Performance Analysis of Accumulator-based Revocation Mechanisms

Jorn Lapon[1], Markulf Kohlweiss[3], Bart De Decker[2], Vincent Naessens[1]

[1] Katholieke Hogeschool Sint-Lieven, Industrial Engineering
[2] Katholieke Universiteit Leuven, CS-DISTRINET
[3] Katholieke Universiteit Leuven, ESAT-COSIC / IBBT

Abstract. Anonymous credentials are discussed as a privacy friendlier replacement for public key certificates. While such a transition would help to protect the privacy of digital citizens in the emerging information society, the wide scale deployment of anonymous credentials still poses many challenges. One of the open technical issues is the efficient revocation of anonymous credentials. Currently, accumulator based revocation is considered to be the most efficient and most privacy friendly mechanism for revoking anonymous credentials. This paper analyses the performance of three accumulator based credential revocation schemes. It identifies the bottlenecks of those implementations and presents guidelines to improve the efficiency in concrete applications.

Keywords: Anonymous Credentials, Revocation, Accumulators

1 Introduction

In an increasing information-driven society, preserving privacy becomes essential. However, anonymity comes with a snag: it facilitates clandestine and illegal actions. Anonymous credentials, such as *idemix*[1] and *U-Prove*[2], promise a solution. They protect the user's privacy, while ensuring accountability. Another emerging trend is the increasing amount of mobile applications and services. People use their mobile device to read mails, access social networks, order tickets, and much more. As these mobiles are easily lost or stolen, effective revocation mechanisms are essential. As anonymous transactions should still remain anonymous, this is not a trivial task. Efficient revocation mechanisms are crucial, especially if anonymous credentials are used in large scale settings such as electronic identity cards and e-passports.

Currently, the most efficient solutions [3–5], among the privacy friendly ones, use cryptographic accumulators to support revocation. Verifier local revocation [6] is also efficient for some scenarios, but has other disadvantages. For instance, if a credential is ever revoked, signatures created with that credential before its revocation, become linkable. Accumulators have been used for a variety of applications such as time-stamping [7], fail-stop signatures [8], credential or membership revocation [3], and broadcast authentication [9, 10].

Contrary to certificate revocation lists (CRL) in public-key infrastructures, that act as a blacklist of revoked certificates, accumulators are primarily used for *white-list* revocation mechanisms. The accumulator acts as a white-list representing the 'identifiers' of valid credentials. Because of the compact representation and efficient proofs of membership used in such schemes, accumulators are more suitable for *white-list* revocation than CRL-based mechanisms. Accumulator schemes that could be used for *black-list* revocation exist in the literature [11, 12]. Their *non-membership proofs* are, however, less efficient.

Our Contribution. This paper evaluates and compares three accumulator schemes for the revocation of anonymous credentials based on white-listing: the scheme proposed by Camenisch and Lysyanskaya `CL`, [3]; the scheme due to Nguyen `LN` [4]; and the construction due to Camenisch, Kohlweiss and Soriente `CKS` [5]. We compare their computational and storage performance and evaluate their suitability for massive deployment (e.g. in a national eID infrastructure). Finally, relevant optimisation guidelines are proposed.

The paper is structured as follows: Section 2 introduces some technologies used in this paper. Section 3 briefly describes the implemented accumulator schemes. Some implementation details and configuration settings are given in Section 4 followed by the results in Section 5. These results are further discussed in Section 6 and finally some conclusions are drawn.

2 Definitions

Cryptographic Accumulators. A *cryptographic accumulator*, first introduced by Benaloh and de Mare [7], is a construction which allows the accumulation of a number of elements into one value. The size of this value is independent of the number of elements incorporated. For each accumulated element, there is a witness that allows to prove that the element is contained in the accumulator. It must be infeasible, for the adversary, to find a witness for an element that is not included in the accumulator. Camenisch and Lysyanskaya [3] further extended this notion into *dynamic accumulators*. In dynamic accumulators adding and removing values and updating individual witnesses can be done dynamically [1].

Anonymous Credentials. Anonymous credential systems [13, 14, 1, 2] allow for anonymous yet accountable transactions between users and organisations. Moreover, selective disclosure allows the user to reveal only a subset of possible properties of the attributes embedded in the credential: e.g. a credential with the user's date of birth as an attribute can be used to prove that the owner is over 18 without disclosing the exact date of birth or other attributes. The accumulator based revocation schemes are especially useful for anonymous credentials in which multiple shows are unlinkable [13–15]. All these schemes originate from existing group signatures and identity escrow schemes. To add support for accumulator-based revocation, the user needs to prove during the show of such a credential, that a hidden identifier bound to the credential is still contained in the accumulator.

