Unlinkable Attribute-Based Credentials with Practical Revocation on Smart-Cards

Jan Hajny and Lukas Malina

Department of Telecommunications
Brno University of Technology, Brno, Czech Republic
{hajny,malina}@feec.vutbr.cz
http://crypto.utko.feec.vutbr.cz

Abstract. Attribute-based credentials are cryptographic schemes designed to enhance user privacy. These schemes can be used for constructing anonymous proofs of the ownership of personal attributes. The attributes can represent any information about a user, e.g., age, citizenship or birthplace. The ownership of these attributes can be anonymously proven to verifiers without leaking any other information. The problem of existing credential schemes is that they do not allow the practical revocation of malicious or expired users when slow off-line devices (for example, smart-cards) are used for storing attributes. This prevents existing systems from being used on eIDs (electronic ID cards), employees' smart-cards or, for example, library access cards. In this paper, we propose a novel cryptographic scheme which allows both expired user revocation and de-anonymization of malicious users on commercially available smart-cards. In addition to the full cryptographic specification of the scheme, we also provide implementation results on .NET V2+ and MultOS smart-card platform.

Keywords: Revocation, privacy, anonymity, smart-cards, credentials.

1 Introduction

Attribute-based credential schemes were proposed [9] to provide more privacy during the verification of users' attributes. By using attribute-based credentials, users can anonymously prove their possession of some attributes. These attributes can represent any personal data such as age, citizenship or valid driver's license. In contrast to classical authentication, the identity of attribute holders is never released. Thus, the verification process is anonymous and with many additional features protecting users' privacy. By using attribute-based credentials in eID (electronic ID cards), citizens would be able to prove their age, citizenship or any other attribute without releasing their identity or any other private information which might be abused by verifiers. With the increasing number of electronic services, smart-card applications and the approaching European eID cards, it is necessary to provide a cryptographic scheme with as many privacy-enhancing features as possible. These features have been demanded in both U.S. and EU official documents [22,18,21]. To preserve privacy, attribute-based credential schemes should provide following features.

- **Anonymity**: user's identity stays hidden during the verification of attribute ownership.
- **Untraceability**: attribute issuers are unable to trace issued attributes and their owners.
- **Unlinkability**: verification sessions of a single user are mutually unlinkable. This feature prevents from user profiling.
- **Selective disclosure of attributes**: users can selectively choose to disclose only a subset of private attributes to verifiers. Only attributes necessary for obtaining a service are disclosed.
- **Non-transferability**: lending of credentials is prevented.
- ***Revocation***: invalid, lost, stolen or expired credentials are revocable.
- ***Malicious user identification***: although proving attribute ownership is an anonymous process, the identity of malicious users and attackers can be revealed.

In particular, the last two items are very difficult to achieve using existing schemes. In this paper, we present a scheme which supports all the features.

1.1 Example Scenario

In this paper, we use a sample demonstration scenario to provide a practical example of using attribute-based credentials. Although many examples are available (e.g., proving age on a teenage webchat, proving citizenship on borders, proving legal drinking age), we chose a municipal library scenario. In a municipal library, users are required to pay quarterly fees to be allowed to borrow books. A citizen can be issued an attribute attesting to the paid fee. The attribute is stored on citizen's eID card (a smart-card) and the card is able to compute a proof of ownership of that attribute. By using the attribute-based credentials, a user can use the eID card to rent books. He just simply waves his contactless smart-card when he leaves the library with books. The first privacy-enhancing feature, *anonymity*, assures that nobody can link the identity of the user to the type of books he reads. This protects reader's privacy because his reading habits should be considered a private information. The *untraceability* feature prevents even the library which issued the attribute from seeing what books are read by a particular person. The *unlinkability* feature prevents from the profiling of users, all visits of a single user at the library are mutually unlinkable and the library is unable to get to know what books were read subsequently. In some scenarios, user profile can be so specific that it allows de-anonymization. The unlinkability feature prevents user profiling and such de-anonymization. The *selective disclosure of attributes* lets the user show only the attribute attesting to paid quarter fees. No other information or attributes are released. The *non-transferability* feature prevents readers from lending their cards to users who don‘t pay annual fees. Finally, the *revocation* and *malicious user identification* features allow library to expel and identify readers who violate the library's rules (e.g., steal books).

2 Current State in Revocation Techniques

To provide user privacy, the verification session must be totally anonymous, unlinkable to other sessions and revealing no personal or traceable information. In contrast to this requirement, some linking is required in situations where the eID card is lost, stolen or destroyed. In these cases, the credential must be revoked so that it cannot be used any time in future. Furthermore, if the user breaks the rules of the library (e.g., steals some books or does not return books in time), there must be a mechanism for the identification of such misbehaving users. The problem of existing attribute-based credentials is that they do not support the revocation of credentials and the identification of malicious users if off-line, computationally weak devices (such as smart-cards) are used for storing attributes. We provide a short overview of revocation techniques used in existing credential systems together with reasons why we consider them impractical.

