Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 197 (2008) 141–148

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Vulnerabilities in Anonymous Credential Systems

R. Bhaskar[1] K. Chandrasekaran[2] S. V. Lokam[1]
P. L. Montgomery[1] R. Venkatesan[1] Y. Yacobi[1]

Abstract

We show the following:

(i) In existing anonymous credential revocation systems, the revocation authority can link the transactions of any user in a subset T of users in $O(\log |T|)$ fake failed sessions.

(ii) A concern about the DLREP-I anonymous credentials system described in [1] and [2].

Keywords: Anonymous credential system, trust certification, DLREP-I

1 Introduction

An *Anonymous Credential (AC)* is a vector of attributes certified by a trusted certification authority. It can be verified by anybody, such that the holder (the "prover") can selectively disclose its components. For example, she may choose to disclose only the fact that she has a valid driver's license, but not her age. Some of the required properties of an AC are (there are many other requirements, but in this paper we discuss only the following) :

(i) It should be possible for a user to selectively disclose attributes.

(ii) An AC must be hard to forge,

(iii) A user's transactions must be *unlinkable*, and

(iv) An AC must be *revokable*.

[1] Microsoft Research
E-Mail: {rbhaskar,satya,peter.montgomery,venkie,yacov}@microsoft.com

[2] College of Computing, Georgia Tech, Atlanta, GA. Work done while at Microsoft Research.
E-Mail: karthe@gatech.edu

doi:10.1016/j.entcs.2007.12.023

In the literature, "revocation" has more than one meaning. For example, *anonymity revocation* has the obvious meaning: the user is anonymous until *Revocation Authority (RA)* exposes her identity. Usually this is followed by entering her ID into a revocation list. Revocation may be *partial* or *total.* In the former a subset of the entries in the vector is revoked, while in the latter the whole vector is revoked (i.e., the user is revoked).

We distinguish between *personal* and *impersonal* credentials. The former include driver's license, passports, attestation that a person is older than 21 years, etc. The latter include impersonal transferable objects such as e-cash and concert tickets. We focus on the former. A personal credential must be tied to an ID. It is not the bearer of the "older than 21" credential who is certified as such; it is some particular person. Some notion of the identity of that person must be encoded into the credential, even if the intended use of this credential is with minimal exposure of other personal information. For example, a user may want to enter a bar proving just that she is old enough. She needs at the minimum a photo ID that includes the credential "older than 21." And if she is caught misbehaving the police may ask for additional identifying information that she can then disclose. Impersonal credentials are also revokable. Our discussion about revocation is focused on personal credentials, and applies to partial and total revocations.

Ideally, RA should not be able to revoke capriciously. In the case of *impersonal* credentials there is a beautiful example where this is made possible. In the 80s Chaum ([8] and [9]) showed an anonymous e-cash system where double spent coins (double spending is a crime that should lead to revocation) expose some secrets about the user. This can be used for anonymity revocation. There were many followers of this genre. Unfortunately, we do not know of any other example in the context of *personal* credentials. Here it is usually up to the RA to decide on revocation, and he must be trusted not to abuse his power.

Our *linkability attack* applies to all existing anonymous *personal* credential systems known to us. We show that revocation (partial or total) implies that a malicious RA can link user's transactions. Specifically, we show that a malicious Revocation Authority (RA) can link the transactions of anybody in a set T of users in $O(\log(|T|))$ fake failed verification attempts, in which RA manipulates the revocation list, L. One may be tempted to assume that if RA is not trusted not to link, then all we have to do is tell the user to inspect the revocation list, L. First, in some systems the list is encoded in such a way that the user may not be able to distinguish malicious manipulation from normal dynamic behavior. Second, assume that the user can tell the difference. Consider a 1^{st} run with good L followed by a 2^{nd} run with a bad L. The user will cooperate with the 1^{st} and refuse to cooperate with the 2^{nd}. This would lead to a similar effect. Realizing that RA must be trusted not to link user's transactions (a very natural assumption) may help to simplify the system. There is a cost in trying to design a system that can do with less trust.

In addition, we raise a concern about the DLREP-I anonymous credential system described in [1] and [2]. Specifically, we show that at the end of the issuing protocol there, the Certification Authority (CA) is still left in an ambiguity about which credentials he has just certified.

2 Linkability attack by a corrupt Revocation Authority

We now show a general attack based on a collusion between a corrupt revocation authority and verifiers. The attack assumes a general approach to revocation observed in several existing systems. In the appendix (section 4), we illustrate how the systems in [7], [5], and [4] fit this description.