Bilinear maps. Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be (multiplicative) groups of prime order q. A bilinear map (also known as a pairing) from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$ is a computable map e: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with the following properties:

1. *Bilinearity*: for all $u \in \mathbb{G}_1, v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}$: $\mathsf{e}(u^a, v^b) = \mathsf{e}(u, v)^{ab}$
2. *Non-degeneracy*: for all generators $g \in \mathbb{G}_1, h \in \mathbb{G}_2$: $\mathsf{e}(g, h)$ generates $\mathbb{G}_T$.
3. *Efficiency*: there is an efficient algorithm $\mathtt{BMGen}(1^k)$ that outputs (q, $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, e, g, h) to generate the bilinear map (with k the security parameter and g and h generators) and an efficient algorithm to compute $\mathsf{e}(u, v)$ for any $u \in \mathbb{G}_1, v \in \mathbb{G}_2$

As the protocols proposed in [3] and [4] are both constructed using symmetric pairings, only symmetric bilinear maps ($\mathbb{G}_1 = \mathbb{G}_2$) will be considered in this paper.

3 Accumulator Schemes

This section briefly discusses the schemes in [3–5] (i.e. `CL`, `LN` and `CKS`) and summarizes their properties. We give a common interface for accumulator based revocation of anonymous credentials based on these systems. For a more detailed discussion, we refer to the original papers.

The common interface defines the protocols required for processing anonymous credentials with accumulator-based revocation. The schemes under evaluation, all specify these protocols, hence, we did not modify the protocols in any major way. We do, however, implement a common book-keeping approach that deviates slightly from the one given in the referred papers. An archive table H records the history of the accumulator and allows to derive the list of added and revoked elements (V_a resp. V_r) at a given time.

The entities participating in the protocols are: the issuer I, responsible for the creation and revocation of credentials; the user U, the owner of an anonymous credential; and the verifier V. The verifier checks the revocation status of the user with the help of a zero knowledge proof ($\mathsf{authCred}$). In the schemes below, the following notation will be used to denote a local computation performed by X and a protocol between X and Y respectively:

$X : (out_c; out_x) \leftarrow f(in_c; in_x)$

$X \leftrightarrow Y : (out_c; out_x; out_y) \leftarrow f(in_c; in_x; in_y)$

The computations take public input in_c and secret input in_x, in_y from X and Y respectively and result in outputs out_x and out_y to X and Y respectively; out_c is public output. Empty inputs and outputs are represented by ε.

$I : (pk_I, acc, H = \emptyset; sk_I) \leftarrow \mathsf{initScheme}(1^k, N_t; \varepsilon)$
is a probabilistic key generation algorithm that is executed by the issuer. It initializes the environment for the credential scheme for a given security level k. The second input is the capacity of the accumulator N_t, i.e., the maximum number of elements that can be accumulated. The public key pk_I also fixes the set X of all elements that can potentially be accumulated (with $|X| = N_t$). acc is the initial cryptographic accumlator. The history H, is initially empty.

$U \leftrightarrow I : (acc', H'; cred_U; \varepsilon) \leftarrow \mathsf{issueCred}(acc, H, pk_I; \varepsilon; sk_I)$
is a probabilistic interactive algorithm run by the issuer and a user. The issuer issues the credential $cred_U$ to the user and adds the credential's identifier $id_C \in \mathsf{X}$ to the accumulator acc. The credential includes witness information wit_C and a private key, that is unknown to the issuer. Only the issuer can add new credentials, as the secret key sk_I is required for issuing. The new history $H' = H \cup \{\langle id_C, \text{“}\texttt{add}\text{”}\rangle\}$ is updated accordingly.

$I : (acc', H'; \varepsilon; \varepsilon) \leftarrow \mathsf{revokeCred}(acc, H, pk_I; sk_I, id_C)$
is a probabilistic algorithm that is executed by the issuer to revoke the credential id_C. The new history becomes $H' = H \cup \{\langle id_C, \text{“}\texttt{delete}\text{”}\rangle\}$.

$U : (\varepsilon; wit_C{}') \leftarrow \mathsf{updWitness}(acc', H', pk_I; wit_C)$
is a deterministic algorithm, usually executed by the user, that updates the witness to correspond with the latest accumulator value acc'. However, as no secret data is required, this protocol can be performed by another, possibly untrusted, entity. The duration of witness updates depends on the number of elements added or revoked since the last witness update. The latter can be inferred from the book-keeping information H.

$U : (\varepsilon; boolean) \leftarrow \mathsf{verifyAccu}(acc, pk_I; id_C, wit_C)$
is a deterministic algorithm to verify that id_C is indeed accumulated in acc based on the up-to-date witness information wit_C.