Blacklisting of Credential IDs
Some credential systems, for example U-Prove [20], use a credential identifier embedded to each transaction. The identifier is a unique and unchangeable number linked to the credential. This number can be used for revoking the credential by putting it on a blacklist. Nevertheless, this approach destroys unlinkability (the unique credential identifier creates a link among user's verification sessions). Furthermore, a credential can be revoked only if a verifier has already seen the credential before and there is no mechanism for revoking credentials by their issuers. That is why the issuer has no power to revoke invalid credentials.

Blacklisting of Secrets
The technique for blacklisting of secrets, used, for example, in [2], allows an invalid credential to be revoked by using the knowledge of secret keys used for its construction. This technique can be used in cases where secret keys of users are revealed and for example made public on the Internet. In that case, a revocation authority can create a blacklist based on these keys to prevent verifiers from accepting credentials based on leaked keys. Nevertheless, this technique works only if the user secrets are revealed. But in most cases, the secret keys never leave a protected device (like a smart-card), therefore they cannot be revoked. Moreover, lost, stolen or expired credentials (e.g., stored on a smart-card) cannot be revoked because their secret keys never become public.

Epochs of Lifetime
Epochs of lifetime are the official revocation technique of idemix [5]. Here, a credential carries an epoch of validity as a special attribute. In this case, the verifier can check whether the credential is fresh. The user is required to renew his credential for every new epoch. The disadvantage of this mechanism is that the revocation of credentials is never immediate, the revoker must wait until their expiration and the issuer must stop issuing new credentials. The second major disadvantage is that the user must periodically run the issuance protocol with the issuer (or designated entity) to update his credential.

Accumulator Proofs
The most recent technique, used for example in [6,16], allows both issuers and verifiers to revoke credentials immediately by publishing so called whitelists. In this technique, a user must provide a proof that his credential is included on a list of valid credentials. This can be done anonymously and efficiently by using so called accumulators which accumulate all non-revoked users. Most efficient techniques are based on bilinear maps [16]. The disadvantage of these solutions is that the user must update his secrets every time any other user is revoked from the system. This is not a big problem when the user uses an online computer for his verification. On the other hand, if the user uses only an offline device, like a smart-card, then he is unable to update his secrets. Therefore, the user is unable to use his credentials after some other users are revoked from the system. We aim to provide a system which can be used for everyday verification in libraries, pubs or hotels, therefore the smart-card implementation is crucial. That is the reason why we consider the accumulator-based techniques impractical.

Verifiable Encryption of Secrets
The user identity or personal secrets can be encrypted inside the credential in such a manner that only a trusted authority can do the revocation or identity disclosure using decryption. In this case, the system might be considered insecure from the perspective of a user who does not fully trust the authority. In fact, this is likely a problem since users would not welcome a scheme where a fixed third party can learn all information about their verification sessions, including their identities. In practical scenarios, the user would have no choice from multiple trusted authorities. This even more degrades his trust in such a dictated authority. Furthermore, there is a problem with unlinkability because the verifiable encryption must be randomized for each session, which might be inefficient. Revocation by verifiable encryption is mentioned in specifications [5,20] without further details and supporting infrastructure description.

2.1 Our Contribution

In this paper, we propose the first scheme with practical revocation and malicious user identification which is deployable on off-line smart-cards. By adding the support of smart-cards, we allow the application of attribute-based credentials to eIDs. Furthermore, our scheme provides scalable revocation of particular privacy-enhancing features. This allows not only the revocation of credentials but also the revocation of untraceability and the revocation of anonymity of malicious users.

- **Revocation of Credentials**:
  - **Immediate Revocation**: there is no need to wait for the credential lifetime expiration, credentials can be revoked immediately.
  - **Issuer and Verifier Driven Revocation**: Revocation is available to both attribute issuers and verifiers. Any of these entities can initiate the revocation process.

- **Verifier Local Revocation (VLR)**[4]: valid users do not have to update their credentials or download any values after some other users are revoked.
- **Computationally Efficient Revocation**: computational complexity of the user's part of verification protocol does not depend on the number of revoked users.
- **Off-line Verification**: the verification session runs between the user and verifier only. There is no need to contact other parties.

The revocation of credentials is an extremely important feature but in some cases it is not enough just to revoke users from the system. In cases where damage was done, the service providers need a technique for learning the identity of attackers to make them responsible. We add more granularity to revocation in our scheme by allowing the revocation of particular privacy-enhancing features.

- **Revocation of Unlinkability**: in non-critical policy breaches, the verifier can inspect user's past behavior by revoking unlinkability. All past sessions of a particular user can be inspected without releasing his identity.
- **Revocation of Anonymity**: in critical policy breaches, it is possible to revoke the anonymity of a user to make him responsible for his acts.

We acknowledge that these revocation features must be strongly protected against a misuse. That is the reason why we spread the ability to do revocation over more entities. In our system, the issuer, verifier and a third authority must cooperate to revoke any privacy-enhancing feature. By such distribution, we limit the probability of misusing the revocation by a single authority. To provide more security (and user trust in our system), the third party can be distributed using multi-party computations. Moreover, the user has the freedom to choose his own attribute issuer among many commercial subjects, therefore he does not have to trust a fixed designated revocation authority but rather liberally chooses an entity he trusts most.

