A practical system for globally revoking the unlinkable pseudonyms of unknown users

Stefan Brands *Liesje Demuynck*
Bart De Decker

Report CW 472, December 2006

Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

A practical system for globally revoking the unlinkable pseudonyms of unknown users

Stefan Brands *Liesje Demuynck*
Bart De Decker

Report CW 472, December 2006

Department of Computer Science, K.U.Leuven

Abstract

We propose the first single sign-on system in which a user can access services using unlinkable digital pseudonyms that can all be revoked in case he or she abuses any one service. Our solution does not rely on key escrow: each user needs to trust only her own computing device with following our protocols in order to be assured of the unconditional untraceability and unlinkability of her pseudonyms. In applications where users hook pseudonyms up to legacy identifiers or legacy accounts at service providers, our system guarantees that service providers (even if they collude with the pseudonym issuer) do not gain any correlation powers over users. Our solution involves two novel ingredients: a technique for invisibly chaining all of a user's pseudonyms in a manner that permits the revocation of all of them on the basis of any one of them (without knowing the user's identity with the issuer) and a sublinear-time proof that a committed number is not on a blacklist without revealing additional information about the number. Our solution is highly practical.

CR Subject Classification : E.3 [Data]: Data Encryption – Public key cryptosystems.

A practical system for globally revoking the unlinkable pseudonyms of unknown users

Stefan Brands
Credentica & McGill School of Comp. Sc.
740 Notre Dame St. W., Suite 1500
Montreal (QC), Canada H3C 3X6
brands@{credentica.com,cs.mcgill.ca}
www.credentica.com

Liesje Demuynck
K.U.Leuven, Dept. of Comp. Sci.
Celestijnenlaan 200A
B-3001 Heverlee, Belgium
Liesje.Demuynck@cs.kuleuven.be
www.cs.kuleuven.be

Bart De Decker
K.U.Leuven, Dept. of Comp. Sci.
Celestijnenlaan 200A
B-3001 Heverlee, Belgium
Bart.DeDecker@cs.kuleuven.be
www.cs.kuleuven.be

December, 2006

Abstract

We propose the first single sign-on system in which a user can access services using unlinkable digital pseudonyms that can all be revoked in case he or she abuses any one service. Our solution does not rely on key escrow: each user needs to trust only her own computing device with following our protocols in order to be assured of the unconditional untraceability and unlinkability of her pseudonyms. In applications where users hook pseudonyms up to legacy identifiers or legacy accounts at service providers, our system guarantees that service providers (even if they collude with the pseudonym issuer) do not gain any correlation powers over users. Our solution involves two novel ingredients: a technique for invisibly chaining all of a user's pseudonyms in a manner that permits the revocation of all of them on the basis of any one of them (without knowing the user's identity with the issuer) and a sublinear-time proof that a committed number is not on a blacklist without revealing additional information about the number. Our solution is highly practical.

1 Introduction

Traditionally, most authenticated relations between users and online services are established on the basis of username and password. As users interact with more and more online services, however, passwords become increasingly vulnerable to phishing and to replay by dishonest service providers. In addition, users are struggling to remember usernames and passwords and which to apply where, which in turn poses a significant burden on the support systems of service providers. As a result, more and more organizations are migrating to secure *single sign-on* (SSO) systems for their users.

This research was performed under the auspices of McGill University (School of Computer Science) from May 2005 until February 2006 when the second author was visiting the first author at Credentica. Liesje Demuynck is supported by a research assistantship and travel credit from the Fund for Scientific Research, Flanders (Belgium).

SSO systems allow a user to access many services in a session without having to manually authenticate more than once. On the flip side, SSO systems give organizations the ability to globally revoke all access privileges of users in case they abuse any one service. This capability is desirable in intra-organizational settings where SSO is used for giving employees and possibly business partners online access to corporate resources: when an employee leaves the company, for example, the organization can centrally revoke all access privileges of that employee.

The demand for secure SSO systems goes beyond organizational boundaries. In the past five years, industry efforts have resulted in a number of specifications and standards aimed at *cross-organizational* SSO. The trust model of cross-organizational SSO is much more complex than that of intra-organizational SSO, however. Indeed, to date very few organizations have adopted SSO systems in consumer-facing settings, and autonomous service providers are even more reluctant to play together in the same SSO system. A major reason for this lack of adoption is that the current generation of cross-organizational SSO systems create potential privacy and security problems for both users and service providers. Namely, these systems revolve around a central *identity provider* that sees in real time which users interact with what service providers. The identity provider can also arbitrarily deny access or revoke all access capabilities of any given user at any time. While these powers may be desirable in intra-organizational settings, they tend to be overly invasive to autonomous users and service providers.

The SSO system that we propose in this paper overcomes these problems.

1.1 Outline of our solution

Our system also relies on the introduction of a central identity provider (which could, of course, be one of the service providers itself), but any unnecessary powers in that identity provider are eliminated. The system works as follows.

The identity provider is responsible for issuing to each user a number of *digital pseudonyms*, which are a special kind of authentication tokens. Users hook their pseudonyms up with service providers and authenticate in subsequent visits by proving knowledge of a secret pseudonym key. Digital pseudonyms are unconditionally unlinkable and untraceable, even vis-à-vis collusions of service providers and the identity provider; thus, by using a different pseudonym with each service provider, each user can ensure that his or her account information with each service provider cannot be compiled into a super-dossier. In case users hook pseudonyms up to legacy identifiers or accounts at service providers, the privacy guarantee of our system is that the correlation powers of service providers and the identity provider do not increase. Replay attacks are prevented, because secret pseudonym keys are never disclosed when authenticating to service providers. Assuming user devices transparently manage pseudonyms on behalf of their users, users can be given an SSO experience; for instance, a single password could locally unlock all of a user's pseudonyms for the duration of a session.

To enable global revocation of all of a user's pseudonyms in case the user abuses any one service, the identity provider "chains" all of the user's pseudonyms in a manner that permits the revocation of all of them on the basis of any one of them. Hereto the identity provider invisibly encodes into all of the pseudonyms of each user a set of random numbers unique to that user (without the identity provider knowing those numbers). For each pseudonym that a service provider associates with a user, the service provider requires its user to disclose one of these encoded random numbers. By disclosing a different random number for each pseudonym, users preserve the unconditional

unlinkability of their pseudonyms. At the same time, service providers can blacklist disclosed numbers in such a manner that users can efficiently prove that their invisibly encoded numbers are not blacklisted without revealing any additional information about them.

This revocation technique does not impinge on user privacy, nor does it give covert powers to service providers and the identity provider. Firstly, the encoding of the invisible numbers into digital pseudonyms requires the cooperation of the user at issuing time. Secondly, in order to be able to blacklist a user, a service provider must ask *all* its users to prove that they are not on its blacklist. Thirdly, in order to compute a blacklist proof users require the blacklist as input, and so they can inspect the blacklist and sanction unreasonable requests for blacklist proofs. Fourthly, proving that one is not on the revocation list does *not* reveal any information about one's identity.

1.2 Comparison to other work

The Liberty Alliance[29] has created an SSO specification called ID-FF. Instead of authenticating to each individual service provider, in ID-FF the user is always redirected to the identity provider, which proceeds by authenticating the user on the service provider's behalf. The user can use any conventional authentication mechanism with the identity provider. Upon having authenticated the user, the identity provider informs the service provider of this fact (using an XML-based framework called SAML). ID-FF provides a "privacy" feature in that the identity provider, upon having authenticated a user on behalf of a service provider, will communicate to that service provider a random alias (also called pseudonym by Liberty Alliance) for that user rather than the user's real name. However, user aliases in Liberty Alliance ID-FF are not pseudonyms at all: they are centrally generated and doled out by the identity provider, which knows the linkage between all of a user's aliases and that user's identity.

Blind signatures, invented in the eighties by Chaum [16, 17], allow users to authenticate at service providers using unconditionally unlinkable pseudonyms. However, when using blind signatures as pseudonyms it is impossible to revoke the pseudonyms of a fraudulent user, whether on the basis of the user's identity with the issuer or on the basis of misuse of any one service. Thus, blind signatures provide privacy for users by trading away security for service providers.

Various adaptations of blind signatures have been proposed to enable global revocation in the context of electronic cash systems, to ensure either (1) that a designated party can identify all e-coin payments of a particular account holder or (2) that all of a payer's payments can be identified if that user engages in any one fraudulent payment transaction. In the context of SSO systems, these two features correspond to the ability to revoke all of a user's pseudonyms for a known user (i.e., based on the user's identity with the issuer) and of an unknown user, respectively. Proposals to extend electronic cash systems with one or both of these revocation features have been published by, amongst others, Brickell, Gemmell, and Kravitz [7], Stadler, Piveteau, and Camenisch [30], Camenisch, Maurer, and Stadler [14], Jakobsson and Yung [24], and Davida, Frankel, Tsiounis, and Yung [23]. Unfortunately, in all of these and other proposals, the privacy of users is in fact illusional; most of the presented techniques [7, 30, 14, 24, 23] rely on key escrow: the bank encodes into each e-coin a tracing key that its user must disclose in encrypted form at payment time, so that it can be decrypted by a designated "escrow agent" (or set of parties) if needed. Furthermore, in e-cash systems not requiring a trusted escrow agent [10], users have to settle for computational privacy only. This is however *not* the standard for e-cash schemes.