$U \rightarrow V : (\varepsilon;\varepsilon;boolean) \leftarrow \mathsf{authCred}(acc, pk_I; cred_U; \varepsilon)$
is a two-party non-interactive zero-knowledge proof protocol that allows the user to prove to the verifier that $cred_U$ is a valid credential (i.e. authentic and not revoked).

3.1 CL Scheme

Camenisch and Lysyanskaya [3] were the first to introduce an accumulator scheme for the revocation of anonymous credentials. The scheme extends the collision-resistant accumulator defined by Baric and Pfitzman, based on the *strong RSA* assumption, allowing dynamic updates of the accumulated set. The core of the accumulator is constructed as follows:

$acc = g^{\prod_{id_i \in V_a \backslash V_r} id_i} \mathsf{mod}\ n$, $wit_C = g^{\prod_{id_i \in V_a \backslash V_r}^{id_i \neq id_C} id_i} \mathsf{mod}\ n$, verifyAccu: $acc \stackrel{?}{=} (wit_C)^{id_C} \mathsf{mod}\ n$,
with n an RSA modulus (i.e. the product of two safe primes p and q), g a quadratic residue modulo n, id_C the accumulated value (X is the set of random primes from a predefined interval), wit_C the witness of the user and $id_i \in V_a \backslash V_r$ the currently accumulated values.

The authors applied the accumulator scheme to the identity escrow scheme due to Ateniese et al. [16], which was at that time, the most efficient and secure identity escrow scheme known. Later on, the efficiency of the protocol has been further improved. One of these schemes, based on the so-called *SRSA*-CL-signatures [17], is used in the credential scheme proposed in [15]. This scheme, that is also used in *idemix*, can be easily combined with the proof that a committed value has been accumulated, using Pedersen commitments. This proof was also mentioned in the paper [3]. We integrate the accumulated value id_Cas an attribute of the credential. For the signature, we compute A such that $Z = S^v R_1^{s_C} R_2^{id_C} A^e \mathsf{mod}\ n$ with $S \in_R QR_n$, $R_1, R_2, Z \in_R \langle S \rangle$, random prime e, master secret s_C and accumulated value, prime $id_C \in \mathsf{X}$.

3.2 LN Scheme

Nguyen [4], was the first to use bilinear maps to implement a dynamic accumulator for revocation. The security of the accumulator is based on the *q-SDH* assumption, with q an upper-bound on the number of elements to be accumulated. The scheme employs a symmetric bilinear map e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$[4].

$acc = g^{\prod_{id_i \in V_a \backslash V_r} (id_i + s)}$, $wit_C = g^{\prod_{id_i \in V_a \backslash V_r}^{id_i \neq id_C} (id_i + s)}$, verifyAccu: $\mathsf{e}(h^{id_C} h_{pub}, wit_C) \stackrel{?}{=} \mathsf{e}(h, acc)$,
with g, h generators of $\mathbb{G}_1$, $id_C \in_R \mathbb{Z}_q^*$, the accumulated value and wit_C the witness of the user, $id_i \in V_a \backslash V_r$ the currently accumulated values, and issuer secret $s \in_R \mathbb{Z}_q^*$ with $h_{pub} = h^s$.

The signature scheme, used by the authors, is based on the signature scheme due to Boneh and Boyen [18] : $\sigma = (h_0 h_1^{s_C})^{1/(id_C + s_I)}$ with h_0, h_1 generators of $\mathbb{G}_1$, master secret $s_C \in_R \mathbb{Z}_q^*$, accumulated value id_C and issuer secret s_I. This scheme is proven secure [19] under the *q-SDH* assumption [18].

3.3 CKS Scheme

A recent scheme (2009) implementing dynamic accumulators was proposed by Camenisch, Kohlweiss and Soriente [5]. Similar to the LN-scheme, this scheme uses a bilinear map e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$. However, the construction of the accumulator is different and is based

[4] Note: in the original paper, the group operations were expressed using the additive notation.

on another assumption, the *n-DHE* assumption. Additionally, the *n-HSDHE* assumption is required for the proof that a hidden value is accumulated.

$acc = \prod_{id_i \in V_a \setminus V_r} (g_{N_t+1-i})$, $wit_C = \prod_{id_i \in V_a \setminus V_r}^{id_i \neq id_j} (g_{N_t+1-i+j})$, verifyAccu: $\frac{e(g_i, acc)}{e(g, wit_C)} \stackrel{?}{=} z$, with g a generator of the group $\mathbb{G}_1$, N_t, the capacity of the accumulator such that $\mathsf{X}=[g_1 = g^{s^1}, \ldots, g_{N_t} = g^{s^{N_t}}]$, state information $[g_1, \ldots, g_{N_t}, g_{N_t+2}, \ldots, g_{2N_t}]$, issuer secret s and the set of currently accumulated values $V_a \setminus V_r$.