3 Proposed Attribute-Based Credential Scheme Architecture

In this section, the novel scheme for attribute-based credentials is proposed. The entities, general communication pattern and cryptographic design of underlying protocols are described in this section. The security analysis of the scheme is provided in Section 4.

3.1 Entities

There are four entities in the proposed scheme. Some of them are in possession of secret keys. The cryptographic construction of keys and their usage are described further in Chapter 3.4.

- **Issuer - I**: the entity who issues personal attributes to Users. Issuers also cooperate during the revocation of anonymity. All Issuer, Revocation Referee and Verifier must cooperate to reveal the identity of a malicious user. Issuer is equipped with a key K_I.
- **Revocation Referee - RR**: a single entity who generates system parameters *params*, cooperates with the Issuer during the attribute issuance and with Issuer and Verifier during the revocation of anonymity. RR works as a privacy guarantee because he decides about the type of revocation (credential revocation, unlinkablity revocation or anonymity revocation) based on the evidence provided by the Verifier. No entity is able to revoke without RR, yet RR is not a fully trusted party. RR cannot revoke or reveal any private information alone, only in cooperation with I and V. RR is equipped with a secret key K_{RR}.
- **User - U**: the entity who is in possession of a smart-card with issued attributes. The user can anonymously prove the attribute ownership by using the smart-card. Each smart-card has a secret master key K_U unique for each User needed for the attribute proof generation. Additionally, a secret session key K_S is generated for each verification session. The K_S key randomizes the sessions to make them completely unlinkable.
- **Verifier - V**: the entity who verifies User's attribute ownership (sometimes called relying party). Using the transcript of the verification session and the evidence of a rule breach, Verifiers can ask RR for revocation. If RR decides that revocation is rightful, User's master key can be anonymously revoked or, in more serious cases, identity of a malicious User can be disclosed. Verifiers need only pre-shared system parameters. They do not communicate with other parties during User verification (the process runs off-line).

3.2 General Overview of Proposed Scheme

The architecture of the proposed scheme, briefly introduced in [15], is depicted in Figure 1. The scheme is composed of four protocols - `Setup`, `IssueAtt`, `ProveAtt` and `Revoke`.

- $(params, K_{RR}, K_I) \leftarrow$`Setup`$(k, l, m)$: this algorithm is run by RR and Issuer. `Setup` inputs security parameters (k, l, m) and outputs system parameters *params*. RR's private output of the protocol is the K_{RR} key and I's private output is the K_I key.
- $K_U \leftarrow$`IssueAtt`$(params, K_I, K_{RR})$: the protocol outputs User's master key K_U. The master key is needed by the User for creating the attribute ownership proof in the `ProveAtt` protocol. By using advanced cryptographic techniques, the K_U is generated in such a way that only User's smart-card learns it although both RR and Issuer must contribute data and collaborate on K_U creation.

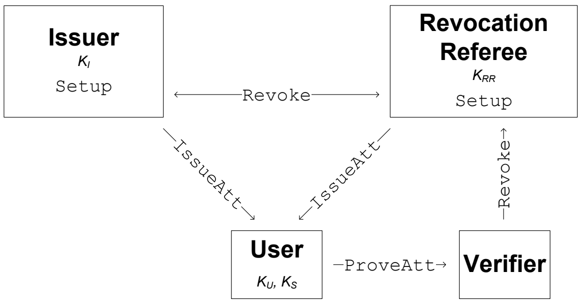

Fig. 1. Architecture of Proposed Scheme

- $proof$ ←`ProveAtt`($params, K_U$): using public system parameters and the K_U generated by the `IssueAtt` protocol, it is possible to build attribute $proof$ using the `ProveAtt`. For each $proof$, a unique session key K_S is generated by the User. The proof is anonymized and randomized by K_S. The protocol runs between the Verifier and User's smart-card. By `ProveAtt`, the User proves his ownership of attributes.
- rev ←`Revoke`($params, proof, K_{RR}, K_I$): in special cases (e.g., smart-card loss, theft or damage), the issued attributes can be revoked or even the malicious Users can be de-anonymized. In that case, the $proof$ transcript is sent by the Verifier to the RR with adequate evidence for revocation. RR evaluates the evidence and opens the $proof$ transcript using his K_{RR}. Depending on the type of revocation chosen by RR, the RR can either blacklist the attribute by publishing anonymous revocation information rev on a public blacklist or provide the Issuer with information necessary for User identification. The Issuer is then able to identify and charge the malicious User.

3.3 Used Cryptographic Primitives and Notation

The crucial building blocks of our scheme are: discrete logarithm commitments, Σ-protocols [10] for proofs of discrete logarithm knowledge and representation [8], proofs of discrete logarithm equivalence [8] and the Okamoto-Uchiyama trapdoor one-way function [19].

DL Commitments
To commit to a secret value $w \in \mathbb{Z}_q$, where q is a large prime, we use a simple computationally hiding and perfectly binding commitment. Let $p : q|p-1$ be a large prime and g a generator of order q in $\mathbb{Z}_p^*$. Then, $c = g^w \bmod p$ is a simple commitment scheme secure under the DL assumption. After publishing c, the secret w is computationally hidden (hiding property) but the committer is perfectly bound to his w (binding property) and unable to change w without changing c.

