Privacy ABCs: Now Ready for Your Wallets!

Author 1
University
author1@email.org

Author 2
University
author2@email.org

Author 3
University
author3@email.org

ABSTRACT

The paper presents the last bit necessary for making anonymous attribute-based credential schemes (ABCs) practical for large-scale applications that are using smart cards as users' devices for storing credentials: the integration of a fast credential scheme with an efficient offline revocation mechanism. Using proven building blocks, namely wBB signatures, keyed-verification credentials and k-times anonymous proofs, we construct a practical scheme for proving personal attributes anonymously, unlinkably, untraceably and, most importantly, with the verifier-local revocation (VLR) functionality that is running on standard existing smart cards. To prove the practicality of the design, we implemented all the proposed protocols using an off-the-shelf card, benchmarked the proving protocol, compared to existing solutions and put all the source codes on the GitHub as an open source.

CCS CONCEPTS

• **Security and privacy → Cryptography**; **Privacy-preserving protocols**; *Hardware security implementation.*

KEYWORDS

privacy, smart cards, credentials, identity, attributes, revocation

ACM Reference Format:
Author 1, Author 2, and Author 3. 2020. Privacy ABCs: Now Ready for Your Wallets!. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Using anonymous attribute-based credentials (ABCs), users can prove their personal attributes (such as age, citizenship or ticket ownership) without revealing their identity. Furthermore, advanced privacy-enhancing features are provided by ABC schemes, such as the unlinkability, untraceability or selective attribute disclosure.

While the ABC schemes are known for a long time since the publication of [4, 8, 11], their implementation on constrained offline devices was a hard problem for a long time due to high computational complexity. In particular, the implementations of core protocols on smart cards were very impractical until very recently [5, 6, 17, 23]. The implementations with efficient large-scale revocation are still completely missing on smart cards and only available for online and computationally strong user devices.

In this paper, we finally present a scheme that holds all privacy-enhancing features, is provably secure, provides efficient revocation even in applications with millions of users and yet it is implemented and benchmarked on a standard smart card. Since we consider smart cards the most convenient devices for storing and proving personal attributes due to their security, durability and portability, we believe that results presented in this paper will contribute to the practical deployment of ABC technologies in applications such as eID cards, e-ticketing and mass transportation.

1.1 State of the Art

There are several implementations of ABC schemes on programmable smart cards available, such as [5, 12, 17, 23]. However, these implementations lack the revocation, which is a crucial feature for removing misbehaving or invalid users from the system. Revocation was the topic of many papers [7, 9, 14, 18, 20, 21], but none of them proposed practical protocols that can be used in large-scale applications with smart cards due to the following issues: use of unsupported operations (e.g., bilinear pairing), need for periodic updates of a smart card content, need for online communication, loss of unlinkability, only user-driven revocation or missing security proofs. Lueks *et al.* [16] proposed a revocation scheme with low computational cost based on the Vuller's and Alpár's (VA's) Idemix implementation [23] that is part of the IRMA Project[1]. The disadvantage is limited unlinkability within one epoch and need for revocation list re-computation for each verifier, which is very impractical. The scheme was further extended by Verheul [22] to avoid the disadvantages but requires bilinear pairings, that are currently unsupported on smart cards. Efficient revocation scheme for smart-cards was proposed by Camenisch at al. in [6]. However, the integration of the revocation protocols with any ABC scheme is not described, nor implemented. Recently, Camenisch et al. [5] present Keyed-Verification Anonymous attribute-based Credentials (KVAC) based on algebraic MAC and Boneh-Boyen signatures. The solution is designed directly for smart cards. The implementation of all proving protocol algorithms is around 40% faster than the VA's implementation of Idemix [23]. However, the scheme lacks revocation completely.

1.2 Our Contribution

We present the cryptographic design of a novel full-fledged ABC scheme that features practical verifier-local revocation (VLR). Our protocols are based on the combination of keyed-verification proving protocols [10] and k-times unlinkable proofs [6] in revocation protocols. We also present the first implementation results and benchmarks on a standard smart card. To our best knowledge, this

[1] See http://www.irmacard.org

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

paper presents the first practical results from the implementation of a revocable ABC scheme on smart-cards that is usable in large-scale applications with millions of users, such as eIDs.

