Anonymous Credentials on Java Card

Patrik Bichsel[1] · Jan Camenisch[1] · Thomas Groß[1] · Victor Shoup[2]

[1]IBM Research – Zürich
CH-8803 Rüschlikon, Schweiz
{pbi | jca | tgr}@zurich.ibm.com

[2]New York University – Courant Institute
New York, NY 10012
shoup@cs.nyu.edu

Abstract

Secure identity tokens such as *Electronic Identity (eID)* cards are emerging everywhere. At the same time user-centric identity management gains acceptance. *Anonymous credential* schemes are the optimal realization of user-centricity. However, on inexpensive hardware platforms, typically used for eID cards, these schemes could not be made to meet the necessary requirements such as future-proof key lengths and transaction times on the order of 10 seconds. The reason for this is the need for the hardware platform to be standardized and certified. Therefore an implementation is only possible as a Java Card applet. This results in severe restrictions: little memory (transient and persistent), an 8-bit CPU, and access to hardware acceleration for cryptographic operations only by defined interfaces such as RSA encryption operations.

Still, we present the first practical implementation of an anonymous credential system on a Java Card 2.2.1. We achieve transaction times that are orders of magnitudes faster than those of any prior attempt, while raising the bar in terms of key length and trust model. Our system is the first one to act completely autonomously on card and to maintain its properties in the face of an untrusted terminal. In addition, we provide a formal system specification and share our solution strategies and experiences gained and with the Java Card.

This summary is based on the research in [BCGS09].

1 Introduction

Electronic authentication tokens are spreading rapidly. Applications today already include ticketing, access to buildings, and road tolls. A number of countries have issued electronic ID (eID) cards or are about to do so. All these existing or emerging solutions have in common that the user is fully identifiable in the transactions involving the token. Indeed, many of them offer strong cryptographic identification or qualified digital signatures. The resulting loss of privacy is subject to discussion as pointed out by Huysmans [Huys08], but it is not a severe problem for e-government applications. However, a government-issued root of trust is very attractive for secondary use by (commercial) service providers. Here, privacy becomes a real issue. Indeed, in many commercial applications, unique identification is inappropriate, attribute-based authentication highly desired, and suitable privacy protection essential to make the services sustainable.

For instance, consider a teenager accessing an online chat room by eID. Here, the aim is to restrict access to teenagers only. It is crucial that no data other than the age range of the teenager is revealed to the chat provider. Indeed, if all the eID card's information is revealed and gets into the wrong hands, more damage is done than protection gained. Furthermore, consider a citizen using the eID throughout her entire lifetime and with various third parties. Without sufficient privacy protection, service providers could trace and profile the citizen across organizations. This would lead to an erosion of the citizens' trust and result in the non-sustainability of the entire system. We believe that sustainable secondary use is a is a make-or-break requirement for eID systems as well as for any identity token that supports authentication with third parties.

The ability to build comprehensive user profiles in the context of attribute-based authentication carries the need for strong privacy protection further than mere trust erosion would. It implies the need for full anonymity, which includes unlinkability. Examining identity tokens, and in particular eID cards, over a long time period, then the monotonous growth of identity information at service providers can only be overcome by full anonymity by default. This requirement entails further goals by implication: *First*, we need privacy-enhanced credential systems, namely, anonymous credential systems. *Second*, no (unnecessary) trusted third parties should be involved in transactions, i.e., the credential system must be autonomous. Optimally, the user shall only trust her own identity token and no other principal. *Third*, if one considers linkability by timing, the credential system must be able to operate offline, based on long-term certificates.