More recently, Camenisch and Lysyanskaya [8, 11] and Nguyen [28] proposed credential revo-

cation mechanisms that do not rely on key escrow. These proposals make use of so-called dynamic accumulators, which can be used to prove membership on a list in constant time in the list size. However, the security of these accumulators relies on non-standard intractability assumptions, such as the strong RSA assumption [11] and the q-strong Diffie-Hellman assumption [28]. In the case of [11], the proofs of knowledge are statistical zero-knowledge only and the set of accumulatable values is limited to prime numbers in a predefined interval. In addition, the schemes merely allow one to revoke the credentials of users on the basis of their identity with the issuer; it is not possible to globally revoke the pseudonyms of an unknown user.

An accumulator-based membership proof consists of two steps; the computation of the user's current "witness" (which is a secret value related to the prover's accumulated values) and the execution of a zero-knowledge proof of knowledge. Although the latter can be executed in constant time, the former requires a time complexity at least linear in the number of elements deleted from the accumulator. Taking an accumulator to which no elements are added and n elements are removed, and assuming that a small exponentiation has an exponent size equal to the maximal size of an accumulated value, the recomputation of a witness requires at its best n small exponentiations. Remarkably, in the case of [11], these n small exponentiation correspond to only two exponentiations of which the exponents are very large. In addition, the final witness can only be computed when the final blacklist is known. Hence, not all of the user's exponentiations can be precomputed.

In the context of direct anonymous attestation, Brickell et al. [6] suggest a technique in which a user provides the service provider with a value $N_V = \zeta^f$ for f a credential's attribute and ζ a random generator of a group in which the DL-problem is hard. The purpose of N_V is twofold: providing $\mathcal{V}$ with a pseudonym and enabling credential revocation based either on the knowledge of f or (as suggested by an anonymous reviewer) on a list of pseudonyms $\{(N_{V'}) = (\zeta')^{f'}, \ldots\}$. The latter can be achieved by proving in zero knowledge that $\log_{\zeta'} N_{V'} \neq \log_{\zeta} N_V$ for all $N_{V'}$ in this blacklist [15]. This solution has two major drawbacks. Firstly, the prover's privacy is only computational. Second, the proof that a pseudonym is not revoked based on a pseudonym blacklist requires a number of exponentiations linear in the length of the blacklist.

Brands [4] proposed a practical digital credential mechanism that allows an issuer to invisibly encode into all of a user's digital credentials a unique number that the issuer can blacklist in order to revoke that user's credentials. This mechanism does not rely on key escrow and preserves the unconditional untraceability and unlinkability of credentials; as such, it offers the same privacy strength as our proposal. Furthermore, base credentials in the system are as efficient as standard DSA signatures, and the blacklist technique (which consists of repeating a NOT-proof for each blacklist element) is provably secure under the discrete log assumption. However, Brands' proposal does not allow the revocation of all of the pseudonyms of an unknown user. In addition, the complexity of the cryptographic proof for showing that one's invisibly encoded number is not contained on a blacklist grows linearly in the size of the blacklist; in practice, some 40 bytes must be sent along for each blacklist element, and each such data packet involves the computation by the user of one modular exponentiation. As such, the proposal is not practical for large blacklists.

Our proposal addresses both shortcomings by extending Brands' credentials system using two new techniques: a way to generalize Brands' credentials so that an entire set of credentials of an unknown user can be revoked on the basis of something unique to any one of the credentials in the set, and a cryptographic blacklist proof with sublinear complexity in the size of the blacklist. The latter is based on the well accepted discrete logarithm assumption and provides unconditional privacy for the user. In addition, the user can precompute all of her exponentiations long before

the final blacklist is known.

The remainder of this paper is organized as follows. In Section 2 we provide a backgrounder on Brands' credential techniques to the extent necessary to explain our new system. As to motivate our choice for Brands' credentials, we additionally compare this system with other well-known credential systems. In Section 3 we describe our new system in detail and analyze the security and privacy properties of the two new techniques that are at their core. In Section 4 we analyze the practicality of the proposal. Finally, in Section 5 we outline various extensions and variations.

2 Digital credentials

Our new system is based on Brands' digital credentials [5]. Section 2.1 provides a backgrounder on these credential techniques. As to motivate our choice for Brands' credentials, Section 2.2 gives a comparison of Brands' system with other credential techniques.

2.1 Backgrounder of Brands' digital credentials

In the system of Brands [4], credentials are issued by a Credential Authority (CA) that has its own key pair for digitally signing messages. When issuing a credential to a user Alice, the CA through its digital signature binds one or more attributes to a digital credential public key, the secret key of which only Alice knows. The whole package that Alice receives is called a digital credential.

When Alice shows her digital credential to Bob, she not only sends her digital credential public key and the CA's signature, but she also digitally signs a *nonce* using her secret key. This prevents Bob from replaying the digital credential, since in each showing protocol execution a new nonce must be signed, which requires knowledge of Alice's secret key. In the showing protocol, Alice may also selectively disclose a property of the attributes in her digital credential, while hiding any other information about them; to convince Bob that the claimed property is true, Alice's signature on Bob's nonce doubles up as a cryptographic proof of correctness.

Since Alice reveals the digital credential public key and the CA's signature when showing a digital credential, these elements must be uncorrelated to the information that the CA sees when it issues the digital credential, even if the CA tries to cheat. At the same time, the CA must be able to encode the desired attributes into the digital credential, even if Alice tries to cheat. Here is how $l \geq 1$ attributes, $(x_1, \ldots, x_l)$, are encoded in a digital credential in the issuing protocols of Brands. The tuple $(x_1, \ldots, x_l, s)$ is Alice's secret key for the digital credential. Alice generates s at random from $\mathbb{Z}_q$ in the issuing protocol. Even though Alice may disclose some or all of the attributes to Bob in the showing protocol, she keeps s secret at all times; this ensure that only she knows the entire secret key. The digital credential public key is equal to the product $g_1^{x_1} \cdots g_l^{x_l} h_0^s$. The elements $g_1, \ldots, g_l, h_0$ are randomly chosen generators of a group G_q of prime order q; they are part of the CA's public key. (The actual form is slightly more complex, in that s spreads across all elements, but this is irrelevant for the present overview.) The digital credential public key reveals no information about $x_1, \ldots, x_l$: for any public key and for any attribute tuple $(x_1, \ldots, x_l)$, there is exactly one $s \in \mathbb{Z}_q$ that would make the match. At the same time, the following security property holds (see Brands [4, Proposition 2.3.3.]): Regardless of the choice of l, assuming it is infeasible to compute discrete logarithms in G_q, Alice cannot compute a digital credential public key for which she knows more than one secret key. Consequently, by signing the digital credential public key the

CA binds a unique attribute tuple to Alice's digital credential, albeit in an indirect manner: the *CA*'s signature binds Alice's public key, which in turn binds Alice's secret key, which contains all the attributes.

To show a digital credential to Bob, Alice transmits to him the digital credential public key and the *CA*'s digital signature. In addition, she selectively discloses a property of the attributes and digitally signs a nonce using her secret key, as explained. Alice's signature, which is derived from a *proof of knowledge*, proves not only that she knows a secret key but also that the attributes in her digital credential satisfy whichever particular attribute property she is disclosing to Bob. Assuming it is infeasible to compute discrete logarithms in G_q, Bob cannot compute any secret key when presented with Alice's digital credential public key, regardless of which property of $(x_1, \ldots, x_l)$ Alice discloses to him. Alice can demonstrate a wide spectrum of attribute properties to Bob. Among others, using the notation $(x_1, \ldots, x_l) = \text{rep}_{(g_1,\ldots,g_l)} h$ to refer to a *representation* $(x_1, \ldots, x_l)$ such that $h = g_1^{x_1} \ldots g_l^{x_l}$, Alice can prove any of the following properties:

- **Knowledge of a representation containing known attribute values [4, Chapter 3]:** Alice can prove knowledge of a representation $(x_1, \ldots, x_l)$ of $h \in G_q$ with respect to any $(g_1, \ldots, g_l) \in G_q^l$, and in doing so she can disclose any value x_j ($j \in \{1, \ldots, l\}$). We denote this protocol by $PK\{(\chi_1, \ldots, \chi_{j-1}, \chi_{j+1}, \ldots, \chi_l) : (\chi_1, \ldots, \chi_{j-1}, x_j, \chi_{j+1}, \ldots, \chi_l) = \text{rep}_{(g_1,\ldots,g_l)} h\}$. (Greek letters represent the values that remain unknown to Bob.) More generally, Alice can selectively disclose any subset of encoded attributes.

- **Knowledge and equality of discrete logarithms [18]:** Given values h_1, h_2, g_1, g_2, g_3 and g_4 in G_q, Alice can demonstrate to Bob her knowledge of a tuple (x_1, x_2, x_3) such that $h_1 = g_1^{x_1} g_2^{x_2}$ and $h_2 = g_3^{x_1} g_4^{x_3}$. We denote this protocol by $PK\{(\chi_1, \chi_2, \chi_3) : (\chi_1, \chi_2) = \text{rep}_{(g_1,g_2)} h_1 \wedge (\chi_1, \chi_3) = \text{rep}_{(g_3,g_4)} h_2\}$. It can easily be extended towards proofs of equality of arbitrary exponents using arbitrary base tuples.

- **Knowledge of discrete logarithms constituting successive powers [9, Chapter 3]:** Let $h_1, \ldots, h_n, g_1$ and g_2 be values in G_q. Alice can prove knowledge of values $x, y_1, \ldots, y_n \in \mathbb{Z}_q$ such that $h_i = g_1^{x^i} g_2^{y_i}$ for $i \in \{1, \ldots, n\}$. We denote this protocol by $PK\{(\chi, \gamma_1, \ldots, \gamma_n) : (\chi, \gamma_1) = \text{rep}_{(g_1,g_2)} h_1 \wedge\ (\chi^2, \gamma_1) = \text{rep}_{(g_1,g_2)} h_2 \wedge\ \ldots\ \wedge\ (\chi^n, \gamma_n) = \text{rep}_{(g_1,g_2)} h_n\}$.