Finally, we provide the source code of our implementation on GitHub as an open source.

2 CRYPTOGRAPHIC DESIGN

2.1 Preliminaries

The symbol ":" means "such that", the symbol "||" means concatenation and $|x|$ is the bitlength of x. The symbol $\mathcal{H}$ denotes a secure hash function. We write $a \xleftarrow{\$} A$ when a is sampled uniformly at random from A. Let $\mathbf{e}$ denote a bilinear map.

2.2 Cryptographic design

The communication pattern employs the following entities:

- **Revocation Authority** (RA): assigns and issues a unique revocation handler (the private attribute m_r) to each user. Thanks to this attribute, the revocation authority can revoke users.
- **Issuer** (I): is responsible for issuing attributes (personal attributes m_i) to a user via a cryptographic credential *cred*. The credential is digitally signed by the issuer's secret key.
- **User** (U): gets the credential *cred* which includes issued attributes from the issuer and anonymously proves attributes possession to the verifier. Furthermore, the user have to compute a one-time pseudonym C which is linked to the credential *cred* via the revocation handler m_r.
- **Verifier** (V): verifies the possession of required attributes and the revocation status of the revocation handler.

Our scheme consists of the following algorithms:
$(sk_I, params_I) \leftarrow$ `SetupI`(1^κ): the algorithm inputs the security parameter κ and generates the public system parameters (i.e. the implicit input of all other algorithms), namely a a bilinear group with parameters $params_I = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, \mathbf{e})$ satisfying $|q| = \kappa$, and the issuer's private key $sk_I = (x_0, \ldots, x_{n-1}, x_r) \xleftarrow{\$} \mathbb{Z}_q$, where n is the number of all attributes in the credential. In our implementation, we use a bilinear group generated by the MCL [19] library, namely we use the BN-254 curve. The `SetupI` algorithm is run by the issuer.

$(params_{RA}, sk_{RA}, pk_{RA}, RL) \leftarrow$ `SetupRA`$(1^\kappa, ver_{max})$: the algorithm inputs the security parameter κ and the parameter setting the maximum number of unlinkable sessions per user within one epoch ver_{max}. First, the RA computes its keypair as $(sk_{RA} \xleftarrow{\$} \mathbb{Z}_q, pk_{RA} = g_2^{sk_{RA}})$. Second, the algorithm sets integers $(k, j) : ver_{max} = k^j$. In our implementation, we set $j = 2, k = 10$, i.e. 100 pseudonyms per epoch (e.g., a day). Furthermore, the algorithm chooses random integers $(\alpha_1, \ldots, \alpha_j) \xleftarrow{\$} \mathbb{Z}_q$, computes $h_z = g_1^{\alpha_z}$ for all z from 1 to j, chooses randomizers $(e_1, \ldots, e_k) \xleftarrow{\$} \mathbb{Z}_q$ and signs each of them using the wBB signature [3]: i.e. $\sigma_{e_z} \leftarrow g_1^{\frac{1}{e_z + sk_{RA}}}$ for all z from 1 to k. Finally, the RA generates a revocation list RL including an empty list of revocation handlers RH. The algorithm outputs keys (sk_{RA}, pk_{RA}) and parameters $params_{RA} = (q, \mathbb{G}_1, g_1, k, j, (h_1, \ldots, h_j), (\alpha_1, \ldots, \alpha_j), \{(e_1, \sigma_{e_1}), \ldots, (e_k, \sigma_{e_k})\})$. The `SetupRA` algorithm is run by the revocation authority.

$(\sigma, \sigma_{x_1}, \ldots, \sigma_{x_{n-1}}, \sigma_{x_r}, m_r) \leftarrow$ `Issue`$(sk_{RA}, sk_I, RH, (m_1, \ldots, m_{n-1}))$: the algorithm inputs the private key of the revocation authority sk_{RA}, the issuer's private key sk_I, a list of personal attributes $(m_1, \ldots, m_{n-1})$ and the list of all revocation handlers of all users RH. The algorithm outputs signature on all user attributes σ (i.e., cryptographic credential) and updates the list of revocation handlers RH. The algorithm consists of two sub-algorithms: `IssueRA` and `IssueI`. The `IssueRA` algorithm is run fist and after that it is followed by the `IssueI` algorithm.

