \ldots, m_n\}$ as a user's identity. To be precise, an attribute m is an attribute-value pair in the format attribute=value and A is a set of attributes. For instance, the identity of a user can be described as: $A = \{$"gender = male", "name = bob", "ID = 123456", "role = manager", "branch = Y"$\}$.

Access Policy An access policy ϕ as defined by the BNF grammar in Table 1 expresses the relationship between two attribute sets A and A'. An access policy ϕ is formed by an attribute set A as well as a statement stmt that specifies the relation between A and A'. We have some additional rules for the ϕ where we require $|A| = n > 1$ and $|A'| \leq n$. Besides, in the special case of $|A'| = 1$, the connective must be either AND or NAND. An access policy ϕ outputs 1 if the underlying statement is evaluated to true and outputs 0 otherwise. Taking the attribute set A above as an example, we have $\phi_{\mathsf{stmt}}(A) = \phi_{\mathsf{AND}(A'_1) \wedge \mathsf{OR}(A'_2)}(A) = 1$ for the attribute sets $A'_1 = \{\text{“}\mathsf{role} = \mathsf{manager}\text{”}\}$ and $A'_2 = \{\text{“}\mathsf{branch} = \mathsf{X}\text{”}, \text{“}\mathsf{branch} = \mathsf{Y}\text{”}, \text{“}\mathsf{branch} = \mathsf{Z}\text{”}\}$. Note that the attribute set A' has been implicitly defined by stmt and we simply write ϕ_{stmt} in the subsequent sections when the reference to the attribute set A' is clear.

4.1 Interface

We define an attribute-based anonymous credential system by five algorithms $\mathsf{ABC} = \{\mathsf{KeyGen}, \mathsf{Obtain}, \mathsf{Issue}, \mathsf{Prove}, \mathsf{Verify}\}$ as follows:

1. $\mathsf{KeyGen}(1^k, 1^n) \rightarrow (pk, sk)$: This algorithm is executed by the issuer. On the input of the security parameter k and the attributes upper bound n, it generates a key pair (pk, sk).
2. $(\mathsf{Obtain}(pk, A), \mathsf{Issue}(pk, sk)) \rightarrow (cred \text{ or } \perp)$: These two algorithms form the credential issuing protocol. The first algorithm is executed by the user with the input of the issuer's public key pk and an attribute set A. The second algorithm is executed by the issuer and takes as input the issuer's public key pk and secret key sk. At the end of the protocol, Obtain outputs a valid credential $cred$ produced by Issue or a null value $\perp$ otherwise.
3. $(\mathsf{Prove}(pk, cred, \phi_{\mathsf{stmt}}), \mathsf{Verify}(pk, \phi_{\mathsf{stmt}})) \rightarrow b$: These two algorithms form the credential presentation protocol. The second algorithm is executed by the credential verifier which takes as input the issuer's public key pk and has the right to decide the access policy ϕ_{stmt}. The first algorithm is executed by the credential prover which takes as input the issuer's public key pk, user's credential $cred$ and an access policy ϕ_{stmt} such that $\phi_{\mathsf{stmt}}(A) = 1$. If $\phi_{\mathsf{stmt}}(A) = 0$, the credential holder aborts and Verify outputs $b = 0$. If $\phi = \perp$, prover and verifier complete a proof of possession which proves the validity of credential only instead of a show proof which additionally proves the relation between A and A'. At the end of the protocol, Verify outputs $b = 1$ if it accepts prover and outputs $b = 0$ otherwise.

In the following, we define the key security requirements for an anonymous credential system in the form of *impersonation resilience* and *unlinkability*.

4.2 Security Requirements

4.2.1 Impersonation Resilience. The security goal of an ABC system requires that it is infeasible for an adversary to get accepted by the verifier in

the show proof. The security against impersonation under active and concurrent attacks is described in the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

Game 1 ($\mathsf{imp} - \mathsf{aca}(\mathcal{A}, \mathcal{C})$)

1. ***Setup:*** *$\mathcal{C}$ runs* $\mathsf{KeyGen}(1^k, 1^n)$ *and sends pk to $\mathcal{A}$.*
2. ***Phase 1:*** *$\mathcal{A}$ is able to issue concurrent queries to the* Obtain*,* Prove *and* Verify *oracles where he plays the role of user, prover and verifier, respectively, on any attribute set A_i of his choice in the i-th query. $\mathcal{A}$ can also issue queries to the* IssueTranscript *oracle which takes in A_i and returns the corresponding transcripts of issuing protocol.*
3. ***Challenge:*** *$\mathcal{A}$ outputs the challenge attribute set A^* and its corresponding access policy ϕ^*_{stmt} such that $\phi^*_{\mathsf{stmt}}(A_i) = 0$ and $\phi^*_{\mathsf{stmt}}(A^*) = 1$ for every A_i queried to the* Obtain *oracle during Phase 1.*
4. ***Phase 2:*** *$\mathcal{A}$ can continue to query the oracles as in Phase 1 with the restriction that it cannot query an attribute set A_i to* Obtain *such that $\phi^*_{\mathsf{stmt}}(A_i) = 1$.*
5. ***Impersonate:*** *$\mathcal{A}$ completes a show proof as the prover with $\mathcal{C}$ as the verifier for the access policy $\phi^*_{\mathsf{stmt}}(A^*) = 1$. $\mathcal{A}$ wins the game if $\mathcal{C}$ outputs 1.*

Definition 3. *An adversary $\mathcal{A}$ is said to $(t_{\mathsf{imp}}, \varepsilon_{\mathsf{imp}})$-break the* imp-aca *security of an ABC system if $\mathcal{A}$ runs in time at most t_{imp} and wins in Game 1 such that:*

$$\Pr[(\mathcal{A}, \mathsf{Verify}(pk, \phi^*_{\mathsf{stmt}})) = 1] \geq \varepsilon_{\mathsf{imp}}$$

for a negligible probability $\varepsilon_{\mathsf{imp}}$. We say that an ABC system is imp-aca*-secure if no adversary $(t_{\mathsf{imp}}, \varepsilon_{\mathsf{imp}})$-wins Game 1.*

Note that we reserve the term *unforgeability* of the signature scheme in contrast to some contributions in the literature [**? ? ? ? ? ?**]. One can view our *impersonation resilience* notion as the stronger version of the *misauthentication resistance* from the ABC systems with expressive show proofs [**? ? ?**] which does not cover the active and concurrent adversary besides disallowing adaptive queries. We also introduce a new oracle, namely, IssueTranscript that covers the passive adversary for the issuing protocol. This makes our security definition more comprehensive than that by related works [**? ? ? ?**].