- **Knowledge of a discrete logarithm unequal to zero [4, Chapter 3]:** Let h be a value in G_q. Alice can demonstrate to Bob that she knows a representation (x_1, x_2) of h w.r.t. base tuple $(g_1, g_2) \in (G_q)^2$, such that $x_1 \neq 0$. We denote this protocol by $PK\{(\chi_1, \chi_2) : (\chi_1, \chi_2) = \text{rep}_{(g_1,g_2)} h \wedge\ \chi_1 \neq 0\}$. Brands calls this a NOT proof.

- **AND connections:** All previous formulae can be combined by "AND" connectives. Given formulae $F_1(x_{1,1}, \ldots, x_{1,l_1}), \ldots, F_n(x_{n,1}, \ldots, x_{n,l_n})$ about secrets $(x_{i,1}, \ldots, x_{i,l_i})$ $(i = 1, \ldots, n)$, we denote this protocol by $PK\{(\chi_{1,1}, \ldots, \chi_{n,l_n}) : F_1(\chi_{1,1}, \ldots, \chi_{1,l_1}) \wedge \ldots \wedge F_n(\chi_{n,1}, \ldots, \chi_{n,l_n})\}$.

Under the discrete logarithm assumption, all protocols are perfect honest-verifier zero-knowledge and perfect witness-indistinguishable. They can be made concurrent zero-knowledge at virtually no overhead by using techniques proposed by Damgård [21].

We now briefly review the most important properties of Brands' credential system for our purposes. The considered issuing protocol is that of Brands [4, Section 4.5.2]. Most of the security and privacy results for our system will be immediate consequences of one or more of these properties.

Proposition 1. *The digital credential system of Brands [4] satisfies the following properties.*

1. *If an honest user Alice accepts the credential issuing protocol, she retrieves a credential secret key $(x_1, \ldots, x_l, s)$, a corresponding public key h and a signature sign(h), such that $(h, \text{sign}(h))$ is uniformly distributed over the set $\{(h, \text{sign}(h)) | h \in G_q \setminus \{1\}\}$.*

2. *Under the discrete log assumption, it is infeasible to existentially forge a credential.*

3. *For any credential public key h and signature sign(h), for any tuple $(x_1, \ldots, x_l)$, and for any view of* CA *on a credential issuing protocol in which $p = g_1^{x_1} \ldots g_l^{x_l}$ is used as initial input (with $(x_1, \ldots, x_l)$ known by* CA*), there is exactly one set of random choices that an honest user Alice could have made during the execution of this issuing protocol such that she would have output a credential containing both h and sign(h).*

4. *Let h be a valid credential public key. Under the discrete log assumption and provided that $s \neq 0$, proving knowledge of a representation $(x_1^*, \ldots, x_l^*, s^*)$ of h_0^{-1} w.r.t. $(g_1, \ldots, g_l, h)$ is equivalent to proving knowledge of a valid secret key $(x_1^* s, \ldots, x_l^* s, s)$ corresponding to h. Moreover, the relation $s^* = -s^{-1}$ holds.*

5. *Consider any number of arbitrarily interleaved executions of a showing protocol with a computationally unbounded Bob in which Alice only discloses formulae about the attributes that do not contain s, and in which she uses only proofs of knowledge that are statistically witness-indistinguishable. Whatever information Bob can compute about the credential attributes, he can also compute using merely his a priori information (i.e., without engaging in showing protocol executions) and the status of the requested formulae.*

Assumption 1. *Under the discrete log assumption, if a computationally bounded attacker $\mathcal{A}$, after engaging in an execution of the issuing protocol in which $p = g_1^{x_1^*} \ldots g_l^{x_l^*}$ is used as input, outputs a valid credential containing a secret key $(x_1, \ldots, x_l, s)$, then $(x_1, \ldots, x_l, s) = (x_1^* s, \ldots, x_l^* s, s)$ with overwhelming probability. This assumption remains valid even when polynomially many executions of the issuing protocol are arbitrarily interleaved.*

2.2 Comparison to other credential systems

Before moving on the new system, we compare Brands' system with the CL-based systems of Camenisch and Lysyanskaya [12, 13]. The core of these systems are a new type of signature scheme for signing blocks of messages. Such a scheme, next to providing signature functionality, enables additional protocols for the retrieval of a signature on committed values and for the demonstration of signature possession in zero knowledge. The main asset of this construction is the fact that a credential can be shown unlinkably multiple times. In addition, all pseudonyms based on the same credential can be revoked by simply revoking the credential.

We first compare the complexity of Brands' scheme [5, Section 4.5.2] with the optimized CL-RSA system [12] as described in [2] and the CL-DL system of [13]. This evaluation considers communication sizes and workloads in the number of exponentiations and bilinear pairings. (Multiplications and additions are neglected, as their demand on computational resources is many orders of magnitude smaller.) Table 1 gives an approximation of the workload and communication sizes for the different systems. The communication size is approximated by the number of small values

that is to be transmitted. The workload is approximated by the number of small exponentiations. We assume a small value to have a bitlength equal to a credential attribute. A small exponentiation is an exponentiation where the exponent is a small value. We evaluate an issuing protocol for a credential with l user-chosen attributes which are unknown to the issuer and a showing protocol in which no properties of the attributes are demonstrated. All exponentiations are reduced to small exponentiations using the guideline that an x-bit exponentiation roughly compared to x/y y-bit exponentiations.

In our comparison, we adopt the following assumptions concerning the relations between the parameters in the different schemes. The plausibility of these assumptions is illustrated by means of example parameter sizes.

- Using the notation of Brands [5, Section 4.5.2], we adopt Brands' scheme for a subgroup discrete logarithm construction with parameters p and q such that $6|q| \approx |p|$. For example, $|q| = 160/256$ and $|p| = 1024/1600$.
- Using the notation of Bangerter et al. [2], we assume a CL-RSA scheme with parameters $\ell_s, \ell_c, \ell_m, \ell_e$ and ℓ_n such that $\ell_m \approx \ell_e$, $\ell_m \approx \ell_c + \ell_s$ and $\ell_n \approx 6\ell_m$. For example, $\ell_m = 160/256$, $\ell_c = 160/160$, $\ell_s = 80/80$, $\ell_e = 163/259$ and $\ell_n = 1024/1600$. Note that for $\ell_m = 160$, we have that $\ell_c + \ell_s \approx 1.5\ell_m$. This does not significantly affect our final approximations.
- Using the notation of Camenisch et al. [13]. We assume a CL-DL scheme based on a bilinear mapping over elliptic curves, such that $\log_2 |\mathsf{G}| \approx 6|q|$. For example, $|q| = 160/256$ and $|\mathsf{G}| = 2^{1024}/2^{1600}$. In addition, we assume the complexity of a bilinear pairing to be competitive to that of a small exponentiation.

Compared to the CL-RSA scheme, Brands' credentials are cheaper in all aspects. With respect to CL-DL credentials, they are cheaper to show and only slightly more expensive for Alice to retrieve. It may be argued that Brands' credentials cannot be shown unlinkably. However, this property can be simulated by using multiple copies of Brands' credentials. One additional credential for identical attributes occupies approximately the same amount of space as 9 small values and can be retrieved requiring 7 small exponentiations from Alice. A simple calculation shows that for l attributes, approximately $l/5$ of Brands credentials occupy the same amount of space as one CL-DL credential. Additionally, the retrieval of $l/7 + 2$ of Brands credentials costs roughly as much for Alice as the retrieval of one CL-RSA credential.

Security of Brands' scheme is based on the discrete logarithm assumption in groups of prime order. Security of the CL-RSA and CL-DL schemes are based on respectively the strong RSA assumption and the LRSW-assumption [27]. Both are stronger assumptions and hence easier to break than the DL assumption.

The CL-DL scheme is based on elliptic curves and bilinear mappings. The bilinear mapping must be constructed as such to provide efficient computations as well as adequate security for the discrete logarithm problem. As a consequence, the system's key setup should be chosen very carefully; not any random curve will do. In contrast, the CL-RSA scheme as well as Brands' system are very flexible in their choice of key setup.

Provided the issuer's key-setup is performed correctly, both Brands' system and the CL-DL scheme guarantee unconditional privacy for the user. In Brands' system, this key-setup can easily be checked by ensuring that p and q are prime and that $q|p-1$. In contrast, the CL-RSA scheme

Table 1: Approximation of complexity and communication sizes for Brands and CL-based credentials. Sizes are expressed in the number of small values, while workloads are expressed in the number of small exponentiations.

size of credentials	
Brands	$l + 10$
CL-RSA	$l + 15$
CL-DL	$3l + 4$

issuing protocol					
	#expon. Alice		**#expon. *CA***		**comm.**
	offline	online	offline	online	
Brands	$2l + 6$	3	2	$l + 3$	$l + 34$
CL-RSA	$3l + 14$	7	-	$2l + 21$	$2l + 43$
CL-DL	$2l + 2$	-	$2l + 3$	$l + 3$	$3l + 7$

	#expon. Alice		**#expon. Bob**		**comm.**
	offline	online	offline	online	
Brands	$l + 2$	-	-	$l + 7$	$l + 23$
CL-RSA	$2l + 18$	-	-	$2l + 9$	$2l + 24$
CL-DL	$4l + 8$	-	-	$6l + 8$	$3l + 12$

provides statistical privacy only. Its procedure for checking the issuer's key-setup is more complicated and requires a signed proof of knowledge constructed with binary challenges. In the setting described above, this proof's construction requires approximately $6(l + 1)\ell_c$ small exponentiations while its verification requires approximately $6.5(l + 1)\ell_c$ small exponentiations.