- The `IssueRA`: the algorithm is run between the user and the revocation authority. RA chooses a random and a unique revocation handler m_r and signs it together with the user's identifier ID, i.e. computes $\sigma_{RA} = g_1^{\frac{1}{\mathcal{H}(m_r||ID)+sk_{RA}}}$. RA updates its list of revocation handlers $RH = RH + m_r||ID$ and sends (m_r, σ_{RA}) back to the user.
- The `IssueI`: the algorithm is run between the user and the issuer. The user sends all its attributes $(m_1, \ldots, m_{n-1}, m_r)$ and RA's signature σ_{RA} to the issuer. The issuer checks the signature validity, signs all required attributes with the issuer's secret key as $\sigma = g_1^{\frac{1}{x_0 + m_1 x_1 + \cdots + m_{n-1} x_{n-1} + m_r x_r}}$ and computes auxiliary values $\sigma_{x_i} = \sigma^{x_i}$ for $1 \le i \le n$. The algorithm outputs cryptographic credential σ and auxiliary values $(\sigma_{x_1}, \ldots, \sigma_{x_{n-1}}, \sigma_{x_r})$.

$(C, \hat{\sigma}, \hat{\sigma}_{e_I}, \bar{\sigma}_{e_I}, \hat{\sigma}_{e_{II}}, \bar{\sigma}_{e_{II}}, \pi) \leftarrow$ `Show`$((m_1, \ldots, m_{n-1}, m_r), (\sigma, \sigma_{x_1}, \ldots, \sigma_{x_{n-1}}, \sigma_{x_r}), \{(e_1, \sigma_{e_1}), \ldots, (e_k, \sigma_{e_k})\}, m_{z \in D}, epoch) \leftrightarrow$ `Verify`$(sk_I, pk_{RA}, m_{z \in D}, C, \hat{\sigma}, \hat{\sigma}_{e_I}, \bar{\sigma}_{e_I}, \hat{\sigma}_{e_{II}}, \bar{\sigma}_{e_{II}}, \pi, RL, epoch) \rightarrow (0/1)$: on the user's side, the algorithm inputs user's attributes $(m_1, \ldots, m_{n-1}, m_r)$, the signature σ (i.e. cryptographic credential), the set of randomization pairs $\{(e_1, \sigma_{e_1}), \ldots, (e_k, \sigma_{e_k})\}$, the indices of disclosed attributes $m_{z \in D}$ and the identifier of the current time epoch $epoch$. On the verifier side, the algorithm inputs the verifier's private key $sk_V = ((x_1, \ldots, x_{n-1}, x_r))$, the revocation lists RL and the epoch identifier. The user outputs the pseudonym C and the cryptographic proof π of the attributes possession. The verifier outputs logical value 0/1, i.e. permit or deny access. The algorithms `Show` and `Verify` are run between the user and the verifier. The detailed description of the `Show` and `Verify` algorithms is depicted in Figure 1. First, the user computes pseudonym C by hashing the epoch identifier, a unique per-session value $i = \sum_{z=1}^{j} \alpha_z e_z$, where e_z and α_z are secret user's parameters (stored on a secure device such as a smart card) and the revocation handler m_r. The user then randomizes its credential and computes a proof of knowledge of all its attributes inside the credential. Furthermore, the user proves that the pseudonym C and signature $\hat{\sigma}$ are constructed using the same attribute m_r. Finally, the verifier verifies the proof π and checks whether the pseudonym C is not placed on the revocation list RL.

$RL \leftarrow$ `Revoke`$(RH, RL, sk_{RA}, \pi, C, \{(e_1, \sigma_{e_1}), \ldots, (e_k, \sigma_{e_k})\})$: the algorithm inputs the list of revocation handlers RH, the revocation list RL, the RA's private key and randomizers $\{(e_1, \sigma_{e_1}), \ldots, (e_k, \sigma_{e_k})\}$, and the communication transcript π, C received from the verifier. The algorithm outputs updated revocation list RL. The `Revoke`