4.2.2 Unlinkability. Unlinkability requires that an adversary cannot link the attributes or instances among the issuing protocols and the presentation protocols. We consider two types of unlinkability notions, namely, *full attribute unlinkability* and *full protocol unlinkability*. We require that an adversary, after being involved in the generation of a list of credentials, cannot differentiate the sequence of two attribute sets in the full attribute unlinkability. The security model for full attribute unlinkability under active and concurrent attacks (aunl-aca) is defined as a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

Game 2 ($\mathsf{aunl} - \mathsf{aca}(\mathcal{A}, \mathcal{C})$)

1. ***Setup:*** *$\mathcal{C}$ runs* KeyGen *and sends pk, sk to $\mathcal{A}$.*
2. ***Phase 1:*** *$\mathcal{A}$ is able to issue concurrent queries to the* Obtain, Issue, Prove *and* Verify *oracles where he plays the role of user, issuer, prover and verifier, respectively, on any attribute set A_i of his choice in the i-th query. $\mathcal{A}$ can also issue queries to an additional oracle, namely,* Corrupt *which takes in a transcript of issuing protocol or show proofs whose user or prover, respectively, is $\mathcal{C}$ and returns the entire internal state, including the random seed used by $\mathcal{C}$ in the transcript.*
3. ***Challenge:*** *$\mathcal{A}$ decides the two equal-length, non-empty attribute sets A_0, A_1 and the access policy ϕ^*_{stmt} which he wishes to challenge such that $\phi^*_{\mathsf{stmt}}(A_0) = \phi^*_{\mathsf{stmt}}(A_1) = 1$. $\mathcal{A}$ is allowed to select A_0, A_1 from the existing queries to* Obtain *in Phase 1. $\mathcal{C}$ responds by randomly choosing a challenge bit $b \in \{0, 1\}$ and interacts as the user with $\mathcal{A}$ as the issuer to complete the protocols:*

$$(\mathsf{Obtain}(pk, A_b), \mathsf{Issue}(pk, sk)) \rightarrow cred_b,$$
$$(\mathsf{Obtain}(pk, A_{1-b}), \mathsf{Issue}(pk, sk)) \rightarrow cred_{1-b}.$$

   Subsequently, $\mathcal{C}$ interacts as the prover with $\mathcal{A}$ as the verifier for polynomially many times as requested by $\mathcal{A}$ to complete the protocols in the same order:

$$(\mathsf{Prove}(pk, cred_b, \phi^*_{\mathsf{stmt}}), \mathsf{Verify}(pk, \phi^*_{\mathsf{stmt}})) \rightarrow 1,$$
$$(\mathsf{Prove}(pk, cred_{1-b}, \phi^*_{\mathsf{stmt}}), \mathsf{Verify}(pk, \phi^*_{\mathsf{stmt}})) \rightarrow 1.$$

4. ***Phase 2:*** *$\mathcal{A}$ can continue to query the oracles as in Phase 1 except querying the transcripts of the challenged issuing and show proofs to* Corrupt.
5. ***Guess:*** *$\mathcal{A}$ outputs a guess b' and wins the game if $b' = b$.*

Definition 4. *An adversary $\mathcal{A}$ is said to $(t_{\mathsf{aunl}}, \varepsilon_{\mathsf{aunl}})$-break the* aunl-aca*-security of an ABC system if $\mathcal{A}$ runs in time at most t_{aunl} and wins in Game 2 such that:*

$$|\Pr[b = b'] - \frac{1}{2}| \geq \varepsilon_{\mathsf{aunl}}$$

for a negligible probability $\varepsilon_{\mathsf{aunl}}$. We say that an ABC system is aunl-aca*-secure if no adversary $(t_{\mathsf{aunl}}, \varepsilon_{\mathsf{aunl}})$-wins Game 2.*

Our full attribute unlinkability is more generic than that in Camenisch et al.'s ABC transformation frameworks [**?**] where we assume the challenged attribute sets A_0, A_1 are not equivalent such that $A_0 \neq A_1$. Besides, unlike Ringers et al.'s unlinkability notion [**?**], ours covers both issuing and show proofs as in Camenisch et al.'s privacy notions [**?**], though the latter does not have a Corrupt oracle while the former does.

On the other hand, as far as we know, the full protocol unlinkability has not been considered before. This notion requires that an adversary, after being involved in the generation of a list of credentials, cannot link an instance of issuing protocol and an instance of a show proof that are under the same credential. The full protocol unlinkability under active and concurrent attacks (punl-aca) is defined as a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

Game 3 ($\mathsf{punl} - \mathsf{aca}(\mathcal{A}, \mathcal{C})$)

1. ***Setup:*** *Same to that of Game 2.*
2. ***Phase 1:*** *Same to that of Game 2.*
3. ***Challenge:*** *$\mathcal{A}$ decides the two equal-length, non-empty attribute sets A_0, A_1 and the access policy ϕ^*_{stmt} which he wishes to challenge such that $\phi^*_{\mathsf{stmt}}(A_0) = \phi^*_{\mathsf{stmt}}(A_1) = 1$. $\mathcal{A}$ is allowed to select A_0, A_1 from the existing queries to* Obtain *in Phase 1. $\mathcal{C}$ responds by randomly choosing two challenge bits $b_1, b_2 \in \{0, 1\}$ and interacts as the user with $\mathcal{A}$ as the issuer to complete the protocols in the order*

$$(\mathsf{Obtain}(pk, A_{b_1}), \mathsf{Issue}(pk, sk)) \rightarrow cred_{b_1},$$
$$(\mathsf{Obtain}(pk, A_{1-b_1}), \mathsf{Issue}(pk, sk)) \rightarrow cred_{1-b_1}.$$

   Subsequently, $\mathcal{C}$ interacts as the prover with $\mathcal{A}$ as the verifier for polynomially many times as requested by $\mathcal{A}$ to complete the protocols in the order

$$(\mathsf{Prove}(pk, cred_{b_2}, \phi^*_{\mathsf{stmt}}), \mathsf{Verify}(pk, \phi^*_{\mathsf{stmt}})) \rightarrow 1,$$
$$(\mathsf{Prove}(pk, cred_{1-b_2}, \phi^*_{\mathsf{stmt}}), \mathsf{Verify}(pk, \phi^*_{\mathsf{stmt}})) \rightarrow 1.$$

4. ***Phase 2:*** *Same to that of full attribute unlinkability game.*
5. ***Guess:*** *$\mathcal{A}$ outputs a guessed pair of issuing protocol transcript $\pi_{(O,I)}$ and show proof transcript $\pi_{(P,V)}$ and wins the game if the pair is under the same credential such that $cred_{\pi_{(O,I)}} = cred_{\pi_{(P,V)}}$.*

Definition 5. *An adversary $\mathcal{A}$ is said to $(t_{\mathsf{punl}}, \varepsilon_{\mathsf{punl}})$-break the* punl-aca*-security of an ABC system if $\mathcal{A}$ runs in time at most t_{punl} and wins in Game 3 such that:*

$$|\Pr[cred_{\pi_{(O,I)}} = cred_{\pi_{(P,V)}}] - \frac{1}{2}| \geq \varepsilon_{\mathsf{punl}}$$

for a negligible probability $\varepsilon_{\mathsf{punl}}$. We say that an ABC system is punl-aca*-secure if no adversary $(t_{\mathsf{punl}}, \varepsilon_{\mathsf{punl}})$-wins Game 3.*

For the completeness of the security notion, we define a security notion weaker than unlinkability, namely, full anonymity in the full version [**?**] and show that Fuchsbauer et al.'s ABC system [**?**] cannot achieve this weaker security notion. Furthermore, we prove that the full attribute unlinkability implies full anonymity in an ABC system but the opposite does not hold. We also show that there is no reduction between full attribute unlinkability and full protocol unlinkability. Therefore, we only prove the security against the full attribute unlinkability and the full protocol unlinkability for our proposed ABC system.