Let us expand these three thoughts before we analyze the trust model and hardware setting, in particular, typical smart cards as used to realize eID cards. Luckily, there exist privacy-enhancing technologies addressing our requirement of full anonymity, and allow for attribute-based access control. Anonymous credential systems [LRSW99] and [CaLy01] allow an identity provider to issue an *anonymous credential* to a user. This credential contains attributes such as the user's address or date of birth but also her rights or roles. It may be used to establish *pseudonyms* and prove possession thereof. Using the credential, the user can prove to a third party that she possesses a credential containing a given attribute or role without revealing any other information. We call this property *selective disclosure*. For example, in the child-protection example described earlier, the youngster could use a government-issued credential to prove that she fulfills the requirement on the age range. Thus, it seems that what is urgently needed is an implementation of anonymous credentials on tokens, such as smart cards. The anonymous credential systems proposed by Brands [Bran00] or Camenisch and Lysyanskaya [CaLy01] can be implemented on ordinary computers as described in [CaHe02] without difficulties. However, it may seem they are not suited for implementation on smart cards or USB tokens. Bichsel [Bich07] and Balasch [Bala08] conclude that only systems using joint computation with the terminal can be implemented given the hardware restrictions for the eID scenario. This statement especially holds, if future proof key lengths of at least 1400 bits are considered. But even tremendously reduced systems did not meet the expected transaction times of production eID cards, which are defined to be in the order of 10 seconds.

Beyond the transaction times and key length, there are three more mundane requirements on eID cards imposed by governments and eID technology providers. *First*, the smart card platform should be standardized, for governments and eID technology providers shy away from proprietary technology lock-ins. Also, we envision the anonymous credential system to be deployed as a complement to existing eID systems and not to replace other authentication mechanisms. Even though one could achieve much more efficient solutions with a native card implementation, this would severely hamper the acceptance for the proposal. We therefore base our work the Java Card 2.2.1 standard [SuMi03].

Second, the eID card must be certified, for instance in a Common Criteria for Information Technology Security Evaluation. Clearly, we need to aim at making the certification gap as small as possible and, therefore, use an off-the-shelf smart card with comprehensive certification.

Third, the smart card platform must be well-established and cheap. We therefore restrict ourselves to smart cards that are 3-4 years old and in production in current eID systems. We also follow the standard operation procedures of these smart cards, e.g., to avoid write-operations to EEPROM whenever possible.

The main obstacle to implementing anonymous credential systems on such cards seems to be twofold. *First*, we need to execute fast modular exponentiations, which requires the use of the card's cryptographic co-processor. However, the interfaces offered by the (off-the-shelf) cards' operating system do not give direct access to this, but only offer high-level functionality such as RSA encryption. Consequently, we will use the limited interfaces that standard Java Cards provide. *Second*, typical smart cards are rather limited in the amount of RAM that can be used for computations. This makes it, for instance, hard to store all intermediary results during an authentication transaction.

In this paper, we show that it is feasible to overcome the technical challenges to implement the Camenisch-Lysyanskaya (CL) anonymous credential system on a standard Java Card. We report an implementation of a DAA-alike system [BrCC04] on an off-the-shelf Java Card, a JCOP v2.2/41. It can execute a proof of possession of a credential in a few seconds, which is fast enough for a multitude of eID use cases. Our prototype outperforms all prior anonymous credential system proposals on smart cards by several orders of magnitude. In contrast to prior proposals, our smart card credential system is autonomous, that is, it forgoes any joint computation with the terminal. Our system not only guarantees the secrecy of the user's master key during the card's complete lifecycle, but also protects user's privacy in face of an untrusted terminal.

2 Related Work

In research, there have even been several approaches to implement anonymous credential systems on smart cards. This summary is based on the research by Bichsel et al. [BCGS09], which established the first implementation of an autonomous anonymous credential system on a standard Javacard. Sterckx et al. [SGPV09] independently established a similar result by realizing standard Direct Anonymous Attestation (DAA) [BrCC04] on Java Card. Their result acts as independent confirmation that realization efficient realization of anonymous credentials on card is very well possible.

Bichsel [Bich07] and Balasch [Bala08] focus on providing the arithmetic functionality required by anonymous credential systems, i.e., fast modular arithmetic. Balasch implements the arithmetic using AVR microcontrollers, whereas Bichsel uses the JCOP platform. Danes [Dane07] provides an analysis of different trust models and compares them with respect to security and privacy. He projects computation times using the hardware specification, implicitly assuming a custom operating system, and obtains an execution time of 6 seconds for his preferred protocol. This protocol still assumes trust in the terminal. We provide a comparison of the measurements of the three approaches with ours in Tab. 1. Note that the table focuses on the computation times of the core anonymous credential system and does not account for additional computations such as revocation equations.