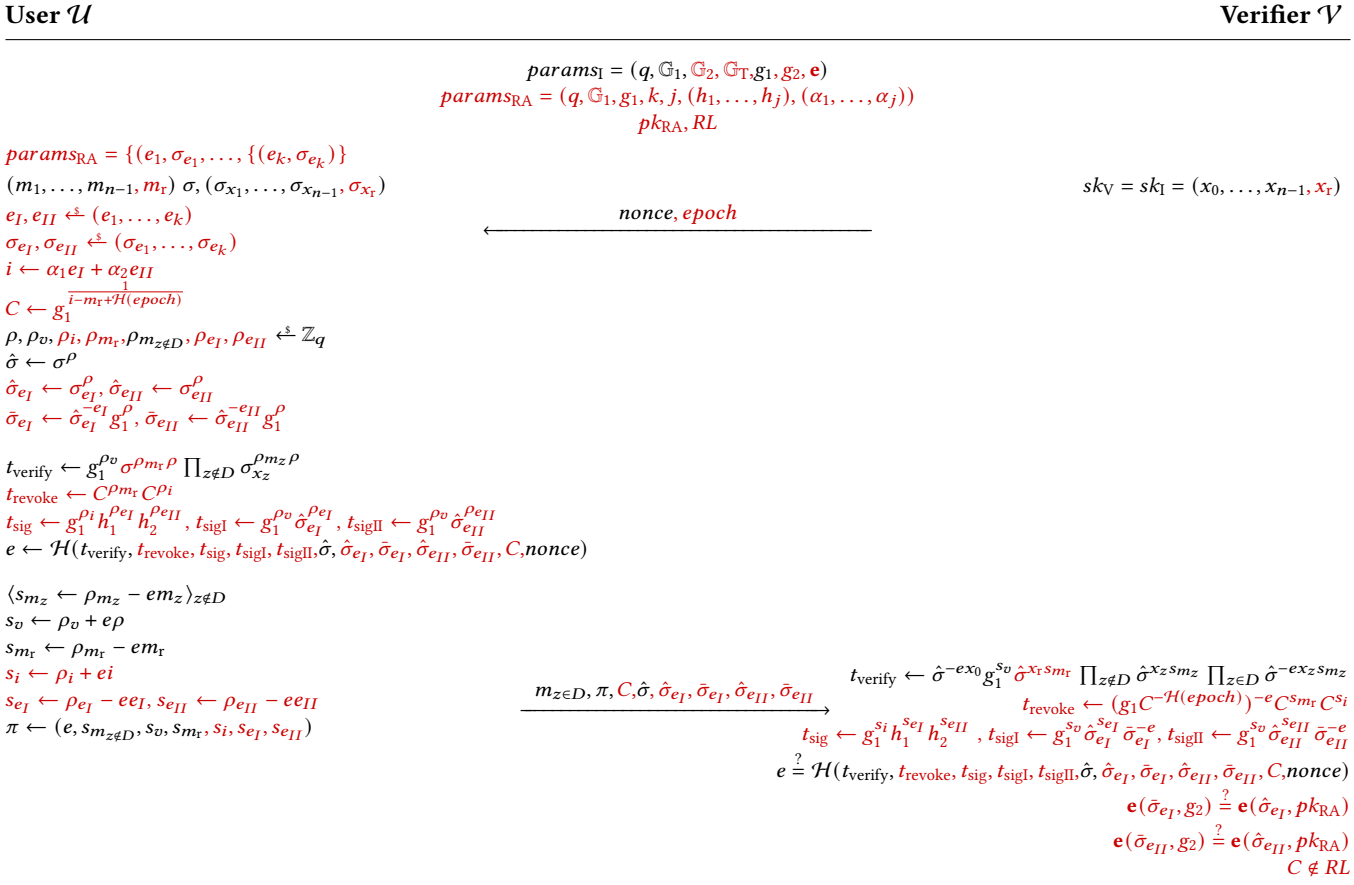

Figure 1: Definition of `Show` and `Verify` algorithms of our scheme (differences to original KVAC scheme are marked red)

algorithm is typically run between the verifier and the revocation authority. The RA is able to reconstruct all pseudonyms of all users for every epoch by computing $C = g_1^{\frac{1}{i - m_{\mathrm{r}} + \mathcal{H}(epoch)}}$, where $i = \alpha_1 e_I + \alpha_2 e_{II}$ is computed for all possible combinations of α_j and e_k (i.e. in case of $j = 2, k = 10$, it is 100 combinations) and m_r is taken from the list of revocation handlers RH.