4.3 Construction

Concisely, a user credential *cred* is an SDH-CL signature *sig* on the MoniPoly commitment C of his attribute set A. Next, the show proofs of our ABC system

is proving the validity of sig and C such that:

$$PK\{(\cdots) : 1 = \mathsf{SDH\text{-}CL.Verify}(C, sig, pk) \wedge \\ 1 = \mathsf{MoniPoly.Verify}Pred(pk, C, A, W, (A', l))\}$$

where $Pred = \{\mathsf{Intersection}, \mathsf{Difference}\}$. The commitment verification algorithms are the main ingredient that form the access policy for our ABC system. We describe the proposed ABC system as follows:

$\mathsf{KeyGen}(1^k)$: Construct three cyclic groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of order p based on an elliptic curve whose bilinear pairing is $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Select random generators $a, b, c \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ and two secret values $x, x' \in \mathbb{Z}_p^*$. Compute the values $a_0 = a, a_1 = a^{x'}, \ldots, a_n = a^{x'^n}, X = g_2^x, X_0 = g_2, X_1 = g_2^{x'}, \ldots, X_n = g_2^{x'^n}$ to output the public key $pk = (\mathsf{e}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, p, b, c, \{a_i, X_i\}_{0 \le i \le n}, X)$ and the secret key $sk = (x, x')$.

$(\mathsf{Obtain}(pk, A), \mathsf{Issue}(pk, sk))$: User interacts with verifier as follows to generate a user credential $cred$ on an attribute set $A = \{m_1, \ldots, m_{n-1}\}$.

1. User chooses a random opening value $o \in \mathbb{Z}_p^*$ to compute $C = \prod_{j=0}^{n} a_j^{\mathsf{m}_j} = \mathsf{Commit}(pk, A, o)$. Subsequently, user selects random $s_1 \in \mathbb{Z}_p^*$ to initialize the issuing protocol by completing the protocol with the issuer:

$$PK\left\{(\alpha_0, \ldots, \alpha_n, \sigma) : M = \prod_{j=0}^{n} a_j^{\alpha_j} b^{\sigma}\right\}$$

   where $\sigma = s_1$ and $\{\alpha_0, \ldots, \alpha_n\} = \{\mathsf{m}_0, \ldots, \mathsf{m}_n\}$.
2. Issuer proceeds to the next step if the protocol is verified. Else, issuer outputs $\perp$ and stops.
3. Issuer generates the SDH-CL signature for M as $sig = (t, s_2, v = (Mb^{s_2}c)^{1/(x+t)})$.
4. If sig is not a valid signature on $A \cup \{o\}$, user outputs $\perp$ and stops. Else, user outputs the credential as $cred = (t, s, v, A = A \cup \{o\})$ where:

$$s = s_1 + s_2, v = \left(a_0^{\prod_{j=1}^{n}(x'+m_j)} b^s c\right)^{1/(x+t)}.$$

4.3.1 Proof of Possession. This protocol proves the ownership of a valid credential $cred$ and the wellformedness of the committed attribute set $A = \{m_1, \ldots, m_n\}$ without disclosing any attribute. The Prove and Verify algorithms interact as follows.
$(\mathsf{Prove}(pk, cred, \perp), \mathsf{Verify}(pk, \perp))$:

1. Verifier requests for a proof of possessions protocol by sending an empty access policy $\phi = \perp$.
2. Prover chooses random $r, y \in \mathbb{Z}_p^*$ to randomize the credential as $cred' = (t' = ty, s' = sr^2, v' = v^{r^2 y^{-1}})$.

3. Setting $v', W = \prod_{j=0}^{n-1} a_j^{\mathsf{w}'_j}$ as the public input where $\{\mathsf{w}'_j\}_{0 \le j \le n-1} = r \times \mathsf{MPEncode}(A - \{o\})$, prover runs the zero-knowledge protocol below with the verifier:

$$PK\Bigg\{(\rho, \tau, \gamma, \alpha_0, \alpha_1, \sigma) : \mathsf{e}(C^\rho b^\sigma c^\rho v'^{-\tau}, X_0) = \mathsf{e}(v'^\gamma, X) \wedge \mathsf{e}(C^\rho, X_0) = \mathsf{e}(W, X_1^{\alpha_1} X_0^{\alpha_0})\Bigg\}$$

where $\rho = r^2, \tau = t', \gamma = y, \{\alpha_j\} = r \times \mathsf{MPEncode}(\{o\}), \sigma = s'$. The protocol above can be compressed as:

$$PK\Bigg\{(\rho, \tau, \gamma, \alpha_0, \alpha_1, \sigma) : \mathsf{e}(W, X_1^{\alpha_1} X_0^{\alpha_0}) \mathsf{e}\left(b^\sigma c^\rho v'^{-\tau}, X_0\right) = \mathsf{e}(v'^\gamma, X)\Bigg\}$$

to realize a more efficient proof.
4. Verifier outputs 1 if the protocol is verified and 0 otherwise.

4.3.2 Show Proofs. A show proof proves the relation between the attribute set A in *cred* and the queried set A' chosen by the verifier. Using the same compression technique from the proof of possession, we describe the single clause show proofs by the following presentation protocols.

AND proof. This protocol allows prover to disclose an attribute set $A' = \{m_1, \ldots, m_k\} \subseteq A$ upon the request from verifier and proves that his credential *cred* contains A'. The showing protocol for AND proof is as follows.

$(\mathsf{Prove}(pk, cred, \phi_{\mathsf{AND}(A')}), \mathsf{Verify}(pk, \phi_{\mathsf{AND}(A')}))$:

1. Verifier requests an AND proof for the attribute set $A' = \{m_1, \ldots, m_k\}$.
2. If $A' \not\subseteq A$, prover aborts and the verifier outputs 0.
3. Else, prover chooses random $r, y \in \mathbb{Z}_p^*$ to randomize the credential as $cred' = (t' = ty, s' = sr, v' = v^{ry^{-1}}, \{\mathsf{w}'_j\}_{0 \le j \le n-k} = r \times \mathsf{MPEncode}(A - A'))$.
4. Setting $v', W = \prod_{j=0}^{n-k} a_j^{\mathsf{w}'_j}$ as the public input, prover runs the zero-knowledge protocol below with the verifier:

$$PK\Bigg\{(\rho, \tau, \gamma, \sigma) : \mathsf{e}\left(W, \prod_{j=0}^{k} X_j^{\mathsf{m}_j}\right) \mathsf{e}(b^\sigma c^\rho v'^{-\tau}, X_0) = \mathsf{e}(v'^\gamma, X)\Bigg\}$$

where $\prod_{j=0}^{k} X_j^{\mathsf{m}_j}$ and $\{\mathsf{m}_j\} = \mathsf{MPEncode}(A')$ are computed by the verifier and $\rho = r, \tau = t', \gamma = y, \sigma = s'$.
5. Verifier outputs 1 if the protocol is verified and 0 otherwise.

ANY and OR proofs. This is the show proof for the threshold statement, and it is an OR proof when the threshold is equal to one. Consider the scenario where

the prover is given an attribute set $A' = \{m_1, \ldots, m_k\}$ and he needs to prove that he has l attributes $\{m_j\}_{1 \le j \le l} \in (A' \cap A)$ without the verifier knowing which attributes he is proving. The showing protocol for the ANY statement is as follows.