Finally, Brands' credentials can easily be incorporated into wallets-with-observers such that all inflow and outflow is prevented. It is not clear whether and how this can be achieved for CL-based credentials.

From this comparison, it is clear that both systems have their specific advantages and disadvantages. CL-based systems allow for multi-show unlinkability. On the downside, the security assumptions are less well-accepted and the showing of credentials becomes more expensive. Brands' scheme, in turn, is based on well-accepted security assumptions and achieves unconditional privacy and efficient showing protocols. Multi-show unlinkability, however, is not achieved.

Because of its benefits in the efficient showing of credentials, we have opted for Brands' system. As argued previously, the option of multi-show unlinkability can be simulated by using multiple copies of the same credentials. The additional cost on the issuing protocol and on the storage requirements is relatively low.

3 The new system

We are now prepared to describe our system and to prove its privacy and security. The principal parties in our system are a user $\mathcal{U}$, an identity provider $\mathcal{IP}$ and l service providers $\mathcal{S}_i$ $(i = 1, \ldots, l)$. $\mathcal{U}$ retrieves her digital pseudonyms from $\mathcal{IP}$ and uses them to authenticate herself to the service providers. In the remainder of the paper, the notation $\mathcal{S}_i$ refers both to the service provider as well as to the provided service.

In order to obtain her pseudonyms, $\mathcal{U}$ contacts $\mathcal{IP}$ and both parties engage into a pseudonym retrieval protocol. As private output of this protocol, $\mathcal{U}$ retrieves a set of l unlinkable pseudonyms, such that each of them encodes the same random tuple $(d_1, \ldots, d_l)$. To access service $\mathcal{S}_i$, $\mathcal{U}$ authenticates herself with her i-th pseudonym and additionally discloses the encoded value d_i. In addition, $\mathcal{U}$ must prove for each $j \in \{1, \ldots, l\}$ in zero knowledge that value d_j encoded into her credential is not on blacklist L_j. Blacklists are formed as follows: for any user $\mathcal{U}$, if $\mathcal{U}$ abuses service $\mathcal{S}_j$ then $\mathcal{U}$'s credential value d_j is added to public blacklist L_j.

We now describe in detail the system setup and the protocols for pseudonym retrieval, pseudonym registration, and subsequent authentication to service providers in a manner that prevents blacklisted (yet unknown) users from gaining service access.

3.1 System setup

To set up the system, identity provider $\mathcal{IP}$ decides on a group G_q of prime order q, in which the discrete logarithm problem is assumed to be hard. She generates a keypair (sk, pk) suitable for issuing digital credentials containing $l+1$ attributes. We assume $(g_1, \ldots, g_{l+1}, h_0) \in G_q$ to be part of $\mathcal{IP}$'s public key pk. Credential public keys are of the form $g_1^{x_1} \ldots g_l^{x_l} g_{l+1}^t h_0^s$, where $(x_1, \ldots, x_l, t, s)$ is the secret key of the credential.

Additionally, each service provider $\mathcal{S}_i$ sets up and publishes an empty list L_i that can only be modified by $\mathcal{S}_i$. $\mathcal{S}_i$ also publishes values $a_i, b_i \in_{\mathcal{R}} G_q$ where $z_i = \log_{a_i} b_i$ is privy to $\mathcal{S}_i$.

A pseudonym is a tuple $(P, sign(P))$. Here, $P \neq 1$ is a credential public key. User $\mathcal{U}$ is said to be the owner of $(P, sign(P))$ if she knows P's secret key $(x_1, \ldots, x_l, t, s)$.

3.2 Pseudonym retrieval

Before retrieving a set of pseudonyms, user $\mathcal{U}$ authenticates her identity to $\mathcal{IP}$ and provides her with any required certifications. Assuming $\mathcal{U}$ meets the enrollment requirements of $\mathcal{IP}$, the following protocol is executed:

1. $\mathcal{U}$ generates random values $d_{(1,1)}, \ldots, d_{(1,l)}, e \in_{\mathcal{R}} \mathbb{Z}_q$ and sends $p_1 = (\prod_{i=1}^{l} g_i^{d_{(1,i)}}) g_{l+1}^e$ to $\mathcal{IP}$.

2. $\mathcal{IP}$ retrieves p_1, picks l random values $d_{(2,1)}, \ldots, d_{(2,l)} \in_{\mathcal{R}} \mathbb{Z}_q$ and computes $p = p_1 \prod_{i=1}^{l} g_i^{d_{(2,i)}}$. She sends $d_{(2,1)}, \ldots, d_{(2,l)}$ to $\mathcal{U}$.

3. Upon receipt, $\mathcal{U}$ creates $d_i = d_{(1,i)} + d_{(2,i)}$ for $i = 1, \ldots, l$ and $p = (\prod_{i=1}^{l} g_i^{d_i}) g_{l+1}^e$.

4. $\mathcal{IP}$ and $\mathcal{U}$ perform l instances of the credential issuing protocol of Brands [4, Section 4.5.2], using p as initial input. As a result, $\mathcal{U}$ obtains l tuples $(P_i, sign(P_i))$, and l values $s_i \in \mathbb{Z}_q$, such that $P_i = (ph_0)^{s_i}$ for $i = 1, \ldots l$. (All protocol executions may be done in parallel.)

Steps 1 to 3 are used to create a random tuple $(d_1, \ldots, d_l) \in_{\mathcal{R}} (\mathbb{Z}_q)^l$, such that neither $\mathcal{U}$ nor $\mathcal{IP}$ can control its final value. Note that, because of the random selection of e by $\mathcal{U}$, this tuple remains unconditionally hidden from $\mathcal{IP}$. Based on $(d_1, \ldots, d_l, e)$, a list of l pseudonyms $(P_i, sign(P_i))$ $(1 \leq i \leq l)$ is then created for $\mathcal{U}$ during step 4.

Provided $\mathcal{U}$ has followed the protocol, the resulting pseudonyms are unconditionally unlinkable and untraceable. $\mathcal{U}$ can also compute a secret key $(d_1 s_i, \ldots, d_l s_i, e s_i, s_i)$ for each pseudonym $(P_i, sign(P_i))$. As a result of Assumption 1, the same tuple $(d_1, \ldots, d_l, e)$ is encoded into all of these secret keys, even when $\mathcal{U}$ tries to cheat during the retrieval protocol. We will refer to $(d_1, \ldots, d_l, e)$ as the tuple encoded into P_i, and to d_j $(j \in \{1, \ldots, l\})$ as the j-th value encoded into P_i.

Proposition 2. *Under the discrete logarithm assumption in G_q and for fixed values $d \in \mathbb{Z}_q$ and $i \in \{1, \ldots, l\}$. For any attacker $\mathcal{A}$ engaging into a pseudonym retrieval protocol with $\mathcal{IP}$ and as such retrieving a valid pseudonym $(P, sign(P))$. With negligible probability, value d is the i-th value encoded into P.*

Hence, it is infeasible to create a pseudonym encoding one or more values that are the same as the values encoded into another user's pseudonym.

3.3 Pseudonym registration with service provider

To register pseudonym $(P_i, sign(P_i))$ with service $\mathcal{S}_i$, user $\mathcal{U}$ shows $(P_i, sign(P_i))$ and discloses value d_i encoded into P_i. $\mathcal{U}$ and $\mathcal{S}_i$ then perform (possibly in signed proof mode)

$$PK\{(\delta_1, \ldots, \delta_{i-1}, \delta_{i+1}, \ldots, \delta_l, \epsilon, \varsigma) : \\ (\delta_1, \ldots, \delta_{i-1}, d, \delta_{i+1}, \ldots, \delta_l, \epsilon, \varsigma) = \mathrm{rep}_{(g_1, \ldots, g_l, g_{l+1}, P_i)} h_0^{-1}\}$$

$\mathcal{S}_i$ accepts the protocol if and only if she accepts this proof and if $P_i \neq 1$ and if $(P_i, sign(P_i))$ constitutes a valid message/signature pair. $\mathcal{S}_i$ then stores (P_i, d_i) and associates it with a new account or perhaps with a legacy account that it maintains on $\mathcal{U}$ (the latter requires a one-time legacy or out-of-band authentication step to ensure the right association is made).

As per Proposition 1 (property 4), this protocol proves Alice's ownership of $(P_i, sign(P_i))$ and proves that the disclosed value d_i is indeed the i-th value encoded into P_i. Furthermore, as a result of Proposition 1 (property 5), $\mathcal{S}_i$ cannot find out more information about the tuple $(d_1^*, \ldots, d_l^*, e^*)$ encoded into P_i, than what she can deduce from her previous knowledge and the fact that $d_i^* = d_i$.

3.4 Accessing a service

Upon having registered her pseudonym with $\mathcal{S}_i$, $\mathcal{U}$ may either disconnect and return later on to access the service, or proceed immediately. In either case, to access the service of $\mathcal{S}_i$, $\mathcal{U}$ and $\mathcal{S}_i$ engage in the following protocol, for blacklists $\{L_1, \ldots, L_l\}$ as defined earlier. In step 1 of the following protocol, $\mathcal{S}_i$ checks whether d_i belongs to her own blacklist L_i; in step 2, $\mathcal{U}$ proves that each j-th value d_j $(j \in \{1, \ldots, l\} \setminus \{i\})$ encoded into P_i does not belong to blacklist L_j.