Given that the authors use very different systems, we want to analyze the systems on the basis of single exponentiations. The difference to Bichsel [Bich07] is apparent and does not need further explanation. The implementation by Balasch [Bala08] can be compared by extrapolation of his measurements. An exponentiation of a base/modulus bit length of 1984 with an exponent of 1024 bits accounts to roughly 270 seconds. We are able to compute such an exponentiation in 1.3 seconds.

Tab. 1: Overview of different approaches to establish anonymous credential systems on a smart card.

	[Dane07]	[Bich07]	[Bala08]	[SGPV09]		[BCGS09]		
Date	2007	2007	2008	2009		2009		
Bit Length	--	72	1024	1024		1280	1536	1984
Transaction	--	450s	133.5s	12.5s	4.2s	7.4s	10.5s	16.5s
Trust Model	Trust terminal	Trust terminal	Trust terminal	Autonomous		Autonomous		
Implementation	none	Java Card JCOP v2.2/41	AVR 8-bit RISK	Java Card 2.1.1	Java Card 2.2.1	Java Card JCOP v2.2/41		

3 Requirements

We base our requirements discussion on the scenario of eID cards for three reasons. *First*, eID technology is likely to pervade entire societies and to affect the life of many citizens. *Second*, its actual hardware platform is particularly challenging for implementing an anonymous credential system. *Third*, it allows us to intuitively motivate requirements that abstractly hold for any application involving personal tokens with severe resource restrictions.

3.1 Application Requirements

Sustainable Secondary Use. The users must be able to use their eID card over their entire lifetime without privacy or trust degradation. A continuous strong privacy protection for all transactions is crucial. This is a key requirement that we are going to meet by using an anonymous credential system.

Autonomous Trust Root. A wide range of trust scenarios must be supported without drawbacks on security or privacy. Particularly, the card must act securely in face of an untrusted or malicious terminal. Therefore, the anonymous credential system must protect the citizen's security and privacy autonomously and cannot (easily) delegate computations to the terminal.

The privacy discussion gains in complexity with the introduction of variable attribute policies, as the terminal may attempt to send the eID card multiple policy requests – without the citizen's knowledge or consent – to infer a profile of the citizen. As the eID card is in principle stateless, it is at the mercy of the terminal. The terminal can easily reset the card and send another policy request. Naive solutions to store the card's state or create an audit log of the terminal's requests are not easily feasible because of the cards limitations in write/erase cycles on persistent memory. Proposals that certify card readers as well as applicable policies are used to confine a potential exposure, be it in terms of obtainable attribute set or of potentially malicious readers.

Long-term Certificates. An eID card must forgo short-term updates, particularly of the keys and certificates, be it because some countries support offline applications (such as vending machines), or because some countries ban card updates outside of a trusted environment. In privacy terms, this allows to prevent a linking by timing.

Performance. An anonymous credential system for eID cards faces stiff performance requirements, notably, the need to complete transactions in mere seconds.

Future Proof Key Length. Currently, lengths of an SRSA modulus size greater than or equal to 1400 bits may arguably be considered future-proof.

3.2 Functional Requirements

Clearly, unique identification, qualified signatures, and disclosure of the citizen's full address are important functional requirements for eID cards; however, we focus on functional requirements with stronger privacy properties.

Anonymous Proof of Possession. The card must be able to issue a proof of possession of a credential. Thus, proving the value of an attribute without leaking any information about the attribute value.

Pseudonym and Selective Disclosure. Card implementations should support the establishment and proof of pseudonyms as well as the selective disclosure of a subset of attributes, without identifying the user.

Age proof. Nowadays, eID cards are often used as basis for a proof of age, mostly in the area of youth protection. Contrary to the common perception of an age proof as a means to show adulthood and to obtain restricted goods (media, alcohol, cigarettes), age proofs are also important to establish protection zones for youngsters on the Internet.