$(\mathsf{Prove}(pk, cred, \phi_{\mathsf{ANY}(l,A')}), \mathsf{Verify}(pk, \phi_{\mathsf{ANY}(l,A')}))$:

1. Verifier requests an $\mathsf{ANY}(l, A')$ proof for the attribute set $A' = \{m_1, \ldots, m_k\}$.
2. Prover randomly selects l-attribute intersection set $I \subseteq (A' \cap A)$. If no such I can be formed, the prover aborts and the verifier outputs 0.
3. Else, prover chooses random $r, y \in \mathbb{Z}_p^*$ to randomize the credential as $cred' = (t' = ty, s' = sr^2, v' = v^{r^2 y^{-1}}, \{\mathsf{w}'_j\}_{0 \le j \le n-l} = r \times \mathsf{MPEncode}(A - I))$.
4. Setting $v', W = \prod_{j=0}^{n-l} a_j^{\mathsf{w}'_j}, W' = \left(\prod_{j=0}^{k-l} a_j^{\mathsf{m}_{2,j}}\right)^{r^{-1}}$ as the public input where $\{\mathsf{m}_{2,j}\}_{0 \le j \le k-l} = \mathsf{MPEncode}(A' - I)$, prover runs the zero-knowledge protocol below with the verifier:

$$PK\Bigg\{(\rho, \tau, \gamma, \iota_0, \ldots, \iota_l, \sigma) : \mathsf{e}\left(W'W, \prod_{j=0}^{l} X_j^{\iota_j}\right) \mathsf{e}\left(\prod_{j=0}^{k} a_j^{-\mathsf{m}_{1,j}} b^{\sigma} c^{\rho} v'^{-\tau}, X_0\right) = \mathsf{e}(v'^{\gamma}, X)\Bigg\}$$

   where $\prod_{j=0}^{k} a_j^{-\mathsf{m}_{1,j}}$ and $\{\mathsf{m}_{1,j}\}_{0 \le j \le k} = \mathsf{MPEncode}(A')$ are computed by the verifier and $\rho = r^2, \tau = t', \gamma = y, \{\iota_j\}_{0 \le j \le l} = r \times \mathsf{MPEncode}(I), \sigma = s'$.
5. Verifier outputs 1 if the protocol is verified and 0 otherwise.

NAND and NOT proofs. This is the showing protocol for the NAND statement which allows a prover to show that an attribute set $A' = \{m_1, \ldots, m_k\}$ is disjoint with the set A in his credential. Note that is a NOT proof when $|A'| = 1$. The showing protocol on the NAND statement is as below.

$(\mathsf{Prove}(pk, cred, \phi_{\mathsf{NAND}(A')}), \mathsf{Verify}(pk, \phi_{\mathsf{NAND}(A')}))$:

1. Verifier requests a NAND proof for the attribute set $A' = \{m_1, \ldots, m_k\}$.
2. If $|A' - A| < k$, prover aborts and the verifier outputs 0.
3. Else, prover chooses random $r, y \in \mathbb{Z}_p^*$ to randomize the credential as $cred' = (t' = ty, s' = sr, v' = v^{ry^{-1}}, \{\mathsf{w}'_j = r\mathsf{w}_j\}_{0 \le j \le n-k}, \{\mathsf{r}'_j = r\mathsf{r}_j\}_{0 \le j \le k-1})$ where $(\{\mathsf{w}_j\}_{0 \le j \le n-k}, \{\mathsf{r}_j\}_{0 \le j \le k-1}) = \mathsf{MPEncode}(A)/\mathsf{MPEncode}(A')$.
4. Setting $v', W = \prod_{j=0}^{n-k} a_j^{\mathsf{w}'_j}$ as the public input, prover runs the zero-knowledge protocol with the verifier:

$$PK\Bigg\{(\rho, \tau, \gamma, \mu_0, \ldots, \mu_{k-1}, \sigma) : \prod_{j=0}^{k-1} a_j^{\mu_j} \neq \mathbb{G}_1 \wedge \mathsf{e}\left(W, \prod_{j=0}^{k} X_j^{\mathsf{m}_j}\right) \mathsf{e}\left(\prod_{j=0}^{k-1} a_j^{\mu_j} b^{\sigma} c^{\rho} v'^{-\tau}, X_0\right) = \mathsf{e}(v'^{\gamma}, X)\Bigg\}$$

where $\prod_{j=0}^{k} X_j^{\mathsf{m}_j}$ and $\{\mathsf{m}_j\} = \mathsf{MPEncode}(A')$ are computed by the verifier and $\{\mu_j\} = \{\mathsf{r}'_j\}, \rho = r, \tau = t', \gamma = y, \sigma = s'$.
5. Verifier outputs 1 if the protocol is verified and 0 otherwise.

NANY proof. This is the showing protocol for the negated threshold statement. Consider the scenario where the prover is given an attribute set $A' = \{m_1, \ldots, m_k\}$ and he needs to prove that an l-attribute set $D \subseteq (A' - A)$ are not in the credential without the verifier knowing which attributes he is proving. The showing protocol on the NANY statement is as below.

$(\mathsf{Prove}(pk, cred, \phi_{\mathsf{NANY}(\bar{l},A')}), \mathsf{Verify}(pk, \phi_{\mathsf{NANY}(\bar{l},A')}))$:

1. Verifier requests a NANY proof for the attributes $A' = \{m_1, \ldots, m_k\}$.
2. Prover randomly selects an l-attribute difference set $D \in (A' - A)$. If no such D can be formed, prover aborts and the verifier outputs 0.
3. Else, prover chooses random $r, y \in \mathbb{Z}_p^*$ to randomize the credential as $cred' = (t' = ty, s' = sr^2, v' = v^{r^2 y^{-1}}, \{\mathsf{w}'_j = r\mathsf{w}_j\}_{0 \le j \le n-\bar{l}}, \{\mathsf{r}'_j = r^2\mathsf{w}_j\}_{0 \le j \le \bar{l}-1})$ where $(\{\mathsf{w}_j\}_{0 \le j \le n-\bar{l}}, \{\mathsf{r}_j\}_{0 \le j \le \bar{l}-1}) = \mathsf{MPEncode}(A)/\mathsf{MPEncode}(D)$.
4. Setting $v', W = \prod_{j=0}^{n-\bar{l}} a_j^{\mathsf{w}'_j}, W' = \left(\prod_{j=0}^{k-\bar{l}} a_j^{\mathsf{m}_{2,j}}\right)^{r^{-1}}$ as the public input where $\{\mathsf{m}_{2,j}\}_{0 \le j \le k-l} = \mathsf{MPEncode}(A' - D)$, prover runs the zero-knowledge protocol with the verifier:

$$PK\Bigg\{(\rho, \tau, \gamma, \delta_0, \ldots, \delta_{\bar{l}}, \mu_0, \ldots, \mu_{\bar{l}-1}, \sigma) : \prod_{j=0}^{\bar{l}-1} a_j^{\mu_j} \neq \mathbb{G}_1 \wedge$$
$$\mathsf{e}\left(W'W, \prod_{j=0}^{\bar{l}} X_j^{\delta_j}\right) \mathsf{e}\left(\prod_{j=0}^{k} a_j^{-\mathsf{m}_{1,j}} \prod_{j=0}^{\bar{l}-1} a_j^{\mu_j} b^{\sigma} c^{\rho} v'^{-\tau}, X_0\right) = \mathsf{e}(v'^{\gamma}, X)\Bigg\}$$

where $\prod_{j=0}^{k} a_j^{-\mathsf{m}_{1,j}}$ and $\{\mathsf{m}_{1,j}\}_{0 \le j \le k} = \mathsf{MPEncode}(A')$ are computed by the verifier and $\{\mu_j\} = \{\mathsf{r}'_j\}, \rho = r^2, \tau = t', \gamma = y, \{\delta_j\}_{0 \le j \le \bar{l}} = r \times \mathsf{MPEncode}(D), \sigma = s'$.
5. Verifier outputs 1 if the protocol is verified and 0 otherwise.