1. $\mathcal{S}_i$ verifies if $d_i \in L_i$. If so, she aborts the protocol and rejects the user's request. (In case blacklist entries never get removed, $\mathcal{S}_i$ may at this point invalidate $\mathcal{U}$'s digital pseudonym.) If not, she proceeds to step 2.

2. If all of the blacklists L_j for $j \in \{1, \ldots, i-1, i+1, \ldots, l\}$ are empty, then $\mathcal{U}$ must prove knowledge to $\mathcal{S}_i$ of her pseudonym key (assuming she is not still in the pseudonym registration session with $\mathcal{S}_i$, in which case this step can be skipped); this can be done using the standard proof of knowledge of a representation, without disclosing any attributes (after all, d_i has already been disclosed and proven to be correct). If, on the other hand, not all of those blacklists are empty, then the following steps are executed for each $j \in \{1, \ldots, i-1, i+1, \ldots, l\}$ for which L_j is not empty:

   (a) Both $\mathcal{U}$ and $\mathcal{S}_i$ look up $L_j = \{y_1, \ldots, y_n\}$. They set $m = \lceil \sqrt{n} \rceil$ for $n = |L_j|$ and compute the coefficients $a_{i,j} \in \mathbb{Z}_q$ $(i, j \in \{1, \ldots, m\})$ of the following polynomials in $\mathbb{Z}_q$.
   $p_1(x) = (x - y_1)(x - y_2) \ldots (x - y_m) = a_{1,m}x^m + a_{1,m-1}x^{m-1} + \ldots + a_{1,0}$
   $p_2(x) = (x - y_{m+1}) \ldots (x - y_{2m}) = a_{2,m}x^m + a_{2,m-1}x^{m-1} + \ldots + a_{2,0}$
   $\vdots$
   $p_m(x) = (x - y_{(m-1)m+1}) \ldots (x - y_n) = a_{m,m}x^m + a_{m,m-1}x^{m-1} + \ldots + a_{m,0}$
   Note that $a_{m,i}$ for $i > n - m(m-1)$ are equal to zero.

   (b) $\mathcal{U}$ chooses random values $r_1, \ldots, r_m \in_{\mathcal{R}} \mathbb{Z}_q$ and generates values $C_k = a_i^{d_j^k} b_i^{r_k}$ for all values $k \in \{1, \ldots, m\}$. She also computes $v_k = p_k(d_j)$, $w_k = a_{k,m}r_m + \ldots + a_{k,2}r_2 + a_{k,1}r_1$ and $C_{v_k} = a_i^{v_k} b_i^{w_k}$ for $k = 1, \ldots, m$. All values C_k, C_{v_k} $(k \in \{1, \ldots, m\})$ are sent to $\mathcal{S}_i$.

   (c) $\mathcal{S}_i$ receives C_k, C_{v_k} for all $k \in \{1, \ldots, m\}$ and checks for each $k \in \{1, \ldots, m\}$ if $C_{v_k} = (C_m)^{a_{k,m}} (C_{m-1})^{a_{k,m-1}} \ldots (C_1)^{a_{k,1}} a_i^{a_{k,0}}$. If this check fails, $\mathcal{S}_i$ aborts and rejects $\mathcal{U}$'s request.

   (d) Next, the following proof of knowledge is executed. The proof makes use of the techniques described in Section 2. $\mathcal{S}_i$ accepts only if she accepts the proof.

$$PK\{(\delta_1, \ldots, \delta_l, \epsilon, \varsigma, \rho_1, \ldots, \rho_m, \upsilon_1, \ldots, \upsilon_m, \omega_1, \ldots, \omega_m) :$$

$$(\delta_1, \ldots, \delta_j, \ldots, \delta_l, \epsilon, \varsigma) = \text{rep}_{(g_1, \ldots, g_{l+1}, P_i)} h_0^{-1} \wedge \tag{1}$$

$$(\delta_j, \rho_1) = \text{rep}_{(a_i, b_i)} C_1 \ \wedge \ \ldots \ \wedge (\delta_j^m, \rho_m) = \text{rep}_{(a_i, b_i)} C_m \wedge \tag{2}$$

$$(\upsilon_1, \omega_1) = \text{rep}_{(a_i, b_i)} C_{v_1} \ \wedge \ \upsilon_1 \neq 0 \ \wedge \ \ldots \ \wedge$$

$$(\upsilon_m, \omega_m) = \text{rep}_{(a_i, b_i)} C_{v_m} \ \wedge \ \upsilon_m \neq 0\} \tag{3}$$

A clarification of what happens in step 2 is in order. In step 2a, elements in L_j are divided into subsets $L_{j,k}$ of size $m = \lceil \sqrt{|L_j|} \rceil$. The polynomials $p_k(.)$ $(k = 1, \ldots, m)$ are then constructed such as to contain only the elements of $L_{j,k}$ as roots. For each k in $\{1, \ldots, m\}$, values C_k and C_{v_k} are constructed in step 2b. C_k hides a power d_j^k of d_j, while C_{v_k} hides the mapping $p_k(d_j)$ of d_j. Note that

$$(C_m)^{a_{k,m}} (C_{m-1})^{a_{k,m-1}} \ldots (C_1)^{a_{k,1}} a_i^{a_{k,0}} = a_i^{a_{k,m}d_j^m + \ldots + a_{k,1}d_j + a_{k,0}} b_i^{a_{k,m}r_m + \ldots + a_{k,1}r_1} = C_{v_k}.$$

In step 2d, $\mathcal{U}$ proves that the values hidden in $C_1, \ldots, C_k$ are consecutive powers of the same value d_j (equation 2), that this value d_j is also the j-th value encoded into P_i (equation 1), and that values $p_k(d_j)$ hidden in C_{v_k} for $k = 1, \ldots, m$ differ from zero (equation 3). The latter proves that d_j is not a root of any of the polynomials p_k $(k \in \{1, \ldots, l\})$, and hence does not belong to L_j.

Proposition 3. *Under the discrete logarithm assumption, provided that $P_i \neq 1$, the subprotocol in step 2 is a perfect honest-verifier zero-knowledge proof that for all $j \in \{1, \ldots, l\} \setminus \{i\}$, the j-th value encoded into P_i does not belong to blacklist L_j.*

Here we give an outline of the proof of this proposition: the full proof will appear in the final version of this paper. Completeness of the protocol is easy to verify. It is perfect honest-verifier zero-knowledge due to the zero-knowledge properties of the proof in step 2d. (The simulator picks m random values $C_1, \ldots, C_m$ in G_q, and computes $C_{\upsilon_1}, \ldots, C_{\upsilon_m}$ as specified by the verification check in step 2c. Step 2d is simulated as in the protocols of Brands.)

To demonstrate soundness, we employ the standard knowledge extractor $\mathcal{K}$ for the protocols in Section 2. Let the output of this extractor be a tuple $(\delta_1, \ldots, \delta_l, \epsilon, \varsigma, \rho_1, \ldots, \rho_m, \upsilon_1, \ldots, \upsilon_m, \omega_1, \ldots, \omega_m)$ satisfying equations 1, 2 and 3 in step 2d. Under the discrete log assumption, this must be the same tuple as the witness-tuple known by $\mathcal{U}$. If this were not the case, $\mathcal{U}$ and $\mathcal{K}$ together could break the DL assumption by finding two different representations of at least one of the values h_0^{-1}, C_k or C_{υ_k} ($k \in \{1, \ldots, m\}$). We will now show that $\upsilon_k = p_k(\delta_j)$ and $\omega_k = a_{k,m}\rho_m + \ldots + a_{k,1}\rho_1$ for each $k \in \{1, \ldots, m\}$. As step 2c succeeds, we have the following equations.

$$\begin{aligned} C_{\upsilon_k} &= (C_m)^{a_{k,m}} (C_{m-1})^{a_{k,m-1}} \ldots (C_1)^{a_{k,1}} a_i^{a_{k,0}} \\ &= (a_i^{\delta_j^m} b_i^{\rho_m})^{a_{k,m}} \ldots (a_i^{\delta_j} b_i^{\rho_1})^{a_{k,1}} a_i^{a_{k,0}} \\ &= a_i^{a_{k,m}\delta_j^m + \ldots + a_{k,1}\delta_j + a_{k,0}} b_i^{a_{k,m}\rho_m + \ldots + a_{k,1}\rho_1} \\ &= a_i^{p_k(\delta_j)} b_i^{a_{k,m}\rho_m + \ldots + a_{k,1}\rho_1} \end{aligned}$$

Hence, $\mathcal{K}$ knows two representation (υ_k, ω_k) and $(p_k(\delta_j), a_{k,m}\rho_m + \ldots + a_{k,1}\rho_1)$ of C_{υ_k}. Under the discrete log assumption, it must be that both representations are the same.
As the roots of $p_k(.)$ ($k = 1, \ldots, m$) are exactly the elements in L_j, the fact that $\upsilon_k \neq 0$ for all $k \in \{1, \ldots, m\}$ implies that d_j is not on the list L_j.

The fact that δ_j is indeed the j-th value encoded into P_i follows easily from Proposition 1 (property 4) and the requirement in Section 3.3 that $P_i \neq 1$.

The proposition now easily results from the techniques of Damgård [21] for concurrent zero-knowledge proofs.