To keep this contribution as a short paper, we refer to the original papers for more details on the protocols, as they include the motivation for the algorithms design, formal security analysis and more cryptographic background: [5] describes the credential scheme, [6] describes the revocation scheme and [3] describes the wBB signatures.

3 IMPLEMENTATION RESULTS

We provide the proof-of-concept implementation of our scheme in order to benchmark and compare it with the existing schemes. The scheme implementation consists of the smart card side (i.e. entity representing the user) and the terminal side (i.e. entity representing the issuer/verifier and the revocation authority). Both implementations are available on the GitHub public repository: Link Anonymized. In case of the smart card application, only standard MultOS API and free public development environment (Eclipse IDE for C/C++ Developers, SmartDeck 3.0.1, MUtil 2.8) were used. We used standard off-the-shelf programmable smart cards, namely MultOS ML4 contact smart cards (MCU SC23Z018, 1.75 kB RAM, 252 kB ROM, 18 kB EEPROM, OS MultOSv4.3.1), which support only T=0 transmission protocol. The MultOS platform was selected due to its wide support of modular arithmetic and elliptic curve operations (ECC scalar multiplication and ECC addition), see [13] for more details. For the terminal application, OpenSSL [2], MCL [19] and GMP [1] libraries were used. The GMP library achieved the best performance results from all tested libraries which supports bilinear pairing operations, see [15] for more details.

Figure 2 depicts the comparison between our implementation (blue and red) and the original keyed-verification scheme lacking revocation [5] (black and white) for different numbers of attributes stored and disclosed. The figure shows the times for maximum $n-1$ disclosed attributes, where n is the number of stored attributes on the card since the revocation handler (the attribute m_r) is always present but never disclosed. The `Show` and `Verify` algorithms of both schemes were implemented using a pairing-friendly BN-254 curve generated by the MCL library. We stress that the original keyed-verification implementation had to be extended to support