Revocation. Revocation is of central importance for eID systems. The card needs to be revoked when the owner declares her eID card lost or stolen. As the traditional approach of revocation lists implies privacy hazards for honest citizens, we need to explore privacy-preserving revocation mechanisms.

4 Protocol Design and Realization

4.1 Design Decisions

An implementation of a full-fledged CL anonymous credential system [IBM09] with all bells and whistles may be challenging on older cards. That is, features such as precise range proofs [Boud00] (i.e., proving that the date of birth contained in the credential issued has a distance of at most *n* years from the current date) or encoding all the more than 20 typical fields of a standard identity card, require several exponentiations. There exist further efficiency measures such as the efficient encoding of finite-set attributes [CaGr08]; however, we do not consider these techniques in this summary. Whereas a native implementation on smartcards with access to a crypto co-processor can manage these computations, an implementation on the older Java Card as we used may result in a computation time on the order of 70-100 seconds. While maintaining that realization of credential systems on smartcards is well feasible, we propose to rely on the tamper resistance of the hardware for such attribute-related proofs on slow cards.

Similarly to the model that the Trusted Computing Group has taken for their TPM chips with the Direct Anonymous Attestation (DAA) protocol [BrCC04], we have a two-stage approach. We use anonymous credentials to have the smart card prove that it is a valid (and intact) card and therefore can be trusted to make statements about its bearer. These statements, e.g., an age proof, are then made by the card itself. Consequently, the correctness of these statements is not enforced cryptographically but by the tamper resistance of the card. Thus, we will have to protect ourselves against the case when a card is broken open and the cryptographic credentials are extracted. In this case, the extracted credential can be used to back any attribute-related statement. However, breaking a card is costly and will mostly be done for economical reasons. Thus, employing techniques that protect from massive sharing might be the most appropriate action.

Our proposed solution is therefore as follows. We issue a Camenisch-Lysyanskaya credential [CaLy01] with a secret key m_0 on an eID card and store all attribute information about the citizen in the card independently of the credential. When the citizen wants to use the cards for some privacy-protecting authentication, we let the eID card compute a valid attribute statement (based on the attribute information stored on the card) and sign it with the Fiat-Shamir heuristic [FiSh86] during a proof of possession of the issued credential. On top of that, the card a revocation mechanism along the lines of DAA [BrCC04] as follows: it provides a discrete log commitment, i.e., $C= g_r^{m0}$ on the secret key m_0 with a random base g_r during each transaction (where it is ensured by the proof of possession that this is chosen correctly). This commitment is a pseudo-random value with computational hiding properties. However, it allows for the detection of revoked cards as authorities can check C against $g_r^{m'0}$ for each m'_0 retrieved from the revocation list and, if there is a match, decline the transaction (or take legal action). We refer to [BCGS09] for an exact protocol specification of the Java Card version and to [IBM09] for a specification of the entire anonymous credential system.

4.2 Realization

[BCGS09] focuses on low-level realization of an anonymous credential system on Java Card, in particular on how to realize fast computations for the credential system with delegations to the crypto co-processor. Here, we focus on a solution strategy to split computations for a proof

of possession in a *policy-independent pre-computation* phase and a *policy-dependent finalization computation* to finish the proof based on the user's inputs.

Policy-independent pre-computation. The system blinds the credential of the user with randomness and computes the initial commitments of the Zero-Knowledge Proof of Knowledge. These computations are independent from user interaction and contain most expensive computations. Even though they should be done freshly, they can be already executed, when the card is inserted into the reader. Our prototype implementation requests this pre-computation as soon as the interaction with the card starts. Usually, it is finished when the user make his user-consent decisions.

Policy-dependent finalization computation. Once the user made a decision on the policy, for instance, which attributes to disclose, the card finalizes the transaction. When only executes a proof of possession, at this stage only inexpensive computations are needed (cryptographic hash function, multiplications). Thus, the waiting time of the user till completion of the proof after pressing the "submit" button is low. We summarize the performance measurements in Tab. 2.