Proposition 4. *Whatever information about $(d_1, \ldots, d_l)$ in P that a computationally unbounded service provider $\mathcal{S}_i$ can compute after any number of arbitrarily interleaved executions of the protocol in step 2, for lists L_j and for the same or different pseudonyms $(P, sign(P))$ with $P \neq 1$, she can also compute using merely her a-priori information and the shown formulae $d_j \notin L_j$.*

Proof. We start by evaluating one execution of the protocol for blacklists $L_1, \ldots, L_l \subset \mathbb{Z}_q$. Let V be the view of $\mathcal{S}_i$ in the protocol and let $(d_1, \ldots, d_l)$ be any tuple satisfying the property $d_j \notin L_j$, for all $j \in \{1, \ldots, l\} \setminus \{i\}$. There are exactly $q - 1$ representations $(d_1 s, \ldots, d_l s, es, s)$ of P that encode tuple $(d_1, \ldots, d_l)$. We now show that for each of these representations $(d_1 s, \ldots, d_l s, es, s)$, there is exactly one set of random choices that a user $\mathcal{U}$ could have made such that V is $\mathcal{S}_i$'s view in a protocol in which representation $(d_1 s, \ldots, d_l s, es, s)$ is used. We prove this statement for the subprotocol defined by steps 2a - 2d for one $j \in \{1, \ldots, l\} \setminus \{i\}$. The more general claim can easily be deduced from this fact.

We first spell out in full detail the proof in protocol step 2d:

1. $\mathcal{U}$ computes $s' = -1/s$, picks random values $x_{d_1}, \ldots, x_{d_l}, x_e, x_{s'}, x_{r_1} \ldots, x_{r_m}, x_{v_1}, \ldots, x_{v_m}$, $x_{w_1}, \ldots, x_{w_{m-1}}$ and x_{w_m}, and computes $t_h = g_1^{x_{d_1}} \ldots g_l^{x_{d_l}} g_{l+1}^{x_e} P_i^{x_{s'}}$, $t_1 = a_i^{x_{d_j}} b_i^{x_{r_1}}$, $t_k = C_{k-1}^{x_{d_j}} b_i^{x_{r_k}}$ for $k \in \{2, \ldots, m\}$ and $t_{v_k} = C_{v_k}^{x_{v_k}} (1/b_i)^{x_{w_k}}$ for $k \in \{1, \ldots, m\}$. Values t_h, t_k and t_{v_k} ($k \in \{1, \ldots m\}$) are sent to $\mathcal{S}_i$.

2. $\mathcal{S}_i$ picks a random value $c \in_{\mathcal{R}} \mathbb{Z}_q$ and sends it to $\mathcal{U}$.

3. $\mathcal{U}$ computes values $s_{d_k} = cd_k + x_{d_k}$ for $k \in \{1, \ldots, l\}$, $s_e = ce + x_e$, $s_{s'} = cs' + x_{s'}$, $s_{r_1} = cr_1 + x_{r_1}$, $s_{r_k} = c(r_k - d_j r_{k-1}) + x_{r_k}$ for $k \in \{2, \ldots, m\}$, $s_{v_k} = c(1/v_k) + x_{v_k}$ and $s_{w_k} = c(w_k/v_k) + x_{w_k}$ for $k \in \{1, \ldots, m\}$. All values are computed in $\mathbb{Z}_q$. She then sends $s_{d_1}, \ldots, s_{d_l}, s_e, s_{s'}, s_{r_1}, \ldots, s_{r_m}, s_{v_1}, \ldots, s_{v_m}, s_{w_1}, \ldots, s_{w_m}$ to $\mathcal{S}_i$.

4. $\mathcal{S}_i$ accepts the protocol if $g_1^{s_{d_1}} \ldots g_l^{s_{d_l}} g_{l+1}^{s_e} P_i^{s_{s'}} (1/h_0)^{-c} = t_h$, $a_i^{s_{d_j} + z_i s_{r_1}} C_1^{-c} = t_1$, $C_{k-1}^{s_{d_j}} b_i^{s_{r_k}} C_k^{-c} = t_k$ for $k \in \{2, \ldots, m\}$ and $C_{v_k}^{s_{v_k}} a_i^{-z_i s_{w_k} - c} = t_{v_k}$ for $k \in \{1, \ldots, m\}$.

Based on this description, $\mathcal{S}_i$'s view in steps 2a - 2d is a tuple $(L_j, C_1, \ldots, C_m, C_{v_1}, \ldots, C_{v_m}, T, C, S)$ with (T, C, S) denoting the communication between $\mathcal{U}$ and $\mathcal{S}_i$ during the proof of step 2d. More precisely:

$$\begin{aligned} T &= (t_h, t_1, \ldots, t_m, t_{v_1}, \ldots, t_{v_m}) \\ C &= (c) \\ S &= (s_{d_1}, \ldots, s_{d_l}, s_e, s_{s'}, s_{r_1} \ldots, s_{r_m}, s_{v_1}, \ldots, s_{v_m}, s_{w_1}, \ldots, s_{w_m}). \end{aligned}$$

The set of random choices of user $\mathcal{U}$ is a tuple $(r_1, \ldots, r_m, X)$, with $X = (x_{d_1}, \ldots, x_{d_l}, x_e, x_{s'}, x_{r_1} \ldots, x_{r_m}, x_{v_1}, \ldots, x_{v_m}, x_{w_1}, \ldots, x_{w_m})$ denoting the set of random choices made by $\mathcal{U}$ during the execution of step 2d.

As b_i is a generator of G_q, values $r_1, \ldots, r_m$ are uniquely defined as $r_k = \log_{b_i}(C_k a_i^{-d_j^k})$ for $k \in \{1, \ldots, m\}$. Also, based on the proof protocol of step 2d and the fact that $\mathcal{U}$ is an honest user, values x_k are uniquely determined as follows.

$$\begin{aligned} x_k &= s_k - ck \mod q \text{ for } k \in \{d_1, \ldots, d_l, e, s', r_1\} \\ x_{r_k} &= s_{r_k} - c(r_k - d_j r_{k-1}) \mod q \text{ for } k \in \{2, \ldots, m\} \\ x_{v_k} &= s_{v_k} - c(1/v_k) \mod q \text{ for } k \in \{1, \ldots, m\} \\ x_{w_k} &= s_{w_k} - c(w_k/v_k) \mod q \text{ for } k \in \{1, \ldots, m\} \end{aligned}$$

Note that in these equations, values v_k and w_k ($k = 1, \ldots, m$) are uniquely determined by L_j, d_j, $r_1, \ldots, r_{m-1}$ and r_m.

Hence, the stated property holds for one execution of the protocol. It is easy to see that our argumentation remains valid, even when multiple executions of the protocol are arbitrarily interleaved. □

The following result can now easily be seen to hold, based on Propositions 1, 3 and 4.

Proposition 5. *Under the discrete log assumption, the following holds for any registered pseudonym $(P_i, sign(P_i))$ and d_i that $\mathcal{S}_i$ has accepted, assuming $\mathcal{S}_i$ accepts $\mathcal{U}$'s blacklist proof. With overwhelming probability, $\mathcal{U}$ is the owner of a valid pseudonym $(P_i, sign(P_i))$ which has not been revoked and*

for which d_i is the i-th value encoded into P_i. Furthermore, $\mathcal{S}_i$ cannot find out any more information about the values encoded into P_i than what she can deduce from her a-priori information, the fact that $(P_i, sign(P_i))$ has not been revoked and the fact that d_i is the i-th value encoded into P_i.

We also have the following result.

Proposition 6. *Given non-empty sets $D_1, \ldots, D_l \subset \mathbb{Z}_q$, for any pseudonym $(P, sign(P))$ such that P encodes a tuple $(d_1, \ldots, d_l) \in D_1 \times \ldots \times D_l$, for any view of $\mathcal{IP}$ in an execution of a retrieval protocol and for any $j \in \{1, \ldots, l\}$. There are exactly $(\prod_{i=1}^{l} |D_i|).(q-1)^{l-1} q^{2(l-1)} \neq 0$ sets of random choices that an honest user $\mathcal{U}$ could have made during the execution of this retrieval protocol, such that she would have output $(P, sign(P))$ as her j-th pseudonym.*

That is, a computationally unbounded $\mathcal{IP}$ cannot link a pseudonym $(P, sign(P))$ to its retrieval protocol, even if she would know the tuple $(d_1, \ldots, d_l)$ encoded into P. This is an immediate result of Proposition 1 (property 3) and the specifications of the credential issuing protocol ([4, Section 4.5.2]). Namely, there are exactly $\prod_{i=1}^{l}(|D_i|)$ tuples $(d_1, \ldots, d_l, e)$ such that p (and hence P) will be correctly formed. Furthermore, only 1 set of random choices remains during the j-th instance of the credential issuing protocol, and $q^2(q-1)$ sets of choices during each other instance $i \in \{1, \ldots, l\} \setminus \{j\}$.

4 Efficiency analysis

We now evaluate the complexity of the system, based on communication sizes and the number of exponentiations in G_q. (We neglect multiplications and additions, as their demand on computational resources is many orders of magnitude smaller.)

The pseudonym retrieval protocol is executed only once between $\mathcal{U}$ and $\mathcal{IP}$. It requires $\mathcal{U}$ to perform $9l + 1$ exponentiations in G_q, of which $3l + 1$ exponentiations can be precomputed. $\mathcal{IP}$ in turn performs $3l + 1$ exponentiations. A total of $2l + 2$ elements in G_q, and $3l$ elements in $\mathbb{Z}_q$ are communicated. By way of example, if we take $l = 100$ and primes p and q of 1024 and 160 bits respectively, this amounts to 901 exponentiations for $\mathcal{U}$, 301 exponentiations for $\mathcal{IP}$, and 31kB of transferred data. With regard to the practicality of the service access protocol, the following optimizations can be taken into account:

1. The proofs of knowledge of step 2d, for all $j \in \{1, \ldots, l\} \setminus \{i\}$, can be collapsed into a single proof protocol. As a result, equation 1 has to be performed only once.