4.4 Efficiently Enabling Composite Statements

Composite statements, such as, composed of multiple high-level conjunctions, can be realized with MoniPoly efficiently. For that, we propose an efficient strategy instead of naively repeating the show proofs multiple times for an access policy with a composite statement.

The prover runs a proof of possession protocol followed by a proof to show that the committed attributes from every clause in the composite statement is part of the committed attributes in the credential. For instance, given the composite statement $\mathsf{stmt} = \mathsf{AND}(A'_1) \wedge \mathsf{ANY}(l, A'_2)$ where $k_1 = |A'_1|, k_2 = |A'_2|$, a prover can run the showing protocol as follows. Let $W_{A'_1} = \prod_{j=0}^{n-k_1} a_j^{\mathsf{w}'_{A'_1,j}}, W_{A'_2} =$

$\prod_{j=0}^{n-l} a_j^{\mathsf{w}'_{A'_2,j}}, W'_{A'_2} = \prod_{j=0}^{k_2-l} a_j^{\mathsf{m}'_{A'_2,2,j}}$ where $\{\mathsf{w}'_{A'_1,j}\}_{0\le j\le n-k_1} = r^2 \times \mathsf{MPEncode}(A - A'_1)$, $\{\mathsf{w}'_{A'_2,j}\}_{0\le j\le n-l} = r \times \mathsf{MPEncode}(A - I)$, $\{\mathsf{m}'_{A'_2,2,j}\}_{0\le j\le k_2-1} = r^{-1} \times \mathsf{MPEncode}(A'_2 - I)$ for a randomly selected $r \in \mathbb{Z}_p^*$. Setting $v', M_1, M_2, \bar{W}$ as public inputs, the prover runs the showing protocol on ϕ_{stmt} as follows:

$$PK\Bigg\{(\rho,\tau,\gamma,\iota_0,\ldots,\iota_l,\sigma):$$
$$\mathsf{e}\left(W_{A'_1}, \prod_{j=0}^{k} X_j^{\mathsf{m}_{A'_1,j}}\right)\mathsf{e}\left(W'_{A'_2}W_{A'_2}, \prod_{j=0}^{l} X_j^{\iota_j}\right)\mathsf{e}\left(\prod_{j=0}^{k_2} a_j^{-\mathsf{m}_{A'_2,1,j}}(b^\sigma c^\rho v'^{-\tau})^2, X_0\right)$$
$$= \mathsf{e}(v'^{2\gamma}, X)\Bigg\}$$

where $\prod_{j=0}^{k_1} X_j^{\mathsf{m}_{A'_1,j}}, \prod_{j=0}^{k_2} a_j^{\mathsf{m}_{A'_2,2,j}}, \{\mathsf{m}_{A'_1,1,j}\}_{0\le j\le k_1} = \mathsf{MPEncode}(A'_1), \{\mathsf{m}_{A'_2,1,j}\}_{0\le j\le k_2} = \mathsf{MPEncode}(A'_2)$ are computed by the verifier and $\rho = r^2, \tau = t', \gamma = y, \{\iota_j\}_{0\le j\le l} = r \times \mathsf{MPEncode}(I), \sigma = s'$. It is thus obvious that for any composite statement of k clauses, we can run the protocol above in a similar way using $k+2$ pairings. In precise, the $k+1$ pairings on the left-hand side correspond to the k clauses and a credential. Lastly, the corresponding credential elements in the pairings at the left-hand side and right-hand side are brought up to the power of k, respectively. Note that the complexity of $k+2$ parings does not change even when negation clauses are involved.

4.5 Security Analysis

4.5.1 Impersonation Resilience. We establish the security of the *MoniPoly* ABC system by constructing a reduction to the (co-)SDH problem. To achieve tight security reduction, we make use of Multi-Instance Reset Lemma [**?**] as the knowledge extractor which requires the adversary $\mathcal{A}$ to run N parallel instances of impersonation under active and concurrent attacks. The challenger $\mathcal{C}$ can fulfill this requirement by simulating the $N-1$ instances from its given SDH instance which is random self-reducible [**?**]. Since this is obvious, we describe only the simulation for a single instance of impersonation under active and concurrent attacks in the security proofs.

Theorem 4. *If an adversary $\mathcal{A}$ $(t_{\mathsf{imp}}, \varepsilon_{\mathsf{imp}})$-breaks the imp-aca-security of the proposed anonymous credential system, then there exists an algorithm $\mathcal{C}$ which $(t_{\mathsf{cosdh}}, \varepsilon_{\mathsf{cosdh}})$-breaks the co-SDH problem such that:*

$$\frac{\varepsilon_{\mathsf{cosdh}}}{t_{\mathsf{cosdh}}} = \frac{\varepsilon_{\mathsf{imp}}}{t_{\mathsf{imp}}},$$

or an algorithm $\mathcal{C}$ which $(t_{\mathsf{sdh}}, \varepsilon_{\mathsf{sdh}})$-breaks the SDH problem such that:

$$\varepsilon_{\mathsf{imp}} \le \sqrt[N]{\sqrt{\varepsilon_{\mathsf{sdh}}} - 1} + \frac{1 + (q-1)!/p^{q-2}}{p} + 1,$$
$$t_{\mathsf{imp}} \le t_{\mathsf{sdh}}/2N - T(q^2).$$

where N is the total adversary instance, $q = Q_{(O,I)} + Q_{(P,V)}$ is the total query made to the Obtain *and* Verify *oracles, while $T(q^2)$ is the time parameterized by q to setup the simulation environment and to extract the SDH solution. Consider the dominant time elements t_{imp} and t_{sdh} only, we have:*

$$\left(1 - \left(1 - \varepsilon_{\mathsf{imp}} + \frac{1 + (q-1)!/p^{q-2}}{p}\right)^N\right)^2 \leq \varepsilon_{\mathsf{sdh}}, 2Nt_{\mathsf{imp}} \approx t_{\mathsf{sdh}}.$$

Let $N = (\varepsilon_{\mathsf{imp}} - \frac{1+(q-1)!/p^{q-2}}{p})^{-1}$, we get $\varepsilon_{\mathsf{sdh}} \geq (1-e^{-1})^2 \geq 1/3$ and the success ratio is:

$$\frac{\varepsilon_{\mathsf{sdh}}}{t_{\mathsf{sdh}}} \geq \frac{1}{3 \cdot 2Nt_{\mathsf{imp}}}$$
$$\frac{6\varepsilon_{\mathsf{sdh}}}{t_{\mathsf{sdh}}} \geq \frac{\varepsilon_{\mathsf{imp}}}{t_{\mathsf{imp}}} - \frac{1 + (q-1)!/p^{q-2}}{t_{\mathsf{imp}}p}$$

which gives a tight reduction.