Tab. 2: Computation time comparison for different bit lengths of the modulus.

Modulus Length	**1280 bit**	**1536 bit**	**1984 bit**
Pre-computation	5.2s	7.8s	13.3s
Policy dependent	2.2s	2.6s	3.2s
Total	7.4s	10.5s	16.5s

5 Conclusion and Outlook

We present the first efficient implementation of an anonymous credential system on a standard Java Card. Our system nurtures sustainable secondary use of the user's identity because of the multi-use unlinkability of the credential system. Our system performs the entire computation on card and independently from a potentially malicious terminal. Therefore, we fulfill our major requirement for an autonomous trust root. In addition, our anonymous credential system offers long-term certificates and, therefore, does not require updates when doing many unlinkable proofs.

Our Java Card implementation is capable of efficient proofs of possession of identity credentials. Even though our implementation is able to include several attributes in a credential, we opt to trust the hardware for attribute statements for efficiency on older cards.

In conclusion, we are confident that anonymous credential systems are realizable of smart cards and can help to establish strong privacy-preserving eID cards.

Acknowledgment

We would like to thank the BlueZ group of IBM Research - Zurich for the continuous support and extremely insightful discussions. This work has been funded by the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement no. 216483. Victor Shoup is supported by NSF award number CNS-0716690.

Literature

[Bala08] J. M. Balasch Masoliver. Smart card implementation of anonymous credentials. Master's thesis, K.U.Leuven, Belgium, 2008.

[BCGS09] P. Bichsel, J. Camenisch, T. Groß, and V. Shoup. Anonymous credentials on a standard Java Card. In Proc. ACM CCS, 2009, pages 600-610. ACM Press, Nov. 2009.

[Bich07] P. Bichsel. Theft and misuse protection for anonymous credentials. Master's thesis, ETH Zürich, Switzerland, November 2007.

[Boud00] F. Boudot. Efficient proofs that a committed number lies in an interval. In EUROCRYPT '00, vol. 1807 of LNCS, pages 431-444. Springer, 2000.

[Bran00] S. Brands. Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, 2000.

[BrCC04] E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In Proc. 11th ACM CCS, pages 225-234. ACM Press, 2004.

[CaGr08] J. Camenisch and T. Groß. Efficient attributes for anonymous credentials. In Proc. 15th ACM CCS, pages 345-356. ACM Press, Nov. 2008.

[CaHe02] J. Camenisch and E. Van Herreweghen. Design and implementation of the idemix anonymous credential system. In Proc. 9th ACM CCS. ACM Press, 2002.

[CaLy01] J.Camenisch and A. Lysyanskaya. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. EUROCRYPT '01, vol. 2045 of LNCS, pages 93-118. Springer, 2001.

[Chau85] D. Chaum. Security without identification: Transaction systems to make big brother obsolete. Comm. of the ACM, 28(10):1030-1044, Oct. 1985.

[Dane07] L. Danes. Smart card integration in the pseudonym system Idemix. Master's thesis, University of Groningen, 2007.

[FiSh86] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In CRYPTO '86, vol. 263 of LNCS, pages 186-194. Springer, 1987.

[Huys08] X. Huysmans. Privacy-friendly identity management in egovernment. In The Future of Identity in the Information Society, vol. 262/2008 of IFIP International Federation for Information Processing, pages 245-258. IFIP, Springer, June 2008.

[IBM09] IBM. Cryptographic protocols of the Identity Mixer library, v. 1.0. IBM Research Report RZ3730, IBM Research, 2009.

[LRSW99] A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. Selected Areas in Cryptography, vol. 1758 of LNCS. Springer, 1999.

[SGPV09] M. Sterckx, B. Gierlichs, B. Preneel, and L. Verbauwhede. Efficient implementation of anonymous credentials on Java Card smart cards. In first IEEE Workshop on Information Forensics and Security (WIFS 1999), pages 106-110. IEEE, 1999.

[SuMi03] Sun Microsystems. Java Card platform specification 2.2.1. [online; 18 April 2009], Oct. 2003. http://java.sun.com/javacard/specs.html.