2. The check in step 2c of the blacklist proof can be sped up using the batch verification techniques proposed by Bellare et al. [3]. $\mathcal{S}_i$ hereto chooses random values $o_1, \ldots, o_m$ in a set $V \subset \mathbb{Z}_q$, and checks the following equation:

$$\prod_{k=1}^{m} C_{v_k}^{o_k} \stackrel{?}{=} a_i^{\sum_{k=1}^{m} a_{k,0} o_k} \prod_{i=1}^{m} C_i^{\sum_{k=1}^{m} a_{k,i} o_k}$$

   If this check succeeds, the probability that $\mathcal{S}_i$ correctly accepts step 2c is at least $1 - 1/|V|$.

3. $\mathcal{S}_i$ can complement blacklists using whitelists. A whitelist $L'_j \subset L_j$ represents the set of values for which $\mathcal{U}$ has already passed the blacklist proof. A tuple $(L'_1, \ldots, L'_l)$ of whitelists

Table 2: Complexity measurements for the access protocol for an aggregate (delta-)blacklist of size $|L^*| = \sum_{i=1}^{l} |L_i^*|$, with $|L_1^*| = \ldots = |L_l^*|$, p a 1024-bit prime and q a 160-bit prime

$\|L^*\|$	**#expon. by $\mathcal{U}$**	**#expon. by $\mathcal{S}_i$**	**transferred data size**
1000	3270	2974	0.218 MB
10,000	8022	7132	0.542 MB
100,000	25446	22378	1.731 MB
1,000,000	79302	69405	5.403 MB

is stored, both by $\mathcal{U}$ and by $\mathcal{S}_i$, for each credential $(P, sign(P))$. Assuming the elements in L_j are ordered chronologically, it is sufficient for $\mathcal{U}$ and $\mathcal{S}_i$ to only store the last value that passed the proof. Whenever $\mathcal{U}$ requests access to a service, she merely needs to perform a blacklist proof with respect to the "delta-blacklists" $L_j^* = L_j \setminus L_j'$ for $j = \{1, \ldots, l\}$.

4. All of $\mathcal{U}$'s exponentiations can be precomputed using a slight variation of Brands' error correction factors technique [5, Section 5.4.2]. Appendix A presents a detailed description of this protocol. Note that these precomputation can be performed even before the final blacklist is known. All that is needed is an upper bound on $\sqrt{n}$ for n the size of the blacklists.

5. By employing her private value z_i, $\mathcal{S}_i$ can collapse her multi-exponentiations $a_i^x b_i^y$ into one exponentiation of the form $a_i^{x+z_i y}$.

Taking these optimizations into account, $\mathcal{U}$ must perform $8\sum_{j=1,j\neq i}^{l}(\lceil\sqrt{|L_j|}\ \rceil) + l + 2$ exponentiations in G_q, while $\mathcal{S}_i$ must perform $7\sum_{j=1,j\neq i}^{l}(\lceil\sqrt{|L_j|}\ \rceil) + 2l + 6$ exponentiations. A total of $4\sum_{j=1,j\neq i}^{l}(\lceil\sqrt{|L_j|}\ \rceil) + 3$ elements in G_q and $5\sum_{j=1,j\neq i}^{l}(\lceil\ \sqrt{|L_j|}\ \rceil) + (l + 5)$ elements in $\mathbb{Z}_q$ are communicated between the parties.

Table 4 shows the communication and computation complexity for the access protocol, using different sizes of L_j^* $(j = 1, \ldots, l)$, a prime p of 1024 bits, a prime q of 160 bits and $l = 100$.

When doing a blacklist proof for blacklist increments that have no more than about a dozen entries, it is computationally more efficient to simply repeat (in parallel) the basic NOT proof of Brands [4, Section 3.4.1] for each list entry. Beyond that, our sublinear technique is more efficient. In addition, for an aggregate list size of more that 10000 entries, our blacklisting technique requires less exponentiations from $\mathcal{U}$ than the dynamic accumulator techniques of [8, 11].

5 Extensions and variations

Abuse of any one service in practice may not necessitate banning the abuser from the entire system. In some cases, it may suffice to ban the abuser from accessing just that service. This can be accomplished by simply blacklisting the public key of the pseudonym. In others, it may suffice to ban the abuser from accessing a subset of all services. As an example, consider online discussion forums where each service is a different forum. If a user posts hate massages in one forum, say, it may be valid to ban her access to all forums on similar topics, but perhaps it is unreasonable if she could also be banned from other forums. This can easily be accomodated by giving users different

batches of pseudonyms for use at different service providers. Furthermore, users may be banned only temporarily by deleting their blacklisted numbers from the blacklists at a later stage.

The issuing protocol of Brands [4, Section 4.5.2] allows the so-called refreshing of digital credentials. This technique can be applied in a straightforward manner to our system. For example, if $\mathcal{U}$ loses the secret key of some or all of her pseudonyms, she could get a fresh set of pseudonyms with the same encoded values from the identity provider by refreshing one of her previous pseudonyms; in order to avoid linkability at this time, one pseudonym that is issued to her could be set aside solely to allow the bootstrapping of other pseudonyms with the same encoded values at later stages.

It is straightforward to extend our system by incorporating the technique of Brands [4] for revoking a user's credentials on the basis of that user's identity with the issuer. Brands' technique for blacklisting known users relies on repeating a NOT proof for each element in the issuer's blacklist, but we can apply our sublinear blacklist proof to the issuer's blacklist as well.

Our blacklist protocol of steps 2a - 2d of Section 3.4[1] requires linear complexity *only* in the number of multiplications for calculating coefficients $a_{i,j}$. The number of exponentiations and the size of the communication is sublinear in the length of the blacklist. More precisely, $\mathcal{U}$ performs $8\lceil\sqrt{|L_j|}\rceil$ exponentiations in G_q and $\mathcal{S}_i$ performs $7\lceil\sqrt{|L_j|}\rceil + 3$ exponentiations. A total of only $4\lceil\sqrt{|L_j|}\rceil$ elements in G_q and $5\lceil\sqrt{|L_j|}\rceil + 2$ elements in $\mathbb{Z}_q$ are communicated.

The protocol can also easily be transformed in an equally efficient protocol for proving than an element is on a whitelist. All that is needed is the transformation of equation 3 in step 2d to an equation of the form $((0,\omega_1) = \mathrm{rep}_{(a_i,b_i)}C_{v_1} \ \vee \ \ldots \ \vee \ (0,\omega_m) = \mathrm{rep}_{(a_i,b_i)}C_{v_m})$

Our blacklisting technique can readily be adapted to fit any homomorphic commitment scheme for which similar basic zero-knowledge proofs are available. Among others, it can be used with the RSA-based commitment scheme of brands [5, Section 2.3.3] and the integer commitment scheme of Damgård and Fujisaki [22]. Note that the latter does not support Brands' NOT-proof for proving that a discrete logarithm differs from 0. Instead, the NOT relation must be demonstrated by proving a statement $[(x \geq 1) \vee (x \leq -1)]$. This can be achieved in constant time using well-known techniques [26, 20]. In both cases, the resulting zero-knowledge blacklisting proofs require $O(|L|^{1/2})$ exponentiations from both parties and $O(|L|^{1/2})$ communicated values.

Finally, we note that our blacklist technique has applications beyond credential mechanisms. For example, it can be used to enable membership revocation in group signature schemes [19, 1] and in identity escrow schemes [25].

References

[1] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO*, pages 255–270, 2000.

[2] Endre Bangerter, Jan Camenisch, and Anna Lysyanskaya. A cryptographic framework for the controlled release of certified data. In *Twelfth International Workshop on Security Protocols*, 2004.

[3] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In *EUROCRYPT*, pages 236–250, 1998.

[1] Equation 1 of step 2d can be omitted for a proof that $d \notin L_j$ without d having to be encoded into a credential.

[4] S. A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, Cambridge, MA, USA, 2000. Available for free download from `http://www.credentica.com/the_mit_pressbook.php`.

[5] S. A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, Cambridge, MA, USA, 2000. Available for free download from `http://www.credentica.com/the_mit_pressbook.php`.

[6] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *ACM Conference on Computer and Communications Security*, pages 132–145, 2004.

[7] Ernest F. Brickell, Peter Gemmell, and David W. Kravitz. Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In *SODA*, pages 457–466, 1995.

[8] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT*, pages 93–118, 2001.

[9] Jan Camenisch. *Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem*. PhD thesis, ETH Zurich, 1998. Reprint as vol. 2 of *ETH Series in Information Security and Cryptography*, ISBN 3-89649-286-1, Hartung-Gorre Verlag, Konstanz, 1998.

[10] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT*, pages 302–321, 2005.

[11] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO*, pages 61–76, 2002.

[12] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN*, pages 268–289, 2002.

[13] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO*, pages 56–72, 2004.

[14] Jan Camenisch, Ueli M. Maurer, and Markus Stadler. Digital payment systems with passive anonymity-revoking trustees. *Journal of Computer Security*, 5(1):69–90, 1997. Abbridged version in: Computer Security - ESORICS 96, Vol. 1146, pages 33-43, Springer-Verlag.