Σ-protocols

Σ-protocols [10] can be used for proving the knowledge of secrets and for proving the construction correctness without leaking additional information. We use the protocols described in [8] to prove the knowledge of a discrete logarithm (the protocol $PK\{\alpha : c = g^\alpha\}$), discrete logarithm equivalence (the protocol $PK\{\alpha : c_1 = g_1^\alpha \wedge c_2 = g_2^\alpha\}$) and discrete logarithm representation with respect to public generators (the protocol $PK\{(\alpha, \beta, \gamma) : c = g_1^\alpha g_2^\beta g_3^\gamma\}$). These protocols can be translated to full zero-knowledge protocols [11] thus they can be proven to leak no more information than intended. They can run non-interactively with computational security using [14]. With some restrictions, the protocols can be used in groups with hidden order by sending answers in $\mathbb{Z}$ [13]. Various types of proofs of knowledge and a framework for creating proofs can be found in [8].

Okamoto-Uchiyama Trapdoor One-Way Function

Let $n = r^2 s$ and r, s be large safe primes. Pick $g \in \mathbb{Z}_n$ such that $g \bmod r^2$ is a primitive element of $\mathbb{Z}_{r^2}^*$. Then $c = g^x \bmod n$ is a trapdoor one-way function with r as a trapdoor [19]. Value x can be computed using the trapdoor as $x = \dfrac{((c^{r-1} \bmod r^2) - 1)/r}{((g^{r-1} \bmod r^2) - 1)/r} \bmod r$. The function is secure if the factorization of n is hard. Size recommendations for n are the same as for RSA.

Notation

For various proofs of knowledge or representation, we use the efficient notation introduced by Camenisch and Stadler [8]. The protocol for proving the knowledge of discrete logarithm of c with respect to g is denoted as $PK\{\alpha : c = g^\alpha\}$. The proof of discrete log equivalence with respect to different generators g_1, g_2 is denoted as $PK\{\alpha : c_1 = g_1^\alpha \wedge c_2 = g_2^\alpha\}$. A signature by a traditional PKI (e.g., RSA) scheme of a user U on some data is denoted as $Sig_U(data)$. The symbol ":" means "such that", "|" means "divides", "$|x|$" is the bitlength of x and "$x \in_R \{0,1\}^l$" is a randomly chosen bitstring of maximum length l.

3.4 Cryptographic Specification of Protocols

$(params, K_{RR}, K_I) \leftarrow \texttt{Setup}(k, l, m)$ protocol: the goal of the protocol is to generate system parameters $params$, RR's revocation key K_{RR} and Issuer's key K_I. The protocol inputs security parameters k, l, m (k is the length of the challenge/hash function used, l relates to the length of Users' secrets, and m is the verification error parameter). The Issuer generates a group H defined by a large prime modulus p, generators h_1, h_2 of prime order $q : |q| = 2l$ and $q|p-1$. The Revocation Referee RR generates group G for the Okamoto-Uchiyama Trapdoor One-Way Function. G is defined by the modulus $n = r^2 s$ with r, s large primes ($|r| > 720, |r| > 4.5l, |n| \geq 2048$, $r = 2r' + 1$, $s = 2s' + 1$, r', s' are primes), generator $g_1 \in_R \mathbb{Z}_n^*$ of order $ord(g_1 \bmod r^2) = r(r-1)$ in $\mathbb{Z}_{r^2}^*$ and $ord(g_1) = rr's'$ in $\mathbb{Z}_n^*$. RR also randomly chooses its secrets $S_1, S_2, S_3 : |S_1| = 2.5l, |S_2| = l, |S_3^{-1} \bmod \phi(n)| = l$ and $GCD(S_1, \phi(n)) = GCD(S_2, \phi(n)) = GCD(S_3, \phi(n)) = 1$. Finally, RR computes an attribute seed $A_{seed} = g_1^{S_1} \bmod n$

(public, common for all Users, linked to a specific personal data type, e.g. citizenship) and values $g_2 = g_1^{S_2} \bmod n, g_3 = g_1^{S_3} \bmod n$. There might be more types of attribute seeds (different A_{seed_i}'s and S_{1_i}'s) related to different attributes Users want to prove. In that case, each unique A_{seed_i} represents one attribute, e.g. nationality, driving permission or legal voting age.[1] These seeds can be aggregated together by multiplying mod$n$. Thus, in general, the credential construction gathers more attributes. In the rest of the paper, we consider for simplicity only one A_{seed} in credential, making attribute and credential the same.

The values $q, p, h_1, h_2, n, g_1, g_2, g_3, A_{seed}$ are made public as system parameters *params*, while r, s, S_1, S_2, S_3 are securely stored at RR as K_{RR} key. Additionally, we use a traditional digital signature scheme (e.g., RSA). Issuers and Users are equipped with a private/public key-pair for digital signatures. This can be accomplished by existing techniques for PKI. The Issuer's private key represents the K_I.