The signature scheme originates from the same Boneh and Boyen signatures as the LN scheme, which is further modified by Camenisch et al. [20] for the issuance of anonymous credentials and proven secure in [21] under the *q-SDH* assumption: $\sigma = (g_i h_0 h_1^{s_C})^{1/(c+s_I)}$ with h_0, h_1 generators of $\mathbb{G}_1$, master secret s_C, issuer secret s_I, random number c, and the accumulated value g_i.

4 Implementation

4.1 Implementation Notes

To compare the schemes discussed above, they are all implemented in C++. The bilinear maps applied in the LN and CKS scheme, are initialized using the PBC Library [22]. This library is built on top of the GNU Multiple Precision Arithmetic Library (GMP [23]), which performs the underlying mathematical operations. To make the comparison as fair as possible, the CL scheme, which does not use bilinear maps, is implemented using the GMP library directly. Where applicable, optimized versions for applying pairings and multi-exponentiations in both libraries are used. However, further optimisations may still be possible.

Various protocols are used to prove knowledge of a valid credential. Most of the papers use the notation introduced by Camenisch and Stadler [24]. Nevertheless, the implementation of these schemes was not straightforward. We had to deal with many details and small differences: e.g., some schemes use a group of known order, others of hidden order; interactive versus non-interactive proofs of knowledge; the length of random values and nonces. In the implementation, the proofs of knowledge were made non-interactive. Interactive proofs can be converted to non-interactive ones, using the Fiat Shamir heuristic [25].

For the anonymous credential schemes, only the required minimal set of attributes are added to the credentials (i.e the master secret). Thus, the overhead in storage and computation, resulting from additional attributes embedded in the anonymous credential, is not reflected in the results. Likewise, for interactive protocols, the communication overhead is not considered.

4.2 Configuration

The pairing-based schemes, LN and CKS are constructed using symmetric pairings. The pairing used, is a '*Type A*' pairing, as defined in the PBC Library. This pairing is the fastest, available in the library, and allows the user to select the field-size and subgroup-size. However, as the implementation is independent of the type of pairing, other symmetric pairings could be used as well. The '*Type A*' pairing has the following properties: e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ on a supersingular curve $E : y^2 = x^3 + x$ with embedding degree 2, field-size $l_q = 1024$ bits, and a subgroup of size $l_r = 192$ bits.

For the *idemix*-based CL scheme, the following system parameters[5] are used: $l_n = 2048, k' = 80, k'' = 80, l_e = 838, l'_e = 120, l_v = 2965, l_\Gamma = 1632, l_\rho = l_m = 498, l_H = 256, l_{id_i} = 160$.

5 Results

This section reports the results of 3 experiments. The storage analysis deals with the size of key-material in the scheme. The computational complexity analysis illustrates the complexity of the protocols and the timing analysis validates the results of the complexity analysis by running the actual protocols. The experiments were executed with the security parameters of section 4.2 on a DELL Latitude P9600 @ 2.53GHz, 4GB RAM.

5.1 Storage Analysis

For each of the implementations, Table 1 summarizes the bit-sizes of the private and public key of the issuer, one credential, and the accumulator. Additionally, the size of one accumulated value is listed. Pairings generally allow better results with respect to the size of cryptographic keys than other schemes. This is reflected in the paper of Nguyen [4]. However, as can be seen in table 1, the difference is less extreme than in the paper. Since the PBC Library does not provide the pairing proposed in Nguyen's paper, another type of pairing, was used, resulting in a larger subgroup $\mathbb{G}_1$.

A more important observation is the fact that the public key of the issuer (pk_I) in the CKS scheme contains state information that depends on the capacity of the accumulator. Although, this information can be omitted for most of the protocols, it is required to make witness updates. This will have an impact on how this scheme is used in practice. In case of massive deployment, for instance, witness updates will require special purpose update servers.

Finally, the elements accumulated in the CL and LN scheme are exponents, while in the CKS scheme they are group elements.