To modularize the proof for Theorem 4, we categorize the way an adversary impersonates in Table 2. This is like the approach in the tight reduction proof for the SDH-CL signature scheme proposed by Schäge [**?**]. Subsequently, we differentiate $\mathcal{A}$ into $\mathcal{A} = \{\mathcal{A}_{bind}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\}$ corresponding to four different simulation strategies by $\mathcal{C}$. We omit the proof for the binding property of MoniPoly commitment scheme $\mathcal{A}_{bind}$ which has been described in Theorem 3 and can be trivially applied here.

In each of the simulation strategy, we consider only the success probability of breaking the SDH problem which is weaker than the DLOG problem such that $\varepsilon_{\mathsf{sdh}} \geq \varepsilon_{\mathsf{dlog}}$. Let $M^* = \prod_{j=1}^{n}(x' + m_j^*)$ and $M_i = \prod_{j=1}^{n}(x' + m_{i,j})$ where $A^* = \{m_j^*\}$ and $A_i = \{m_j\}$, respectively, the DLOG problem can be solved whenever the forgery v^* produced by $\mathcal{A}$ equals to a v_i which has been generated by $\mathcal{C}$ such that:

$$\begin{aligned}
\because v^* &\equiv v_i \\
(a_0^{M^*} b^{s^*} c)^{\frac{1}{x+t^*}} &\equiv (a_0^{M_i} b^{s_i} c)^{\frac{1}{x+t_i}} \\
(a_0^{M^*+s^*\beta+\gamma})^{\frac{1}{x+t^*}} &\equiv (a_0^{M_i+s_i\beta+\gamma})^{\frac{1}{x+t_i}} \\
\therefore \frac{M^* + s^*\beta + \gamma}{x + t^*} &\equiv \frac{M_i + s_i\beta + \gamma}{x + t_i} \mod p
\end{aligned}$$

which leads to:

$$x \equiv \frac{t^*M_i - t_iM^* + \beta(t^*s_i - t_is^*) + \gamma(t^* - t_i)}{M^* - M_i + \beta(s^* - s_i)} \mod p$$

where $\mathcal{C}$ can solve the SDH problem using x. Following the equation, the Type 14 impersonation $(A^*, v^*, s^*) = (A_i, v_i, s_i)$ will not happen as it causes a division by zero. On the other hand, Type 16 represents the impersonation using the

Table 2: Types of impersonation and the corresponding assumptions.

Type	A	$\mathsf{MPEncode}(A)$	s t v	**Adversary**	**Assumption**	**Lemmas**
0	0	1	* * *	$\mathcal{A}_{bind}$	co-SDH	Theorem 3
1	0	0	0 0 0	$\mathcal{A}_1$	SDH	1
2	0	0	0 0 1	$\mathcal{A}_1$	DLOG	1
3	0	0	0 1 0	$\mathcal{A}_2$	SDH	2
4	0	0	0 1 1	$\mathcal{A}_2$	DLOG	2
5	0	0	1 0 0	$\mathcal{A}_1$	SDH	1
6	0	0	1 0 1	$\mathcal{A}_1$	DLOG	1
7	0	0	1 1 0	$\mathcal{A}_3$	SDH	3
8	0	0	1 1 1	$\mathcal{A}_3$	DLOG	3
9	1	1	0 0 0	$\mathcal{A}_1$	SDH	1
10	1	1	0 0 1	$\mathcal{A}_1$	DLOG	1
11	1	1	0 1 0	$\mathcal{A}_2$	SDH	2
12	1	1	0 1 1	$\mathcal{A}_2$	DLOG	2
13	1	1	1 0 0	$\mathcal{A}_1$	SDH	1
14	1	1	1 0 1	$\mathcal{A}_1$	N/A	1
15	1	1	1 1 0	$\mathcal{A}_3$	SDH	3
16	1	1	1 1 1	$\mathcal{A}_3$	N/A	3

Note: * = 1 or 0, 1 = equal, 0 = unequal, N/A = not available

uncorrupted *cred* generated by $\mathcal{C}$ when it answers $\mathcal{A}$'s IssueTranscript queries or Verify queries. If $\mathcal{A}$'s view is independent of $\mathcal{C}$'s choice of (t_i, s_i), we have $(t^*, s^*) \neq (t_i, s_i)$ with probability $1 - 1/p$. This causes Type 16 impersonation to happen with a negligible probability of $1/p$ at which point our simulation fails.

We present Lemma 1, 2 and 3 corresponding to the adversaries $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ as follows. The proofs for the lemmas are in the full version [**?**].

Lemma 1. *If an adversary $\mathcal{A}_1$ $(t_{\mathsf{imp}}, \varepsilon_{\mathsf{imp}})$-breaks the imp-aca-security of the proposed anonymous credential system, then there exists an algorithm $\mathcal{C}$ which $(t_{\mathsf{sdh}}, \varepsilon_{\mathsf{sdh}})$-solves the SDH problem such that:*

$$\varepsilon_{\mathsf{imp}} \leq \sqrt[N]{\sqrt{\varepsilon_{\mathsf{sdh}}} - 1} + \frac{1 + (q-1)!/p^{q-2}}{p} + 1,$$
$$t_{\mathsf{imp}} \leq t_{\mathsf{sdh}}/2N - T(q^2).$$

where N is the total of adversary instances, $q = Q_{(O,I)} + Q_{(P,V)}$ is the number of queries made to the Obtain *and* Verify *oracles, while $T(q^2)$ is the time parameterized by q to setup the simulation environment and to extract the SDH solution.*

Lemma 2. *If an adversary $\mathcal{A}_2$ $(t_{\mathsf{imp}}, \varepsilon_{\mathsf{imp}})$-breaks the imp-aca-security of the proposed anonymous credential system, then there exists an algorithm $\mathcal{C}$ which*

$(t_{\mathsf{sdh}}, \varepsilon_{\mathsf{sdh}})$-*solves the SDH problem such that:*

$$\varepsilon_{\mathsf{imp}} \leq \sqrt[N]{\sqrt{\varepsilon_{\mathsf{sdh}}} - 1} + \frac{1 + (q-1)!/p^{q-2}}{p} + 1,$$
$$t_{\mathsf{imp}} \leq t_{\mathsf{sdh}}/2N - T(q^2).$$

where N is the total of adversary instances, $q = Q_{(O,I)} + Q_{(P,V)}$ is the number of queries made to the Obtain *and* Verify *oracles, while $T(q^2)$ is the time parameterized by q to setup the simulation environment and to extract the SDH solution.*

Lemma 3. *If an adversary $\mathcal{A}_3$ $(t_{\mathsf{imp}}, \varepsilon_{\mathsf{imp}})$-breaks the* imp-aca*-security of the proposed anonymous credential system, then there exists an algorithm $\mathcal{C}$ which $(t_{\mathsf{sdh}}, \varepsilon_{\mathsf{sdh}})$-solves the SDH problem such that:*

$$\varepsilon_{\mathsf{imp}} \leq \sqrt[N]{\sqrt{\varepsilon_{\mathsf{sdh}}} - 1} + \frac{(q-1)!/p^{q-2}}{p} + 1,$$
$$t_{\mathsf{imp}} \leq t_{\mathsf{sdh}}/2N - T(q^2).$$

where N is the total of adversary instances, $q = Q_{(O,I)} + Q_{(P,V)}$ is the number of queries made to the Obtain *and* Verify *oracles, while $T(q^2)$ is the time parameterized by q to setup the simulation environment and to extract the SDH solution.*

Combining Theorem 3, Lemmas 1, 2, and 3 gives Theorem 4 as required.