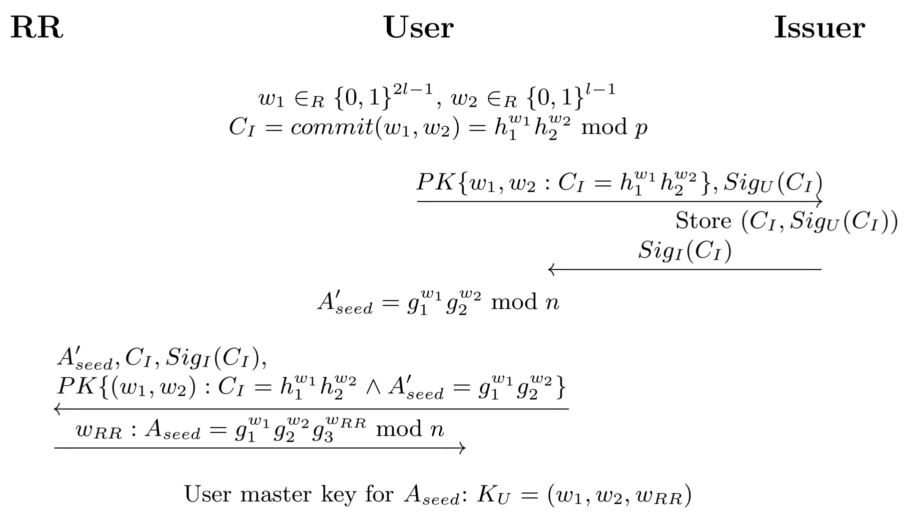
RR
User
Issuer
$w_1 \in_R \{0,1\}^{2l-1}$, $w_2 \in_R \{0,1\}^{l-1}$
$C_I = commit(w_1, w_2) = h_1^{w_1} h_2^{w_2} \bmod p$
$PK\{w_1, w_2 : C_I = h_1^{w_1} h_2^{w_2}\}, Sig_U(C_I)$
Store $(C_I, Sig_U(C_I))$
$Sig_I(C_I)$
$A'_{seed} = g_1^{w_1} g_2^{w_2} \bmod n$
$A'_{seed}, C_I, Sig_I(C_I)$,
$PK\{(w_1, w_2) : C_I = h_1^{w_1} h_2^{w_2} \wedge A'_{seed} = g_1^{w_1} g_2^{w_2}\}$
$w_{RR} : A_{seed} = g_1^{w_1} g_2^{w_2} g_3^{w_{RR}} \bmod n$
User master key for A_{seed}: $K_U = (w_1, w_2, w_{RR})$

Fig. 2. `IssueAtt` Protocol

$K_U \leftarrow$`IssueAtt`$(params, K_I, K_{RR})$ protocol: the first part of the `IssueAtt` protocol runs between the User's smart-card and the Issuer. The communication is not anonymous here, thus the Issuer can physically check the identity of the User, his other attributes etc. Then, User's smart-card generates User's contribution to the master key (w_1, w_2) and commits to these values. The commitment C_I is digitally signed[2] by the User and sent with an appropriate construction

[1] A public list of attributes and their assigned A_{seed_i}'s is maintained by RR. Additional A_{seed_i}'s can be computed and published dynamically, on demand from Issuers.

[2] Here, we rely on already established PKI, e.g., RSA signatures.

correctness proof $PK\{w_1, w_2 : C_I = h_1^{w_1} h_2^{w_2}\}$ to the Issuer. The Issuer checks the proof, the signature and replies with his digital signature on the commitment. In this phase, the User generated and committed to his key contribution. It will be used in all his future attribute proofs. The Issuer approved a new User by signing the committed key contribution.

The second part of the protocol runs between the User's smart-card and RR. In this phase, RR checks the signature of the Issuer on User's commitment C_I and computes his contribution w_{RR} to User's master key such that $A_{seed} = g_1^{w_1} g_2^{w_2} g_3^{w_{RR}} \bmod n$ holds. As a result, the User's smart-card learns all parts of the K_U, namely User's part (w_1, w_2) and RR's part w_{RR}. This triplet forms the discrete logarithm representation of the seed A_{seed} such that $A_{seed} = g_1^{w_1} g_2^{w_2} g_3^{w_{RR}} \bmod n$. This representation can be computed only in cooperation with RR (who knows the factorization of n). Although A_{seed} is shared among all Users as a system parameter, the triplet (w_1, w_2, w_{RR}) is unique for each user, since (w_1, w_2) is randomly generated by each User's smart-card and w_{RR} is generated by RR. Due to the discrete logarithm assumption, Users are stuck to their keys and they are unable to compute other valid keys without knowing K_{RR}. The master key K_U never leaves the smart-card and is stored in card's hardware-protected memory. All operations involving (w_1, w_2, w_{RR}) are computed on the card.

The second part of the `IssueAtt` protocol can be repeated to obtain keys for all demanded attributes. All attributes can be aggregated by multiplying $\bmod n$, keys are aggregated using plain addition. For simplicity, we describe the proof of only 1 attribute. The `IssueAtt` is depicted in Figure 2.