Table 1. Bit-sizes of objects in the credential schemes.

	sk_I		pk_I		$cred_U$		acc		id_C	
CL	l_n	2048	$5l_n + l_r + 3l_g$	15634	$2l_n + k + l_v + l_r$	7719	l_n	2048	l_e	160
LN	$3l_r$	576	$16l_q + l_r$	16576	$6l_q + 2l_r$	6528	$2l_q$	2048	l_r	192
CKS	$3l_r$	576	$(16 + 4N_t)l_q + l_r$	16576+4096N_t	$10l_q + 2l_r$	10624	$2l_q$	2048	$2l_q$	2048

5.2 Computational Complexity Analysis

Table 2 presents the most computationally expensive operations. As shown in the table, the complexity of the *witness update* protocol significantly differs for the three schemes. As each call of authCred requires an up-to-date witness, the efficiency of witness updates is very important.

The CL scheme only requires one exponentiation for newly accumulated elements, and one for newly revoked elements. However, as the size of the exponents is growing linear

[5] Note that the parameters proposed for the CL scheme in [4] do not satisfy the constraints posed by the respective scheme.

Table 2. The most expensive operations (i.e. exponentiations, pairings, multiplications) for the protocols in the credential scheme, with N_a the number of accumulated values and N_r, the number of revoked values. The numbers between brackets denote operations that can be precalculated.

	CL	LN			CKS			
	$exp_{\mathbb{Z}_n}$	$exp_{\mathbb{G}_1}$	$exp_{\mathbb{G}_T}$	*pair*	$*/_{\mathbb{G}_1}$	$exp_{\mathbb{G}_1}$	$exp_{\mathbb{G}_T}$	*pair*
issueCred	18	17		5		10		2
Receiver	*10*	*8*		*5*		*4*		*2*
Issuer	*8*	*9*				*6*		
revokeCred	1	1			1			
updWitness	*1+1*	N_a+N_r			N_a+N_r+1			
verifyAccu	1	1		2				2
authCred	52 [+2]	25	24	9 [+12]		31	28	24 [+4]
Prover	*25 [+2]*	*14*	*10*	*3 [+5]*		*19*	*12*	*9 [+2]*
Verifier	*27*	*11*	*14*	*6 [+7]*		*12*	*16*	*15 [+2]*

with the number of accumulated, respectively revoked elements (i.e $N_a.l_{id_i}$, resp. $N_r.l_{id_i}$), the performance decreases considerably (see Timing Analysis). The LN scheme, on the other hand, requires an exponentiation on a base in $\mathbb{G}_1$ for every element accumulated (N_a) or revoked (N_r), since the last witness update. Updating witness is more efficient in the CKS scheme, as the most expensive operations are a number of multiplications linear in the number of accumulated and revoked elements. Moreover, the scheme requires less expensive operations during the issuance of the credential. However, proving knowledge of a valid credential requires slightly more exponentiations and pairings than in the other schemes. This is due to the fact that id_C is a group element. The credential proof of possession needs to be extended to show that this group element is bound to the other credential attributes. Finally, the table reveals that optimizations of the authCred protocol are possible, especially in the LN scheme, in which twelve pairing operations can be precomputed as they do not alter during the lifetime of the credential. Unfortunately, this requires more storage space. Thus, a balance must be found between storage space and processing efficiency.

5.3 Timing Analysis

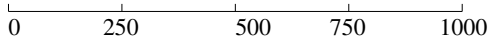
Table 3 shows the results of the experiments that are applied to all the protocols in the three schemes. The results are averaged over 200 test-runs in an accumulator scheme with a maximum capacity of 2500 elements. The witness update results are presented separately.

As can be seen, the results clearly reflect the analysis of the computational complexity. The setup of the CL scheme takes substantially more time than the schemes using bilinear pairings. CL requires the generation of an RSA-modulus as a product of 2 safe primes, which is dominating the setup. Note that the setup of the CL scheme takes on average 1,5 minutes, while the same algorithm takes about 2,5 minutes in *idemix* (which was implemented in Java). The setup time of the CKS scheme, however, includes the creation of state information, which is computed by a large number of exponentiations (twice the capacity of the accumulator). For accumulators with a large capacity (N_t), this may take a substantial amount of the initialisation time. Another interesting fact, not shown in the table, is that for the CL scheme, the generation of a prime number of about 2965 bits takes about $1/3$ of the time needed for the issueCred protocol.

Figure 1 shows the time required for updating a witness, depending on the number of elements (from 1 up to 10000) that have been revoked since the previous witness update. It clearly shows the linear relation with respect to the number of revoked elements. Similar

Table 3. Performance results for the 3 schemes, with respect to the different protocols.

(ms)	CL	LN	CKS
initScheme (sec.)	97	1,26	$1,26s + N_t.4ms$
issueCred	617	365	219
Receiver	274	230	110
Issuer	343	135	109
revokeCred	23	14	0,09
verifyAccu	1,90	130	93
authCred	684	754	938
Prover	389	346	448
Verifier	296	408	490



results are found when elements are added to the accumulator. The figure reveals that the CKS scheme clearly outperforms the others. Nevertheless, the CL and LN scheme may still be useful in specific settings.