4.5.2 Unlinkability. Next, we prove the unlinkability of the proposed ABC system. It is sufficient to show that the witnesses, the committed attributes and the randomized credential in the issuing protocol and presentation protocol, respectively, are perfectly hiding. Then, we demonstrate that every instance of the protocols is uniformly distributed due to the random self-reducibility property. This implies that even when $\mathcal{A}$ is given access to the Obtain, Issue, Prove, Verify and Corrupt oracles, it does not has advantage in guessing the challenged attribute sets. The proofs for Lemma 5 and 7 are in the full version [**?**].

Lemma 4. *The committed attributes and the corresponding witness in the issuing protocol of the ABC system are perfectly hiding.*

Proof. By Theorem 2, the MoniPoly commitment $C = \prod_{j=0}^{n} a_j^{\mathsf{m}_j}$ in the issuing protocol is perfectly hiding. Subsequently, the value $M = Cb^{s_1}$ is a Pedersen commitment which is also perfectly hiding. The same reasoning is applicable on the commitment value in the zero-knowledge protocol $R = \prod_{j=0}^{n} a_j^{\tilde{\mathsf{m}}_j} b^{\tilde{s}_1}$ which has the same structure as that of M. □

Lemma 5. *The initialization of the issuing protocol in the ABC system has random self-reducibility.*

Lemma 6. *The randomized credential in the presentation protocol of the ABC system are perfectly hiding.*

Proof. Given a user's randomized credential $v' = v^{ry^{-1}}$ in the show proof, there are $|\mathbb{Z}_p^*| - 1$ possible pairs of $(r', y') \neq (r, y)$ which can result in the same v'. Besides, for each r, there is a unique y such that:

$$\mathsf{dlog}_{a_0}(v') = \mathsf{dlog}_{a_0}(v) r y^{-1}$$
$$y = \frac{\mathsf{dlog}_{a_0}(v)}{\mathsf{dlog}_{a_0}(v')} \cdot r$$

Since r, y are chosen independently from each other, and of the credential element v, the latter is perfectly hidden. The same reasoning applies on the randomized credential $v' = v^{r^2 y^{-1}}$. □

Lemma 7. *The presentation protocol of the ABC system offers random self-reducibility.*

Theorem 5. *If the initialization of the issuing protocol and the presentation protocol have random self-reducibility, and their witnesses, committed attributes as well as the randomized credential are perfectly hiding, the ABC system is* aunl-aca*-secure.*

Using the similar approach, we show that the security of full protocol unlinkability also holds for the proposed ABC system.

Theorem 6. *If the initialization of the issuing protocol and the presentation protocol have random self-reducibility, and their witnesses, committed attributes as well as randomized credential are perfectly hiding, the ABC system is* punl-aca*-secure.*

5 Evaluation

5.1 Security

We offer a general overview of security properties in comparison with other schemes here and offer the tightness analysis of our own scheme in the full version [**?**].

We summarize the security properties of ABC systems in either SDH or alternative paradigms in Table 3. The table shows that the relevant schemes vary significantly in their fulfilled security requirements. MoniPoly is the only ABC system that achieves the full range of security requirements. At the same time, it is proven secure in the standard model with a tight security reduction.

5.2 Expressivity and Computational Complexity

In Table 4, we compare the MoniPoly ABC system to relevant popular ABC systems with respect to their realized show proofs and asymptotic computational complexities. Table 4 is normalized in that it considers only the asymptotic complexity for the most expensive operations (e.g., the scalar multiplication, modular exponentiation, or pairing).

Table 3: Security properties of related ABC systems.

ABC System	Impersonation Resilience	Anonymity		Unlinkability			Security Model	Hard Problem	Tight Reduction
		Issuing	Possession	Issuing	Possession	I↔P			
ASM [?]	●	●	●	○	○	○	RO	SDH, DDHI	○
TAKS [?]	●	○	●	○	○	○	RO	SDH, DDH	○
AMO [?]	●	○	●	○	●	○	Standard	SDH, DLIN	○
CKS [?]	●	○	○	○	○	○	Standard	DHE, HSDHE	○
SNF [?]	●	○	○	○	○	○	Standard	SDH, DHE, HSDH, TDH	○
ZF [?]	●	○	●	○	●	○	Standard	SDH, HPDH, HSDH, TDH	○
BNF [?]	◐	○	◐	○	○	○	Standard	DLIN, SFP, DHE	○
CKLMNP [?]	●	○	○	●	●	○	Standard	SRSA, DLOG	○
BBDT [?]	●	○	●	○	○	○	Standard	SDH	○
RVH [?]	●	○	○	○	●	○	Standard	whLRSW	○
SNBF [?]	●	○	●	○	○	○	Standard	DLIN, SFP, DHE	○
ON [?]	◐	○	◐	○	○	○	Standard	DLIN, SFP, DHE	○
CDDH [?]	●	○	●	○	○	○	Standard	SCDHI	○
BB [?]	●	○	●	○	○	○	Generic	SDH, MSDH-1	○
BBBB$^+$ [?]	●	◐	◐	○	○	○	RO	SDH, MSDH-1	○
BBDE [?]	●	●	●	○	○	○	Standard	SDH, MSDH-1	○
CG [? ?]	●	○	○	○	○	○	Standard	SRSA	○
CDHK [?]	●	○	○	●	●	○	CRS	SXDH, RootDH, BSDH, SDH, XDLIN, co-CDH, DBP	○
FHS [?]	●	●	●	○	◐	○	Generic	DDH, co-DLOG, co-SDH	○
This Work	●	●	●	●	●	●	Standard	SDH, co-SDH	●

Note: ●: proof provided, ◐: claim provided, ○: no claim, I: Issue, P: Possession
○ in Issuing: only weak anonymity or unlinkability / trusted issuer / no blind issuing

5.2.1 Expressivity over Unrestricted Attribute Space. The MoniPoly ABC system is the first scheme that can efficiently support all logical statements in the show proofs regardless of the types of attribute space (cf. Table 4). That is, MoniPoly operates on arbitrary attributes while offering a wide range of statements in its expressiveness.

We note that the traditional encoding can achieve the same expressiveness, in principle, in an unrestricted attribute space $\mathcal{S}$ as well as string attribute space $\mathcal{S}_S$. However, traditional encoding will yield inefficient proofs.

5.2.2 Expressivity over Finite-Set Attribute Space. Let us now consider the comparison with schemes with only finite-set attribute space $\mathcal{S}_F$. Most of the accumulator-based ABC systems [**?** **?**] are restricted to finite-set attributes only. While MoniPoly supports negation statements in terms of expressivity, their show proofs do not. The restriction to finite-set attributes and monotone (non-negative) formula affords them a low asymptotic complexity in show proofs. However, their setup and issuing protocols are prohibitively expensive with exponential computational and space complexity ($O(2^{n_F})$ [**?**] and $O(2^{\sqrt{n_F}})$ [**?**]), in turn, restricting the number of attributes that can be feasibly encoded.