$proof$ ←`ProveAtt`$(params, K_U)$ protocol: the protocol is used by User's smart-card to construct a proof about attribute ownership. In the protocol, the User proves the knowledge of his master key $K_U = (w_1, w_2, w_{RR})$. The session is randomized by a session key K_S. User creates a commitment C_2 to the session key K_S and proves its correctness. The protocol transcript forms the $proof$ output. The protocol is illustrated in Figure 3 in CS notation and in Figure 4 in full.

RR **User** **Verifier**

$$A_{seed} = g_1^{w_1} g_2^{w_2} g_3^{w_{RR}} \bmod n$$
$$K_S \in_R \{0,1\}^l$$
$$A = A_{seed}^{K_S} \bmod n$$
$$C_1 = g_3^{K_S w_{RR}} \bmod n$$
$$C_2 = g_3^{K_S} \bmod n$$

$$\xrightarrow{\substack{PK\{(K_S, K_S w_1, K_S w_2, K_S w_{RR}) : A = g_1^{K_S w_1} g_2^{K_S w_2} g_3^{K_S w_{RR}} \\ \wedge A = A_{seed}^{K_S} \wedge C_1 = g_3^{K_S w_{RR}} \wedge C_2 = g_3^{K_S}\}}}$$

Fig. 3. `ProveAtt` Protocol in Camenisch-Stadler Notation

User $\qquad A_{seed} = g_1^{w_1} g_2^{w_2} g_3^{w_{RR}} \bmod n$ $\qquad$ **Verifier**

$K_S \in_R \{0,1\}^l$
$A = A_{seed}^{K_S} \bmod n$
$C_1 = g_3^{K_S w_{RR}} \bmod n$
$C_2 = g_3^{K_S} \bmod n$
$r_1, r_2 \in_R \{0,1\}^{m+k+3l}$
$r_3 \in_R \{0,1\}^{m+k+4.5l}$
$r_S \in_R \{0,1\}^{m+k+l}$
$\bar{A}_{seed} = g_1^{r_1} g_2^{r_2} g_3^{r_3} \bmod n$
$\bar{A} = A_{seed}^{r_S} \bmod n$
$\bar{C_1} = g_3^{r_3} \bmod n$
$\bar{C_2} = g_3^{r_S} \bmod n$

$$\xrightarrow{A, \bar{A}, \bar{A}_{seed}, C_1, C_2, \bar{C_1}, \bar{C_2}}$$

$$\xleftarrow{e \in_R \{0,1\}^k}$$

$z_1 = r_1 - eK_S w_1$
$z_2 = r_2 - eK_S w_2$
$z_3 = r_3 - eK_S w_{RR}$
$z_S = r_S - eK_S$

$$\xrightarrow{z_1, z_2, z_3, z_S}$$

$$C_1 \overset{?}{\not\equiv} C_2^{rev} \bmod n$$
$$\bar{A}_{seed} \overset{?}{\equiv} A^e g_1^{z_1} g_2^{z_2} g_3^{z_3} \bmod n$$
$$\bar{A} \overset{?}{\equiv} A^e A_{seed}^{z_S} \bmod n$$
$$\bar{C_1} \overset{?}{\equiv} C_1^e g_3^{z_3} \bmod n$$
$$\bar{C_2} \overset{?}{\equiv} C_2^e g_3^{z_S} \bmod n$$

Fig. 4. `ProveAtt` Protocol in detail

rev ←`Revoke`(*params*, *proof*, K_{RR}, K_I) protocol: the protocol is executed if a User needs to be revoked from the system or if Verifier wants to reveal malicious users (and has a strong evidence for doing so). The transcript of the `ProveAtt` protocol can be forwarded to the RR entity in case of rule breaking. The RR entity can decide about the type of revocation. Credential revocation, unlinkability revocation or anonymity revocation are available.

Credential Revocation
RR knows the factorization of n thus he knows the trapdoor to the Okamoto-Uchiyama trapdoor function. From C_2, he learns the session key K_S and from C_1, his contribution w_{RR} to the User key K_U. RR can publish revocation information $rev = w_{RR}$ on a public blacklist. Then, each Verifier is able to check if the User is blacklisted or not by checking $C_1 \overset{?}{\equiv} C_2^{rev} \bmod n$. The equation holds only for revoked Users. Using this type of revocation, no identity is revealed and no valid users have to update their keys. The revocation does not influence non-revoked users in any sense. Verifiers only need to periodically download the blacklist with short *rev* values. Also Issuers can initiate the revocation, by sending C_I to RR who is able to link C_I to w_{RR}. The revocation information *rev* is then published by RR in the same way as if revocation was initiated by Verifiers.

Unlinkability Revocation
RR can reveal w_{RR} and w'_{RR} from two transcripts of the `ProveAtt` protocol. If $w_{RR} = w'_{RR}$, then the session has been carried out by the same User.

The revocation of unlinkability can be used by Verifiers to inspect past behavior of a suspected User. Again, a strong evidence of rule breach must be provided to RR who can either allow or reject unlinkability revocation.