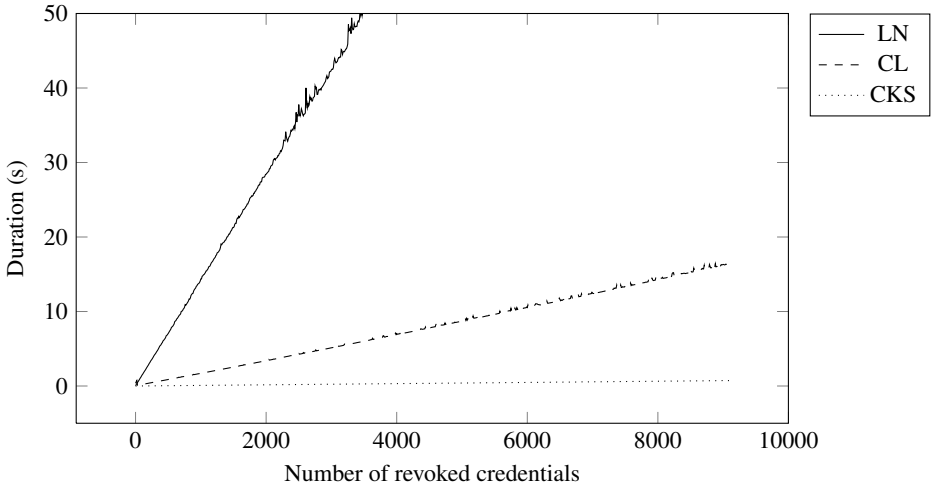


Fig. 1. Performance results for witness updates with respect to the number of revoked credentials, shown graphically.

6 Discussion

6.1 Current Bottlenecks

As the different schemes are based on different security assumptions, a straightforward comparison is difficult. While the CL scheme is based on the *strong RSA* assumption, both CKS and LN schemes are based on the *q-SDH* assumption. However, the CKS accumulator scheme requires two additional assumptions: the *n-DHE* and *n-HSDHE* assumption. According to [26], the *q-SDH* assumption is a weaker assumption than *n-DHE* assumption. As a result, the CKS scheme could have a weaker security than the LN scheme. Additionally, the efficiency of the pairing based systems (i.e. LN and CKS) strongly depends on the efficiency of the selected pairing and its implementation.

When we analyse the signatures, we can observe that the LN and CKS signature schemes have a similar construction. The most important difference is that in the CKS version, the accumulated value is added as a group element, while in the LN scheme it is an exponent. As a consequence, showing a CKS credential requires the proof of a group element. This makes the proof of knowledge (in the authCred protocol) more complicated than the LN version. Nevertheless, the CL scheme outperforms the others for this protocol.

The benefits and drawbacks of the individual schemes are clearly distinct. The construction of the accumulator is important, with a major impact on the updWitness protocol. On the other hand, the efficiency of the issueCred and authCred heavily depends on the design of the credential scheme accompanying the accumulator scheme. Table 4 summarizes the most important bottlenecks of the schemes.

	CL	LN	CKS
initScheme	↘ safe prime generation	↗	↘ state info
issueCred	↘ prime generation		↗
updWitness		↘ exponentiations	↗ (↘ size of state info)
authCred	↗		↘ exp's + pairings

Table 4. Bottlenecks (↘) and benefits (↗) of the schemes

As for efficiency in time and processing, the CL, LN and CKS scheme are comparable, with CL scoring the best on the authCred protocol. However, the LN scheme is faster at the prover side for the same protocol with smaller credentials.

Though still acceptable for most practical applications, the CKS scheme is the slowest for proving a valid credential. On the other hand, this scheme clearly outperforms the others with respect to witness updates. In fact, it's the only scheme that is practical for massive deployment. It is about 180 times faster than the LN scheme, and 22 times faster than the CL scheme. Yet, there is a snag in it. For witness updates, the CKS scheme requires state information, which is linear to the capacity of the accumulator. For instance, with the configuration above, an accumulator for 10 million elements, requires about 4.8 GB of storage. However, since the update of the witness does not require any secret information, special purpose (possibly untrusted) services may perform the update remotely. With respect to storage, the credentials are comparable in size, with the LN-credential the smallest with only 6.528 bits (i.e. 816 bytes).