A general revocation system: There exists a public revocation list L (or its complement). L may be coded with some secret keys. We use $C(L)$ to denote the coded list. Everybody can read $C(L)$, but only RA can write (add and delete) in $C(L)$. Note that since $C(L)$ may be encoded with secret keys the ability to read it does not imply ability to interpret it. The prover, P, and the verifier, V, engage in a dialog, at the end of which V decides if $P \in L$. Ideally it is proved in Zero Knowledge ([7]). The prover may have many pseudonyms that change over time, and some may coexist simultaneously. Once a user is revoked, all her pseudonyms are revoked. Namely, suppose that user i has n pseudonyms p_{ij} , $j = 1, 2, ..., n$.

The revocation condition: If for some fixed i and for any $j = 1, ...n$, L includes some manifestation of p_{ij} then all the n p_{ij} corresponding to the same i are revoked.

We show that a malicious RA can collude with verifiers to run repeated verifications, e.g. by faking failure, that lead to linkability, of sessions of p_{ij} and p_{ik} for any $1 \leq j, k \leq n$, for all i. Our result applies to revocation list or its complement. Without loss of generality we proceed assuming the former. Let S_1, S_2 denote sets, and as usual, let $S_1 \backslash S_2 = \{x \mid x \in S_1 \wedge x \notin S_2\}$.

Suppose that a malicious RA wants to decide if two sessions performed under pseudonyms p and q are in fact done by the same user, namely that there exist i, j, k such that $p = p_{ij}$, and $q = p_{ik}$. In this setting the first session (with p) has already completed, and the session with q is about to begin. Let $\{u\}$ be some manifestation of the user identity, which RA uses to revoke the user whose pseudonym is p.Possible candidates include user identity, long-term user secret or list of all user pseudonyms. All these examples yield trivial linkage (without our attack). Our attack covers all manifestations, even those that by themselves do not give direct linkage. *We do not assume that the manifestation derived from one user pseudonym is equal to the manifestation derived from another pseudonym of the same user (had it been the case there would be a trivial linkage via* $\{u\}$). RA in collusion with a verifier will challenge q twice: once with $C(L \backslash \{u\})$ and then, with $C(L \cup \{u\})$. Then, q and p are linked iff q is proved included in the second, and not included in the first (we assume here that L is a revocation list. If it is its complement then it is the other way around).

More generally, let T be a (manifestation of) a set of users that RA and V want to trace. If they want to find out if a prover belongs to T or not, then two attempts are sufficient - one with $C(L\backslash T)$, and one with $C(L \cup T)$. However, to link the transactions of any user in T, the collusion requires a more elaborate attack extending the above idea with binary search. In that case the attack takes $O(\log |T|)$ transactions.

Remark 2.1 One may argue that a user who is first challenged with a $C(L)$ that does not implicate him, and then with a $C(L')$ that does, may refuse to cooperate in the latter. This refusal achieves the same effect as a failed cooperation. The more general case, that requires a few attempts, is more suspicious, but the point is that we either have to include explicit protections against such attacks, or trust RA not to misbehave this way.

Remark 2.2 In our linkability attack a corrupt RA must betray her primary responsibility of being honest about whom she revokes. One may argue that if this is the case then the system is doomed anyway. However, there is a difference between a permanent false revocation and a *transient* false revocation, as in our attack, that can be "explained" away later (if noticed at all) as some natural malfunction. The checks and balances (properly addressing complaints) that must be included in a well designed system help mitigate the former, but not the latter.

Remark 2.3 There are hopes in the folklore that technology may be invented to eliminate the need to trust RA not to link transactions. So far all the general purpose personal anonymous credentials systems that we reviewed succumb to this attack[3] . It seems that the ordinary trust (not to make permanent false revocations) is insufficient, and additional trust is needed not to engage in the above attack (using temporary revocations).

3 A concern about an issuing protocol

We show that at the end of the issuing protocol of the DLREP-I system [1] and [2] there is still ambiguity about which attributes were certified. We do not know how to exploit it in order to fool the verifier in a showing protocol. However, we think that both technically and legally, the issuer should have certainty about the attributes she signs at the end of the issuing protocol.

3.1 The system

The system is described in [1]. In particular it is sufficient to focus on Figs. 4.7, the issuing protocol, and the explanations surrounding it. For the sake of completeness here are the essentials (but for details look at the source). For a concise description of the same protocol and its context see also [2] fig 7 (note that h in [2] is h' in [1] fig 4.7).

Following the notation of [1], let q, p be large primes such that $q \mid (p-1)$.The Certification Authority (CA) has:

Secret keys : $x_0, y_i, i = 1, 2, ...l$ all in $\mathbb{Z}_q^*$.

[3] Anonymity revocation of double spent e-coins is not included.