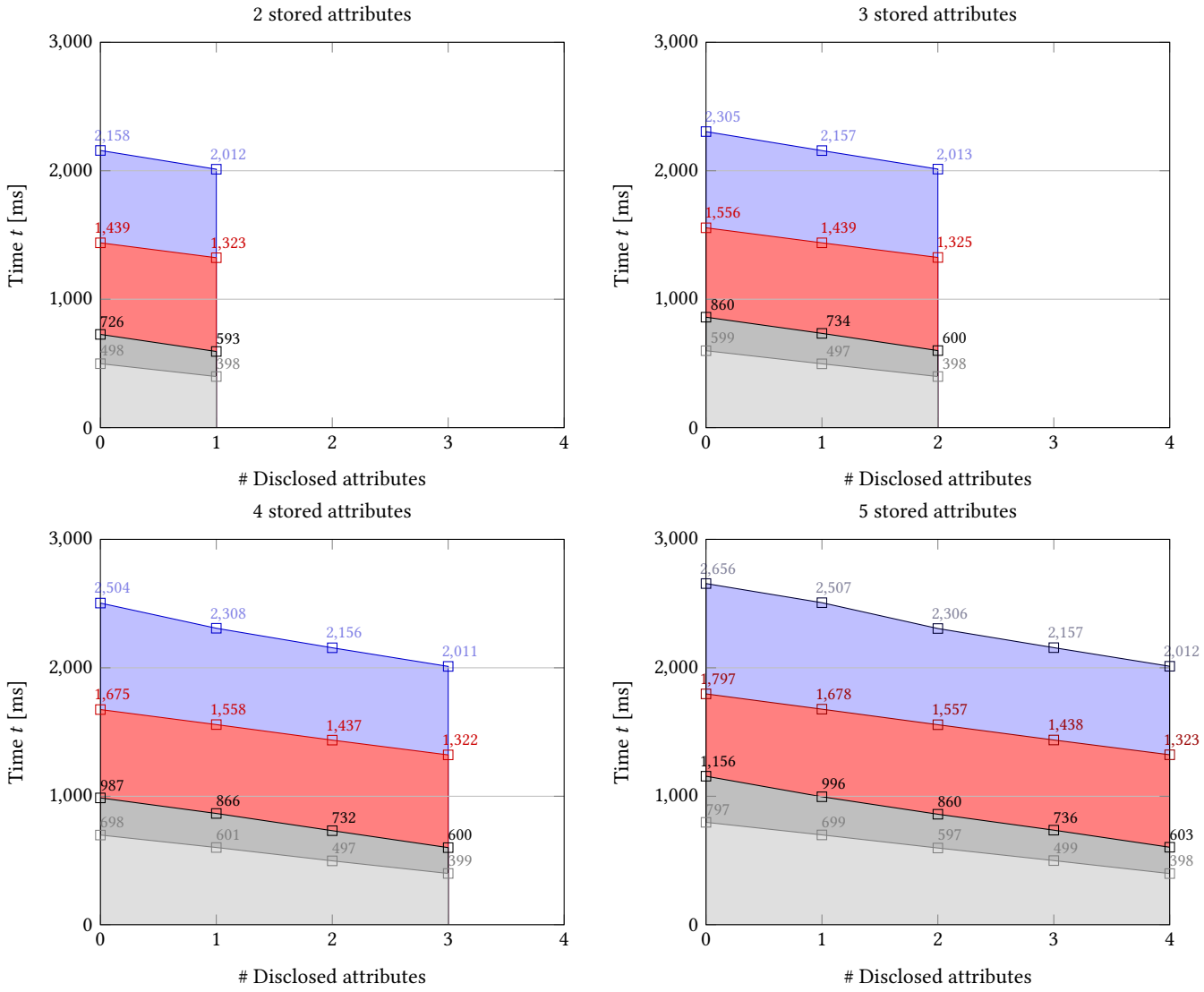

Figure 2: Speed comparison of our Show algorithm implementation with the original keyed-verification implementation lacking revocation mechanisms [5]. Red - our algorithm time, blue - our total time with overhead, black - original algorithm time and grey - original total time with overhead.

BN-254 curve since it supported only the NIST P-192 curve. The algorithm time (in red) shows the time necessary to compute all required operations on the card. The overhead time (in blue) includes the additional time for APDU transmission and MultOS OS processing. Results are the arithmetic means of 10 measurements in milliseconds.

In the case of 2 attributes stored, our scheme requires only 1.4 s to generate the ownership proof. The complexity of the scheme decreases with the number of disclosed attributes, each disclosed attribute reduces Show time by ca. 100 ms. The total time of around 2.7 s is necessary for the proof generation on the card and communication with and computations on the terminal (Raspberry Pi 2 Model B, ARM Cortex-A7, 1 GB RAM, Raspbian 9.3 – 32b). Our implementation is limited to 10 attributes per user, but the available memory resources (approx. 1.75 KB RAM and 7.5 KB usable EEPROM) would allow storing up to 50 attributes on a single card.

4 CONCLUSION

We focused on practical aspects of ABC technologies and presented an integrated scheme that holds all privacy-preserving features, provides efficient revocation and is based on provably secure building blocks. The scheme has been implemented on a standard smart card, benchmarked and the source is openly available on GitHub. As a next step, we plan to further optimize the Show protocol and get the feedback from a real-world deployment.

REFERENCES

[1] 2014. The GNU MP Bignum Library. https://gmplib.org/

[2] 2014. OpenSSL: The Open Source Toolkit for SSL/TLS. https://www.openssl.org/docs/manmaster/crypto/crypto.html

[3] Dan Boneh and Xavier Boyen. 2008. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology* 21, 2 (2008), 149–177.

[4] Stefan A. Brands. 2000. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy.* MIT Press, Cambridge, MA, USA.

[5] Jan Camenisch, Manu Drijvers, Petr Dzurenda, and Jan Hajny. 2019. Fast keyed-verification anonymous credentials on standard smart cards. In *IFIP International Conference on ICT Systems Security and Privacy Protection*. Springer, 286–298.