Large versus small scale environments The scheme that will be selected depends on the characteristics of the concrete application. In small scale environments with a limited num-

ber of revocations or additions, the efficiency of the authCred protocol may be more important than the efficiency of the updWitness protocol. However, an important reason for doing this experiment is to explore the applicability of this technology for the use with an electronic identity card (eID) in a nationwide environment. Compared with the Belgian electronic identity card[6], the accumulator size should contain about 10 million elements. During the introduction of the card, about 2,25 million cards were issued every year, of which 375.000 were revoked.

Suppose in an 'extreme' case, the eID card is used only once a year; this is a valid assumption as a recent survey on the use of the Belgian eID in corporate environments[7] reveals that 56% of the respondents *never* used it. If we can make an interpolation, this would mean that in the best case (i.e. update time grows linearly) an update of 375.000 revoked elements takes about 0.5 minutes, using the fastest update scheme (CKS) and 10 minutes with the CL scheme. While the former may be acceptable in applications such as eID authentication, the latter certainly is not. In the example, we only take the revocations into account, as the accumulator can be precalculated (see *Preissuance-Accumulation* below).

6.2 Practical Solutions

Together with improving the efficiency of the protocols, some relevant application level optimisations can render the schemes practical:

Clustering. Instead of having one accumulator for all users, each user can be assigned to one of M accumulators. Therefore, they could, for instance, be classified per region or even at random into a specific accumulator. As a result, the average number of updates will be about M times less than would be the case with a single accumulator. However, this may entail important privacy consequences. A service will always be able to link an anonymous user to an accumulator. In the worst case, if only one customer of that service is assigned to a particular accumulator, then the service can link all the user's actions.

Preissuance-Accumulation. The example of the Belgian eID shows that during the introduction of the eID, many users are added to the accumulator. To reduce the number of updates, the accumulator could be pre-computed. Meaning that every required element is added to the accumulator and stored securely by the issuer, together with its respective witness.
The CL scheme does not require this pre-computation. As already pointed out by the authors [3], the witness can simply be calculated from the current accumulator by calculating the $(id_x)^{th}$ root of the accumulator, with id_x the new 'accumulated' value. This is possible when the factorisation of the modulus is known.

Delegation of witness updates. To make the CKS scheme practical, without loss of privacy, the witness update should be performed by special purpose update servers. This same strategy may be useful for the other schemes as well. For instance, a resource constrained device, such as a smart card, can delegate the calculation to a more resourceful host.

Epochs. With the protocols provided above, every authCred must be preceded by the updWitness protocol. As described in the CKS paper [5], it is possible to make use of epochs. The accumulator is kept valid for a specific amount of time (i.e. epoch), during which the witness remains valid.

[6] Results obtained from http://godot.be/eidgraphs
[7] SAP Survey: Belgen verdeeld over gebruik van eID op het werk (sept. 2009 by Indigov).

7 Conclusion

The experiments in this paper do not yield a straightforward winner. The revocation of anonymous credentials takes a considerable share in the efficiency of the overall system. Moreover, for nationwide electronic identity cards, the current schemes appear to be inappropriate. Only the CKS scheme may be applicable for massive deployment, provided that special update services are used.

The efficiency of witness updates is an important property in accumulator based revocation systems, and becomes critical in applications with a substantial amount of revocations or additions. The construction of the accumulator has a major impact on the efficiency of the update. Nevertheless, the witness update is also affected by the design of the credential scheme accompanying the accumulator scheme.

The schemes all have different advantages but also different drawbacks. We presented an overview and some relevant guidelines as a lead for future research. Although the accumulators can be used as a building block for anonymous credentials, care must be taken when combining it with a credential scheme.

Acknowledgements We want to thank Lan Nguyen and Jan Camenisch for providing insight in their schemes; Frederik Vercauteren, for the recommendations on the configuration and security of the bilinear maps; and Patrik Bichsel and Alfredo Rial for the discussions on various implementation problems.

This research is partially funded by the Interuniversity Attraction Poles Programme Belgian State, Belgian Science Policy and the Research Fund K.U.Leuven, and the IWT-SBO project (DiCoMas) "Distributed Collaboration using Multi-Agent System Architectures"

Markulf Kohlweiss was supported in part by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government, by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and in part by the European Commission through the ICT and IST programmes under the following contracts: ICT-216483 PRIMELIFE and ICT-216676 ECRYPT II.