Anonymity Revocation
Sometimes, it is necessary to identify malicious users. In that case, RR reveals w_{RR} and finds corresponding C_I since both values are linked by the `IssueAtt` protocol. C_I is then forwarded to Issuer who can de-anonymize the User since he has a database of digitally signed C_I's. The identification is non-repudiable since C_I is digitally signed and perfectly binds the User to the key inside.

4 Security Analysis

The `ProveAtt` protocol assures that legitimate attribute owners are accepted (completeness), dishonest Users are rejected (soundness) and that no additional information about Users is released (zero-knowledge).

Completeness
The honest Users know a valid representation of A_{seed} in the form (w_1, w_2, w_{RR}), so they are almost always accepted. There is a small verification error probability. Since a User does not know $\phi(n)$, he must send answers in proofs of knowledge/representation in $\mathbb{Z}$. Based on [1], to retain the zero-knowledge property, answers must fit within a certain interval, which happens with high probability $P = 1 - 2^{-m}$. Details in [1].

Soundness
The `ProveAtt` protocol is the parallel composition of a subprotocol denoted as $PK\{\alpha : c = g^{\alpha}\}$ described in Section 3.3. We prove its soundness by following the proof of the RSA variant of this protocol [7]. Our proof is adapted to the Okamoto-Uchiyama group which we use. The environment, already specified in sections devoted to setup and issuance, is following: $n = r^2 s, r = 2r' + 1, s = 2s' + 1, g \in_R \mathbb{Z}_n^* : ord(g \bmod r^2)$ in $\mathbb{Z}_{r^2}^*$ is $r(r-1), ord(g)$ in $\mathbb{Z}_n^*$ is $rr's'$ and r', s' are random large primes such that r, s are also primes.

Theorem 1. *Under the assumptions that factoring of n is hard and $\log_g c$ is unknown, given a modulus n, along with elements g, c, it is hard to compute integers a, b such that*

$$1 \equiv g^a c^b \bmod n \textit{ and } (a \neq 0 \textit{ or } b \neq 0). \tag{1}$$

Proof. Suppose there is an algorithm $\mathcal{A}$ that inputs n, g, c and outputs a, b valid in (1). Then we can use $\mathcal{A}$ to either factor n or compute $\log_g c$, both violating assumptions. The output (a, b) satisfies $1 \equiv g^a c^b \equiv g^a g^{\alpha b} \equiv g^{a+\alpha b} \bmod n$, therefore $a + \alpha b \equiv 0 \bmod ord(g)$. We have two cases:

Case 1. Let us consider $a + \alpha b = 0$. Then discrete logarithm $\log_g c$ can be efficiently computed as $\log_g c = \alpha = -\frac{a}{b}$. Case 1 violates the discrete logarithm assumption.

Case 2. Let us consider $a + \alpha b \neq 0$. $\mathcal{A}$ can be used to factor n by choosing α in random, inputting (n, g, g^α) and getting the output (a, b). Using $(a + \alpha b)$, which is a non-zero multiple of $\phi(n)/4$, the adversary can factor n. To efficiently compute a proper factor of n, the adversary can use the technique originally developed for RSA [3]. Case 2 violates the factorization assumption.

Using the Theorem 1, we can prove the soundness like in [7], thus by constructing the knowledge extractor and assuming that the factorization of n is hard. The extractor uses the standard rewinding technique, thus inputs two different valid answers $z, \bar{z}$ on two different challenges $e, \bar{e}$ with the fixed first step $\bar{c}$. The verification equation must hold for both answers: $\bar{c} \equiv g^z c^e \bmod n$ and $\bar{c} \equiv g^{\bar{z}} c^{\bar{e}} \bmod n$. From these two equations, we get $1 \equiv g^{z-\bar{z}} c^{e-\bar{e}} \bmod n$. From the Theorem 1, the User must have used $\log_g c$, since the factorization of n is unknown. Based on the Case 1 of Theorem 1, $(e - \bar{e})$ divides $(z - \bar{z})$, therefore the extractor can extract $\alpha = \frac{z-\bar{z}}{\bar{e}-e}$.

Zero-Knowledge
The subprotocol denoted $PK\{\alpha : c = g^\alpha\}$, as well as the whole `ProveAtt` protocol is composed of classical proof of knowledge Σ-protocols. The zero-knowledge protocol simulator can be constructed in the standard way [12], by choosing random answers and computing the first steps of the protocol from the verification equations. By constructing the Zero-Knowledge simulator, it is possible to prove that no additional information leaks from the protocol. For simplicity, we used challenge e from the Verifier in Figure 4, which would make the protocol secure only against honest Verifiers. Nevertheless, the protocol can be easily modified to become computationally secure against any Verifiers using [14] or fully secure using [11]. In our implementation, we use the Fiar-Shamir heuristic [14] to make the protocol non-interactive and computationally secure. By using a randomized zero-knowledge protocol for each `ProveAtt` session, no other information than the ownership validity is leaked. Thus, the User cannot be identified, traced or profiled.