[6] Jan Camenisch, Manu Drijvers, and Jan Hajny. 2016. Scalable Revocation Scheme for Anonymous Credentials Based on N-times Unlinkable Proofs. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society (WPES '16)*. ACM, New York, NY, USA, 123–133. https://doi.org/10.1145/2994620.2994625

[7] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. 2010. Solving Revocation with Efficient Update of Anonymous Credentials. In *Security and Cryptography for Networks*, Juan A. Garay and Roberto De Prisco (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 454–471.

[8] Jan Camenisch and Anna Lysyanskaya. 2001. *An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation.* Springer Berlin Heidelberg, Berlin, Heidelberg, 93–118. https://doi.org/10.1007/3-540-44987-6_7

[9] Jan Camenisch and Anna Lysyanskaya. 2002. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In *Advances in Cryptology — CRYPTO 2002*, Moti Yung (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 61–76.

[10] Jan Camenish, Manu Drijvers, Petr Dzurenda, and Jan Hajny. 2019. Fast Keyed-Verification Anonymous Credentials on Standard Smart Cards. In *34th International Conference on ICT Systems Security and Privacy Protection - IFIP SEC 2019 (Lecture Notes in Computer Science (LNCS))*. Springer. Lisbon, Portugal.

[11] David Chaum. 1985. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *Commun. ACM* 28, 10 (Oct. 1985), 1030–1044. https://doi.org/10.1145/4372.4373

[12] Antonio De La Piedra, Jaap-Henk Hoepman, and Pim Vullers. 2014. Towards a full-featured implementation of attribute based credentials on smart cards. In *International Conference on Cryptology and Network Security*. Springer, 270–289.

[13] Petr Dzurenda, Sara Ricci, Jan Hajny, and Lukas Malina. 2017. Performance Analysis and Comparison of Different Elliptic Curves on Smart Cards. In *International Conference on Privacy, Security and Trust (PST)*. 1–10. Calgary, Canada, ISBN: 978-1-5386-2487-6.

[14] Jan Hajny, Petr Dzurenda, and Lukas Malina. 2014. Privacy-PAC: Privacy-Enhanced Physical Access Control. *Proceedings of the ACM Conference on Computer and Communications Security*, 93–96. https://doi.org/10.1145/2665943.2665969

[15] Jan Hajny, Petr Dzurenda, Sara Ricci, Lukas Malina, and Kamil Vrba. 2018. Performance Analysis of Pairing-Based Elliptic Curve Cryptography on Constrained Devices. In *2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. IEEE, 1–5.

[16] Wouter Lueks, Gergely Alpár, Jaap-Henk Hoepman, and Pim Vullers. 2017. Fast revocation of attribute-based credentials for both users and verifiers. *Computers & Security* 67 (2017), 308–323.

[17] Wojciech Mostowski and Pim Vullers. 2011. Efficient U-Prove implementation for anonymous credentials on smart cards. In *International Conference on Security and Privacy in Communication Systems*. Springer, Berlin, Heidelberg, 243–260.

[18] Lan Nguyen. 2005. Accumulators from Bilinear Pairings and Applications. In *Topics in Cryptology – CT-RSA 2005*, Alfred Menezes (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 275–292.

[19] Mitsunari Shigeo. 2018. Mcl library. https://github.com/herumi/mcl.

[20] Patrick P. Tsang, Man Ho Au, Apu Kapadia, and Sean W. Smith. 2007. Blacklistable Anonymous Credentials: Blocking Misbehaving Users without Ttps. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07)*. Association for Computing Machinery, New York, NY, USA, 72–81. https://doi.org/10.1145/1315245.1315256

[21] P. P. Tsang, A. Kapadia, C. Cornelius, and S. W. Smith. 2011. Nymble: Blocking Misbehaving Users in Anonymizing Networks. *IEEE Transactions on Dependable and Secure Computing* 8, 2 (2011), 256–269.

[22] Eric R. Verheul. 2016. Practical backward unlinkable revocation in FIDO, German e-ID, Idemix and U-Prove. Cryptology ePrint Archive, Report 2016/217. https://eprint.iacr.org/2016/217.

[23] Pim Vullers and Gergely Alpar. 2013. Efficient Selective Disclosure on Smart Cards Using Idemix. In *Policies and Research in Identity Management*. IFIP Advances in Information and Communication Technology, Vol. 396. Springer Berlin Heidelberg, 53–67.