References

1. Camenisch, J., Herreweghen, E.V.: Design and implementation of the *idemix* anonymous credential system. In Atluri, V., ed.: ACM Conference on Computer and Communications Security, ACM (2002) 21–30
2. Brands, S.: A Technical Overview of Digital Credentials (2002)
3. Camenisch, J., Lysyanskaya, A.: Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In Yung, M., ed.: CRYPTO. Volume 2442 of Lecture Notes in Computer Science., Springer (2002) 61–76
4. Nguyen, L.: Accumulators from Bilinear Pairings and Applications. In Menezes, A., ed.: CT-RSA. Volume 3376 of Lecture Notes in Computer Science., Springer (2005) 275–292
5. Camenisch, J., Kohlweiss, M., Soriente, C.: An Accumulator Based on Bilinear Maps and Efficient Revocation for Anonymous Credentials. In Jarecki, S., Tsudik, G., eds.: Public Key Cryptography. Volume 5443 of Lecture Notes in Computer Science., Springer (2009) 481–500
6. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: Proceedings of the 11th ACM conference on Computer and communications security, Washington DC, USA, ACM (2004) 168–177
7. Benaloh, J.C., de Mare, M.: One-Way Accumulators: A Decentralized Alternative to Digital Sinatures (Extended Abstract). In: EUROCRYPT. (1993) 274–285
8. Bari, N., Pfitzmann, B.: Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In: EUROCRYPT. (1997) 480–494

9. Papamanthou, C., Tamassia, R., Triandopoulos, N.: Authenticated hash tables. In: Proceedings of the 15th ACM conference on Computer and communications security, Alexandria, Virginia, USA, ACM (2008) 437–448
10. Tartary, C.: Ensuring Authentication of Digital Information Using Cryptographic Accumulators. In Garay, J.A., Miyaji, A., Otsuka, A., eds.: CANS. Volume 5888 of Lecture Notes in Computer Science., Springer (2009) 315–333
11. Li, J., Li, N., Xue, R.: Universal Accumulators with Efficient Nonmembership Proofs. In Katz, J., Yung, M., eds.: ACNS. Volume 4521 of Lecture Notes in Computer Science., Springer (2007) 253–269
12. Damgard, I., Tri, N., Opoulos: Supporting Non-membership Proofs with Bilinear-map Accumulators. http://eprint.iacr.org/2008/538 (2008)
13. Chaum, D.: Security Without Identification: Transaction Systems to Make Big Brother Obsolete. Commun. ACM **28**(10) (1985) 1030–1044
14. Camenisch, J., Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In Pfitzmann, B., ed.: EUROCRYPT. Volume 2045 of Lecture Notes in Computer Science., Springer (2001) 93–118
15. Bangerter, E., Camenisch, J., Lysyanskaya, A.: A Cryptographic Framework for the Controlled Release of Certified Data. In Christianson, B., Crispo, B., Malcolm, J.A., Roe, M., eds.: Security Protocols Workshop. Volume 3957 of Lecture Notes in Computer Science., Springer (2004) 20–42
16. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In Bellare, M., ed.: CRYPTO. Volume 1880 of Lecture Notes in Computer Science., Springer (2000) 255–270
17. Camenisch, J., Lysyanskaya, A.: A Signature Scheme with Efficient Protocols. In Cimato, S., Galdi, C., Persiano, G., eds.: SCN. Volume 2576 of Lecture Notes in Computer Science., Springer (2002) 268–289
18. Boneh, D., Boyen, X., Shacham, H.: Short Group Signatures. [27] 41–55
19. Nguyen, L., Safavi-Naini, R.: Efficient and Provably Secure Trapdoor-Free Group Signature Schemes from Bilinear Pairings. In Lee, P.J., ed.: ASIACRYPT. Volume 3329 of Lecture Notes in Computer Science., Springer (2004) 372–386
20. Camenisch, J., Lysyanskaya, A.: Signature Schemes and Anonymous Credentials from Bilinear Maps. [27] 56–72
21. Au, M.H., Susilo, W., Mu, Y.: Constant-Size Dynamic -TAA. In Prisco, R.D., Yung, M., eds.: SCN. Volume 4116 of Lecture Notes in Computer Science., Springer (2006) 111–125
22. PBC: Pairing-Based Cryptography Library by Ben Lynn (2009)
23. GMP: GNU Multiple Precision Arithmetic Library (2009)
24. Camenisch, J., Stadler, M.: Efficient Group Signature Schemes for Large Groups (Extended Abstract). In: CRYPTO. (1997) 410–424
25. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In Odlyzko, A.M., ed.: CRYPTO. Volume 263 of Lecture Notes in Computer Science., Springer (1986) 186–194
26. Cheon, J.H.: Security Analysis of the Strong Diffie-Hellman Problem. In Vaudenay, S., ed.: EUROCRYPT. Volume 4004 of Lecture Notes in Computer Science., Springer (2006) 1–11
27. Franklin, M.K., ed.: Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. In Franklin, M.K., ed.: CRYPTO. Volume 3152 of Lecture Notes in Computer Science., Springer (2004)