5 Implementation Results

User's smart-card requires 9 modular exponentiations, 10 modular multiplications and 4 subtractions to construct an attribute proof. The scheme with 2048b modulus n has been implemented on both PC platform and smart-card platform. For PC implementation, we simulated the protocol in Mathematica software. A batch of 500 000 `ProveAtt` sessions has been successfully evaluated with the time of a single session under 61 ms (including both proof generation and verification) on a middle-class computer (2.53GHz Intel X3440 processor).

For the smart-card implementation, we chose .NET smart-cards and MultOS smart-cards. Gemalto .NET V2+ cards do not allow direct access to modular arithmetic operations [17], thus the time of verification is quite slow and comparable to idemix implementation [2]. We are able to reach the time of verification between 8 and 10 s which is impractical. Therefore, we implemented the scheme on the MultOS ML2-80K-65 cards. These cards allow a hardware acceleration of arithmetic operations through a cryptographic co-processor. With these cards, we are able to run the `ProveAtt` protocol in cca 2 s. Recently, we have measured only a proof-of-concept implementation. A major performance improvement is expected if the code is optimized.

6 Conclusion

In this paper, we present a novel scheme for revocable anonymous credentials. Using the scheme, a User can anonymously convince a Verifier about the possession of an attribute, typically about his age, citizenship or some authorization. By staying anonymous and having the control over all released data, users can protect their privacy during the verification process. Our scheme gathers all required features, so far supported only individually. The proposal is the first practical scheme implementable on off-line smart-cards.

Additionally, we add features unavailable before, mainly scalable off-line revocation and malicious user identity revelation. Finally, we present smart-card implementation results which show the scheme to be very practical and ready for commercial application.

In the proposal, we rely on smart-cards' tamper resistance during the generation and storing of User keys. This hardware-based protection against collusion attacks is sufficient for small-to-medium scale deployment. Our future task is to add cryptographic protection to make the scheme secure even on devices without hardware protection.

Acknowledgement. Research was sponsored by the Technology Agency of the Czech Republic project TA02011260 and the Ministry of Industry and Trade of the Czech Republic project FR-TI4/647.

References

1. Bao, F.: An efficient verifiable encryption scheme for encryption of discrete logarithms. In: Quisquater, J.-J., Schneier, B. (eds.) CARDIS 2000. LNCS, vol. 1820, pp. 213–220. Springer, Heidelberg (2000)
2. Bichsel, P., Camenisch, J., Groß, T., Shoup, V.: Anonymous credentials on a standard java card. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS 2009, pp. 600–610. ACM, New York (2009)
3. Boneh, D.: Twenty years of attacks on the rsa cryptosystem. Notices of the AMS 46, 203–213 (1999)
4. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
5. Camenisch, J., et al.: Specification of the identity mixer cryptographic library, Tech. rep. (2010)

6. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)
7. Camenisch, J., Shoup, V.: Practical verifiable encryption and decryption of discrete logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
8. Camenisch, J., Stadler, M.: Proof systems for general statements about discrete logarithms. Tech. rep. (1997)
9. Chaum, D.: Security without identification: transaction systems to make big brother obsolete. Commun. ACM 28, 1030–1044 (1985)
10. Cramer, R.: Modular Design of Secure, yet Practical Cryptographic Protocols. Ph.D. thesis, University of Amsterdam (1996)
11. Cramer, R., Damgård, I., MacKenzie, P.: Efficient zero-knowledge proofs of knowledge without intractability assumptions. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 354–373. Springer, Heidelberg (2000)
12. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
13. Damgård, I., Fujisaki, E.: A statistically-hiding integer commitment scheme based on groups with hidden order. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 125–142. Springer, Heidelberg (2002)
14. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
15. Hajny, J., Malina, L.: Practical revocable anonymous credentials. In: De Decker, B., Chadwick, D.W. (eds.) CMS 2012. LNCS, vol. 7394, pp. 211–213. Springer, Heidelberg (2012)
16. Lapon, J., Kohlweiss, M., De Decker, B., Naessens, V.: Performance analysis of accumulator-based revocation mechanisms. In: Rannenberg, K., Varadharajan, V., Weber, C. (eds.) SEC 2010. IFIP AICT, vol. 330, pp. 289–301. Springer, Heidelberg (2010)
17. Malina, L., Hajny, J.: Accelerated Modular Arithmetic for Low-Performance Devices. In: 34th International Conference on Telecommunications and Signal Processing, pp. 131–135. IEEE (2011)
18. Naumann, I., Hogben, G.: Enisa: Privacy features of eid cards. Network Security Newsletter 2008, 9–13 (2008)
19. Okamoto, T., Uchiyama, S.: A new public-key cryptosystem as secure as factoring. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 308–318. Springer, Heidelberg (1998)
20. Paquin, C.: U-prove cryptographic specification v1.1, Tech. rep. (2011)
21. The European Commission: Safer internet programme (2012), `http://ec.europa.eu/information_society/activities/sip/policy/index_en.htm`
22. The White House: National strategy for trusted identities in cyberspace (2011), `http://www.whitehouse.gov/sites/default/files/rss_viewer/NSTICstrategy_041511.pdf`