Public keys: g_0 is a generator of a multiplicative subgroup of order q of the integers $\bmod\, p$. $h_0 \equiv g_0^{x_0} \bmod p$, $\quad g_i \equiv g_0^{y_i} \bmod p$.

The user's attributes that CA is supposed to certify by signing are x_i, $i = 1, 2, ...l$. The user's "public key" that encompasses those attributes is

$$h \equiv (\prod_{i=1}^{l} g_i^{x_i}) \bmod p$$

and the "blinded public key" is

$$h' \equiv (hh_0)^{\alpha_1} \bmod p,$$

where α_1 is a blinding element randomly chosen by the user from $\mathbb{Z}_q^*$.

A signature by CA on h' is a pair (c_0', r_0') for which the following *verification condition* holds:

$$c_0' = H(h', g_0^{c_0'} h'^{r_0'})$$

The signing protocol is:

User		**CA**
		$w_0 \in_R \mathbb{Z}_q$
	$\leftarrow$	$a_0 \equiv g_0^{w_0} \bmod p$
$\alpha_{1,2,3} \in_R \mathbb{Z}_q$		
$\alpha_1 \neq 0$		
$h' \equiv (hh_0)^{\alpha_1} \bmod p$		
$c_0' = H(h', g_0^{\alpha_2}(hh_0)^{\alpha_3} a_0)$		
$c_0 \equiv c_0' - \alpha_2 \bmod q$	$\rightarrow$	
	$\leftarrow$	$r_0 \equiv (w_0 - c_0)/(x_0 + \sum_{i=1}^{l} x_i y_i) \bmod q$
$a_0 \equiv g_0^{c_0}(hh_0)^{r_0} \bmod p?$		
$r_0' \equiv (r_0 + \alpha_3)/\alpha_1 \bmod q$		

Claim 3.1 *Suppose that* $(\alpha_1, \alpha_2, \alpha_3)$ *are the blinding elements in a run of the protocol with attributes* $(x_1, ...x_l)$, *and the protocol produces certificate* (c_0', r_0') *for user's "blinded public key"* h'. *Then for all* β *the same certificate with the same* h' *is consistent with attributes* $(\beta x_1, ...\beta x_l)$, *blinding elements* $(\alpha_1/\beta, \alpha_2, \alpha_3/\beta)$, *and* $h_0 \to h_0^{\beta}$.

Proof. In user's side of the protocol, do the following modifications: (i) Replace x_i with $x_i\beta$, $i = 1, 2, ...l$. (shorthand: $x_i \to x_i\beta$), (ii) $\alpha_1 \to \alpha_1/\beta$; $\alpha_3 \to \alpha_3/\beta$; a_0 and α_2 remain unchanged; (iii) $h_0 \to h_0^{\beta}$. The result of these modifications is that

h', c_0', c_0 remain unchanged. Therefore, the original r_0 divided by β can serve as the new r_0 $(r_0 \rightarrow r_0/\beta)$. The check for a_0 passes ok (although this is not important), and r_0' remains unchanged. The verification condition $c_0' = H(h', g_0^{c_0'} h'^{r_0'})$ holds after the attack (since c_0', r_0' and h' remain unchanged). □

We note that h_0 is not controlled by the attacker, however, CA does not use h_0 in the issuing process, hence at the end of issuing, CA is not aware of any problem. It is only at the showing protocol that the verifier uses h_0.

4 Concluding Remarks

We presented two concerns about existing anonymous credential systems, emphasizing the need for additional debate about the definitions of security requirements for anonymous credential systems. It appears that a compromise is required, either in the security requirements or in the amount of trust bestowed on the participants, in order to achieve a practical and efficient anonymous credential system.

References

[1] Stefan Brands: Rethinking public key infrastructure and Digital Certificates; The MIT Press, Cambridge Massachusetts, London England. ISBN 0-262-02491-8

[2] Stefan Brands: A Technical Overview of Digital Credentials; February 2002 (was a white paper in credentica.com).

[3] Stefan Brands: Non Intrusive Identity management; in 3rd Annual PKI R&D Workshop (Keynote Address), http://middleware.internet2.edu/pki04/proceedings/cross_domain_identity.pdf

[4] Stefan Brands, Liesje Demuynck, and Bart De Decker: A Practical System for Globally Revoking the Unlinkable Pseudonyms of Unknown Users, 12th Australasian Conference on Information Security and Privacy, http://www.idcorner.org/wp-content/ACISP2007.pdf

[5] Ernie Brickell, Jan Camenisch, and Liqun Chen: Direct Anonymous Attestation, CCS'04, http://eprint.iacr.org/2004/205/

[6] Jan Camenisch and Anna Lysyanskaya: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation, in B. Pfitzmann (Ed.): EUROCRYPT 2001, LNCS 2045, pages 93-118, 2001.