[15] Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO*, pages 126–144, 2003.

[16] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, pages 199–203, 1982.

[17] David Chaum. Blind signature system. In *CRYPTO*, page 153, 1983.

[18] David Chaum and Torben P. Pedersen. Wallet databases with observers. In *CRYPTO*, pages 89–105, 1992.

[19] David Chaum and Eugène van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[20] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO*, pages 174–187, 1994.

[21] Ivan Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In *EUROCRYPT*, pages 418–430, 2000.

[22] Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, pages 125–142, 2002.

[23] George I. Davida, Yair Frankel, Yiannis Tsiounis, and Moti Yung. Anonymity control in e-cash systems. In *Financial Cryptography*, pages 1–16, 1997.

[24] Markus Jakobsson and Moti Yung. Distributed "magic ink" signatures. In *EUROCRYPT*, pages 450–464, 1997.

[25] Joe Kilian and Erez Petrank. Identity escrow. In *CRYPTO*, pages 169–185, 1998.

[26] Helger Lipmaa. Statistical zero-knowledge proofs from diophantine equations. Cryptology ePrint Archive, Report 2001/086, 2001. `http://eprint.iacr.org/`.

[27] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In *Selected Areas in Cryptography*, pages 184–199, 1999.

[28] Lan Nguyen. Accumulators from bilinear pairings and applications. In *CT-RSA*, pages 275–292, 2005.

[29] Liberty Alliance Project. Liberty id-ff architecture overview, version 1.2. 2005. Available at `http://www.projectliberty.org/specs/draft-liberty-idff-arch-overview-1.%2-errata-v1.0.pdf`.

[30] Markus Stadler, Jean-Marc Piveteau, and Jan Camenisch. Fair blind signatures. In *EUROCRYPT*, pages 209–219, 1995.

A Access protocol based on precomputations

By a slight adaptation of the access protocol in Section 3.4, all of $\mathcal{U}$'s exponentiations can be precomputed. The additional cost imposed by this adaptation is minimal: some additions in $\mathbb{Z}_q$ and the transfer of $2\sum_{j=1,j\neq i}^{l}\lceil\sqrt{|L_j|}\,\rceil$ elements in $\mathbb{Z}_q$. We now describe the optimized protocol in detail. For ease of representation, we assume $l = 2$, such that only one blacklist $L_j (j \neq i)$ must be checked. In addition, we assume $\mathcal{U}$ to know $|L_j|$ for all $j \neq i$. In practice, an upper bound on these values suffices.

Precomputation phase. Let $m = \lceil\sqrt{|L_j|}\,\rceil$ for $j \neq i$. Before engaging into the actual protocol, $\mathcal{U}$ chooses random values $x_{d_1}, \ldots, x_{d_l}, x_e, x_{s'} \in_{\mathcal{R}} \mathbb{Z}_q$ and $z_{k,1}, z_{k,2}, b_{k,1}, b_{k,2}, r_k, x_{r_k} \in_{\mathcal{R}} \mathbb{Z}_q$ for all $k \in \{1, \ldots, m\}$. $\mathcal{U}$ then computes $C_k = a_i^{d_j^k} b_i^{r_k}$ and $C'_{v_k} = a_i^{z_{k,1}} b_i^{z_{k,2}}$ for all $k \in \{1, \ldots, m\}$. She also computes $t = g_1^{x_{d_1}} \ldots g_l^{x_{d_l}} g_{l+1}^{x_e} P_i^{x_{s'}}$, $t_1 = a_i^{x_{d_j}} b_i^{x_{r_1}}$, $t_k = C_{k-1}^{x_{d_j}} b_i^{x_{r_k}}$ for all $k \in \{2, \ldots, m\}$ and $t_{v_k} = a_i^{b_{k,1}} b_i^{b_{k,2}}$ for all $k \in \{1, \ldots, m\}$. All values are stored by $\mathcal{U}$.

Access protocol.

1. This step is identical to step 1 of Section 3.4.
2. The considerations for step 2 are identical as in Section 3.4. For a non-empty blacklist L_j with $j \neq i$, the following steps are are executed.
   (a) This step is identical to step 2a of Section 3.4.
   (b) As before, $\mathcal{U}$ computes $v_k = p_k(d_j)$ and $w_k = a_{k,m}r_m + \ldots + a_{k,1}r_1$ for all $k \in \{1, \ldots, m\}$. In addition, she also computes $e_{k,1} = v_k - z_{k,1} \mod q$ and $e_{k,2} = w_k - z_{k,2} \mod q$ for all $k \in \{1, \ldots, m\}$. Note that $C'_{v_k} a_i^{e_{k,1}} b_i^{e_{k,2}} = a_i^{v_k} b_i^{w_k}$. Values $C_k, C'_{v_k}, e_{k,1}$ and $e_{k,2}$ for all $k \in \{1, \ldots, m\}$ are sent to $\mathcal{S}_i$.
   (c) $\mathcal{S}_i$ receives $C_k, C'_{v_k}, e_{k,1}, e_{k,2}$ for all $k \in \{1, \ldots, m\}$ and checks for each $k \in \{1, \ldots, m\}$ if $C'_{v_k} = (C_m)^{a_{k,m}} (C_{m-1})^{a_{k,m-1}} \ldots (C_1)^{a_{k,1}} a_i^{a_{k,0} - e_{k,1} + z_i e_{k,2}}$. If this check fails, $\mathcal{S}_i$ aborts and rejects $\mathcal{U}$'s request.
   (d) $\mathcal{U}$ performs an optimized proof of knowledge. A high-level description of the proof is given below.

$$
\begin{aligned}
PK\{(\delta_1, \ldots, \delta_l, \epsilon, \varsigma, \rho_1, \ldots, \rho_m, \upsilon_1, \ldots, \upsilon_m, \omega_1, \ldots, \omega_m) : \\
\\
(\delta_1, \ldots, \delta_j, \ldots, \delta_l, \epsilon, \varsigma) = \mathrm{rep}_{(g_1, \ldots, g_{l+1}, P_i)} h_0^{-1} \ \wedge \\
(\delta_j, \rho_1) = \mathrm{rep}_{(a_i, b_i)} C_1 \ \wedge \ \ldots \ \wedge (\delta_j^m, \rho_m) = \mathrm{rep}_{(a_i, b_i)} C_m \ \wedge \\
(\upsilon_1, \omega_1) = \mathrm{rep}_{(a_i, b_i)} (C'_{v_1} a_i^{e_{1,1}} b_i^{e_{1,2}}) \ \wedge \ v_1 \neq 0 \ \wedge \ \ldots \ \wedge \\
(\upsilon_m, \omega_m) = \mathrm{rep}_{(a_i, b_i)} (C'_{v_m} a_i^{e_{m,1}} b_i^{e_{m,2}}) \ \wedge \ \upsilon_m \neq 0\}
\end{aligned}
$$

   The proof itself consists of the following steps.
   i. $\mathcal{U}$ sends values t, t_k and t_{v_k} $(k \in \{1, \ldots m\})$ to $\mathcal{S}_i$.
   ii. $\mathcal{S}_i$ picks a random value $c \in_{\mathcal{R}} \mathbb{Z}_q$ and sends it to $\mathcal{U}$.
   iii. $\mathcal{U}$ computes values $x_{v_k} = b_{k,1} v_k^{-1} \mod q$ and $x_{w_k} = b_{k,2} - w_k x_{v_k} \mod q$, for all $k \in \{1, \ldots, m\}$. Note that, as $v_k \neq 0$, v_k^{-1} is defined. $\mathcal{U}$ also computes $s' = -1/s$ and the following responses:

$$
\begin{aligned}
&s_{d_k} = c d_k + x_{d_k} \mod q \ (\forall k \in \{1, \ldots, l\}) && s_e = ce + x_e \mod q \\
&s_{v_k} = c(1/v_k) + x_{v_k} \mod q \ (\forall k \in \{1, \ldots, m\}) && s_{s'} = cs' + x_{s'} \mod q \\
&s_{w_k} = c(w_k/v_k) + x_{w_k} \mod q \ (\forall k \in \{1, \ldots, m\}) && s_{r_1} = cr_1 + x_{r_1} \mod q \\
&s_{r_k} = c(r_k - d_j r_{k-1}) + x_{r_k} \mod q \ (\forall k \in \{2, \ldots, m\})
\end{aligned}
$$

   She then sends $s_{d_1}, \ldots, s_{d_l}, s_e, s_{s'}, s_{r_1}, \ldots, s_{r_m}, s_{v_1}, \ldots, s_{v_m}, s_{w_1}, \ldots, s_{w_m}$ to $\mathcal{S}_i$.
   iv. $\mathcal{S}_i$ accepts the protocol if the following equations hold.

$$
\begin{aligned}
g_1^{s_{d_1}} \ldots g_l^{s_{d_l}} g_{l+1}^{s_e} P_i^{s_{s'}} (1/h_0)^{-c} &\stackrel{?}{=} t \\
a_i^{s_{d_j}} b_i^{s_{r_1}} C_1^{-c} &\stackrel{?}{=} t_1 \\
C_{k-1}^{s_{d_j}} b_i^{s_{r_k}} C_k^{-c} &\stackrel{?}{=} t_k \qquad (\forall k \in \{2, \ldots, m\}) \\
(C'_{v_k})^{s_{v_k}} a_i^{e_{k,1} s_{v_k} - c} (b_i)^{e_2 s_{v_k} - s_{w_k}} &\stackrel{?}{=} t_{v_k} \qquad (\forall k \in \{1, \ldots, m\})
\end{aligned}
$$