The latest ABC system in this line of work [**?**] proposes a workaround on the negated forms of attributes separately. In this scheme, each of its show proof has $O(L)$ complexity where L is the maximum number of $\wedge$ operators permitted

Table 4: Asymptotic complexity for show proofs in related ABC systems.

Property		**ABC System**							
Attribute Space		$\mathcal{S}_F$		$\mathcal{S}_S + \mathcal{S}_F$			$\mathcal{S}$		
Technique		**Accumulator**		**Trad. Encd.**	**Accumulator**	**Prime Encd.**	**Trad. Encd.**	**Comm.**	**MoniPoly**
Setup		$O(n_F)$	$O(2^{n_F})$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Issuing Protocol	Prover	$O(1)$	$O(1)$	$O(1)$	$O(n_S)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
	Verifier	$O(2^{\sqrt{n_F}})$	$O(n_F)$	$O(n)$	$O(n_S)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Show Proofs: Possession	Prover	$O(n_F)$	$O(L)$	$O(n_S) + O(N)$	$O(n_S) + O(1)$	$O(n) + O(1)$	$O(n)$	$O(n)$	$O(n)$
	Verifier	$O(n_F)$	$O(L)$	$O(n_S) + O(N)$	$O(n_S) + O(1)$	$O(n) + O(1)$	$O(n)$	$O(n)$	$O(1)$
$\mathsf{AND}(A')$	Prover	$O(k_F)$	$O(L)$	$O(n_S - k_S) + O(N)$	$O(n_S - k_S) + O(1)$	$O(n_S - k_S) + O(1)$	$O(n-k)$	$O(n-k)$	$O(n-k)$
	Verifier	$O(k_F)$	$O(L)$	$O(n_S) + O(N)$	$O(n_S) + O(1)$	$O(n_S) + O(1)$	$O(n)$	$O(k)$	$O(k)$
$\mathsf{OR}(A')$	Prover	$O(k_F)$	$O(L)$	$O(n_S k_S) + O(N)$	$O(n_S k_S) + O(1)$	$O(n_S k_S) + O(1)$	✗	✗	$O(n+k)$
	Verifier	$O(k_F)$	$O(L)$	$O(n_S k_S) + O(N)$	$O(n_S k_S) + O(1)$	$O(n_S k_S) + O(1)$	✗	✗	$O(k)$
$\mathsf{ANY}(l, A')$	Prover	$O(k_F)$	$O(L)$	$O(n_S!) + O(N)$	✗	✗	✗	✗	$O(n-l+k)$
	Verifier	$O(k_F)$	$O(L)$	$O(n_S!) + O(N)$	✗	✗	✗	✗	$O(k+l)$
$\mathsf{NAND}(A')$	Prover	✗	$O(L)$	✗	✗	$O(n_S - k_S) + O(1)$	✗	✗	$O(n)$
	Verifier	✗	$O(L)$	✗	✗	$O(n_S) + O(1)$	✗	✗	$O(2k)$
$\mathsf{NOR}(A')$	Prover	✗	$O(L)$	✗	✗	✗	✗	✗	$O(n+k)$
	Verifier	✗	$O(L)$	✗	✗	✗	✗	✗	$O(k)$
$\mathsf{NANY}(\bar{l}, A')$	Prover	✗	$O(L)$	✗	✗	✗	✗	✗	$O(n+k)$
	Verifier	✗	$O(L)$	✗	✗	✗	✗	✗	$O(k+2\bar{l})$
Constant Size Proofs		✓	✓	✗	✓	✓	✗	✓	✓
Flexible Attribute Indexing		✗	✗	✗	✗	✗	✗	✓	✓
Schemes		[?]	[?]	[?]	[?]	[?]	[? ? ? ? ?]	[? ?]	This Work

Note: $\mathcal{S}$: attribute space, $k = |A'| \leq n = |A| = n_S + n_F$, S: string attributes, F: finite-attributes, L: maximum allowed $\wedge$ in CNF, N: maximum attributes allowed in a statement, ✓: realized, ✗: not realized

in a composite conjunctive formulae. Moreover, the additional negated finite-set attributes double the credential size and the already massive public key size.

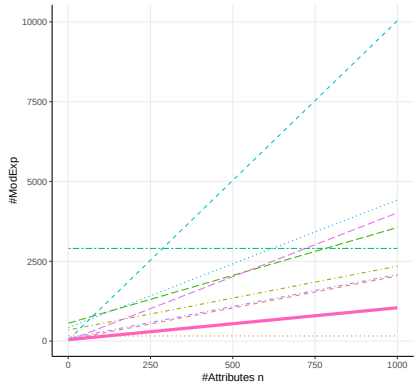
5.2.3 Comparison to Commitment-Based Schemes. MoniPoly bears similarities in terms of computational and communication complexity to other commitment-based ABC systems [**? ?**]. Although MoniPoly does not have constant asymptotic complexity, the verifier is required to compute only three pairings for a single-clause show proof. This makes our scheme the most efficient construction of its kind in this comparison. At the same time, apart from having constant size AND proof similarly to the relevant commitment-based schemes [**? ?**], MoniPoly has constant size possession proof as well as NAND proof.

5.2.4 Parametric Complexity Analysis. We estimate the computational complexity of the schemes listed in Table 4 and present in Figure 1 the complexity for each ABC system at 128-bit security level. While schemes especially crafted for a restricted finite-set attribute space are the fastest schemes in the field, Monipoly is the most efficient ABC system based on commitment schemes and outperforms most schemes in the field, overall. If strength in terms of security properties is a prerequisite, our ABC system outperforms all listed in Table 4 while having efficient constant size show proofs.

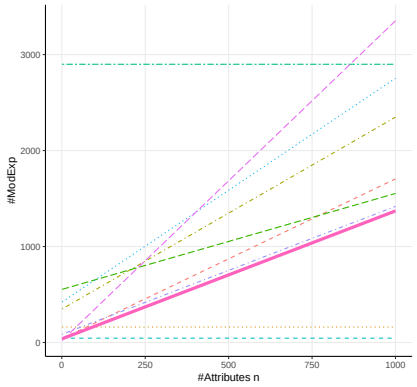
This estimation is based on the following relative computation costs in equivalents of scalar multiplications in $\mathbb{G}_1$:

BLS-12 curve at 128-bit security: for a scalar multiplication in $\mathbb{G}_2$, an exponentiation in $\mathbb{G}_T$ and a pairing, respectively, is about the same as computing 2, 6 and 9 scalar multiplications (M_1) in $\mathbb{G}_1$. The modular exponentiation of RSA-3072 on the other hand is equivalent to $5M_1$.

(a) Proof of possession (128-bit).



(b) AND proof (128-bit).



(c) Scheme

ASM ZF ON CDHK BBBB
SNF SNBF CG FHS This Work

Figure 1: Asymptotic complexity of ABC systems (scalar multiplications in $\mathbb{G}_1$)

We also assume the computational cost in Type-1 pairing friendly curve is equivalent to that of Type-3 as well as $L = 1$ and $N = 1$. The details of the estimation can be found in the full version [**?**].