[7] Jan Camenisch and Anna Lysyanskaya: Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology, 2002, pages 61-76, Springer-Verlag.

[8] David Chaum: Blind Signatures for Untraceable Payments, Advances in Cryptology, Proceedings of Crypto 82, D. Chaum, R.L. Rivest, and A.T. Sherman (Eds.), Plenum, pp. 199-203.

[9] David Chaum: Security Without Identification: Transaction Systems to Make Big Brother Obsolete , (invited) Communications of the ACM, vol. 28 no. 10, October 1985 pp. 1030-1044.

[10] David Chaum, Jan-Hendrik Evertse: A Secure and Privacy-Protecting Protocol for Transmitting Personal Information Between Organizations, Advances in Cryptology: CRYPTO '86, A.M. Odlyzko (Ed.), Springer-Verlag, pp. 118-167.

[11] Lidong Chen: Access with Pseudonyms. Cryptography: Policy and Algorithms 1995: 232-243.

[12] Ivan Damgård: Payment Systems and Credential Mechanisms with Provable Security against Abuse by Individuals, Advances in Cryptology - CRYPTO '88: Proceedings, LNCS Vol 403, 1990, pp. 328-335.

[13] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf: Pseudonym Systems, in Proceedings of the Sixth Annual Workshop on Selected Areas in Cryptography (SAC '99), LNCS Vol 1758, 1999.

Appendix: Trust assumption about the Revocation Authority

Anonymity revocation is cleanly doable in some cases. For example, some anonymous e-cash systems have provisions for exposing encoded secrets iff a coin is double spent ([8] and [9]). In such systems we do not need to put extra trust in RA. However, this isn't the case with general purpose personal credentials, where RA must be trusted not to revoke capriciously. We proceed to describe the revocation mechanisms of some anonymous credentials systems ([7], [5], [4]), in order to show that they fit the general description that we gave in section 2.

The revocation mechanism in [7] uses *dynamic accumulators.* A dynamic accumulator allows a set of values L (in our case L is the set of unrevoked users) to be accumulated into a value λ. Then a witness ω can be given to prove that a value x is accumulated into λ. The verifier evaluates a polynomial-time function g to check that $g(\omega, x) = \lambda$. The security of a dynamic accumulator guarantees that it is infeasible for an adversary to generate a witness for a value that is not accumulated. Furthermore, given a secret key (trapdoor information), a value contained in a dynamic accumulator can be "deleted." In [7], a unique value is associated with each credential at the issue stage; this unique value is not known to the CA/issuing organization. This value and subsequent values corresponding to all unrevoked credentials are accumulated into the list L and the accumulated value λ is made public. During a reveal transaction, the user demonstrates that the unique value in her credential is contained in the accumulator by proving the existence of the witness through a zero-knowledge protocol. When the revocation authority wants to revoke a credential, he uses his secret key to delete the unique value contained in that credential. After a sequence of add/delete updates, users can recompute their witnesses even without the secret key. They can do this by looking up a publicly visible list (e.g. maintained by the RA) for added and deleted unique values from the time they last checked.

To see how Brickell's system [5] meets that description, note that the user uses a pseudonym N_V for each verifier, such that $N_V = \zeta^f$, where f is a user-specific secret value and ζ is a suitable random generator of a group in which the discrete log problem is hard; the RA publishes a list of *revoked* pseudonyms $L = \{N_{V_i}\}$. Any prover who claims that his pseudonym N_V has not been revoked proves in Zero-Knowledge, that $\log_\zeta N_V \neq \log_{\zeta_i} N_{V_i}$ for all N_{V_i}'s in the list. Thus, the manifestation function here is a modified form of discrete log.

Brand's revocation method [4] can also be reduced to the above standard description. In his system, each service provider S_i has a hot list L_i. L_i contains a list of pseudonyms whose corresponding users have misused the system. If the user's pseudonyms are denoted by $\{d_1, ..., d_l\}$ and if she wants to access the service provided by S_j, then she claims that none of $\{d_1, ..., d_l\}$ belong to the hot lists of any of the service providers. To verify this claim, the verifier computes polynomials such that all the hot-listed pseudonyms by all service providers are roots of at least one of these polynomials. Now the user proves that none of her pseudonyms are roots of any of the polynomials computed by the verifier. Thus, the hot lists can be seen as the revocation list, while the manifestation states that a pseudonym d_j of a user with service provider S_j, is not a root of any of the polynomials computed by the verifier.

The attack in Section 2 applies to all these systems.