

On the Security of *Homage* Group Authentication Protocol

Éliane Jaulmes and Guillaume Poupard

DCSSI Crypto Lab
18, rue du docteur Zamenhof
F-92131 Issy-Les-Moulineaux
email: eliane.jaulme@wanadoo.fr Guillaume.Poupard@ens.fr

Abstract. This paper describes two attacks on an anonymous group identification scheme proposed by Handley at *Financial Crypto 2000*. The first attack enables to forge valid proofs of membership for any secret key. As a consequence, any user, registered or not, can be properly authenticated by the group manager. The second attack enables the authority to recover the identity of any user who authenticates. Those two attacks can be very easily conducted in practice, without any heavy computation. Those attacks can be fixed with simple modifications and additions to the protocol but we think that the technique used to issue certificates is conceptually flawed and we propose a way to repair this phase of the protocol using zero-knowledge proof techniques.

Keywords. Anonymity, group authentication, cryptanalysis.

1 Introduction

The problem of secure identification was first introduced by Feige, Fiat, Shamir in Stoc 86 [9]. Later the problem of anonymity in group identification was first posed and studied by A. De Santis, G. Di Crescenzo, G. Persiano and M. Yung [18]. These protocols are immediately converted in perfectly anonymous group identification protocols.

Then, other perfectly anonymous group identification protocols were given by A. De Santis, G. Di Crescenzo and G. Persiano [17] using only quadratic residuosity as the underlying assumption, and by D. Boneh and M. Franklin [3].

Informally speaking, an anonymous group identification protocol is a scheme that enables previously registered users to convince an authority they belong to a specified group without revealing any information about their identity. Such protocols can be designed using public key cryptography tools. During a first phase, users prove their identity to an authority who issues certificates. Then, when a user wants to be authenticated without revealing his identity, he proves that he knows a certificate and convinces the authority that he has been properly registered.

A group authentication protocol has to satisfy some security properties: legitimate users must always be correctly authenticated, non-authorized people must

always be rejected; it should be impossible for the authority to uncover any information on the user who authenticates or link different authentications...

The notion of anonymous group identification is closely related to other notions such as group signature [6, 7, 4, 2, 1] and identity escrow [13], which provide a revocable anonymity, and to multisignatures [15] that enable people to sign messages for a group.

Our results

This paper describes two attacks on an anonymous group identification scheme proposed by Handley [11] and that we recall in section 2. The first attack, described in section 3, enables any registered user to forge valid proofs of membership for any secret key. As a consequence, any user, registered or not, can be properly authenticated by the group manager. Notice that this specific attack is fixed in Hanley's final paper [12].

The second attack, described in section 4, enables the authority to recover the identity of any user who authenticates. This attack cannot be detected and the computational effort for the authority is linear in the number of registered users.

Even if those attacks may be fixed by modifications and additions to the protocol, we think that the technique used to issue certificates is conceptually flawed and that the choice of the parameters is of crucial importance. We propose in section 5 a way to repair the protocol using zero-knowledge proof techniques but we do not claim that the resulting protocol is secure.

2 Description of *Homage*

Throughout this paper, we use the following notations: for any integer n ,

- we use \mathbb{Z}_n to denote the set of integers modulo n ,
- we use \mathbb{Z}_n^* to denote the multiplicative group of invertible elements of \mathbb{Z}_n ,
- we use $\varphi(n)$ to denote the Euler totient function, the cardinality of \mathbb{Z}_n^* .

The *Homage* group authentication protocol [11] consists of two main parts: registration and anonymous authentication. We recall the protocol with the characters Alice and Bob; Alice is a user, member of the group, and Bob is the authority who issues certificates in the first phase and authenticates users in the second one.

2.1 The setup of the scheme

During the *Homage* setup phase, the authority generates the following parameters:

- p a public prime integer,

- g a public generator of the multiplicative group \mathbb{Z}_p^* ,
- $u \in \mathbb{Z}_{p-1}^*$ a public constant,
- z and w two private keys kept secret by the group authority (it is specified in [12] that $z \in \mathbb{Z}_{p-1}^*$ and $w \in [1, p-2]$),
- $v = u^w \bmod p-1$ a public key of the authority.

In order to make the discrete logarithm problem intractable in \mathbb{Z}_p^* , it is required that $(p-1)/2$ should have few prime factors and that all of them should be large. It is specified that the size of p should be at least 2048 bits.

Furthermore, each user has got a private key $x \in \mathbb{Z}_{p-1}$ and the related public key $y = g^x \bmod p$. The public key y is associated with the identity of the user; it is used for group authentication but should also be used for other protocols such as personal identification (for example using the Schnorr scheme [19]), signature (DSA [14]) or encryption (El Gamal [8]). This surprising requirement is proposed as an efficient way to provide “strong dissuasion”, i.e. to avoid that legitimate users of the system give their secret data to unauthorized people. We don’t argue on such a problem that seems closely related to the anonymity the system wants to provide and consequently to the absence of an identity recovery mechanism.

2.2 Registration of a member

Alice, identified by her public key y , first registers and obtains a group certificate issued by Bob. We assume that Bob is convinced that Alice is a legitimate member of the group and that her public key is y . He chooses a random number $a \in \mathbb{Z}_{p-1}^*$ and computes the two following values:

- $\alpha_1 = (gy^z)^a \bmod p$
- $\alpha_2 = a^w \bmod p-1$,

that he sends to Alice as her certificate.

2.3 Anonymous authentication

Alice, owning certificate (α_1, α_2) , wants to convince Bob she is a member of the group without revealing her identity:

- she chooses two random numbers b and c ,
- she computes
 - $\beta_1 = \alpha_1^{c u^b} \bmod p$
 - $\beta_2 = \alpha_2 v^b \bmod p-1$
 - $\beta_3 = g^c \bmod p$,
- she sends β_1 , β_2 and β_3 to Bob,
- Bob computes
 - $\gamma_1 = \beta_2^{\frac{1}{w}} \bmod p-1$
 - $\gamma_2 = \beta_1^{\frac{1}{\gamma_1}} \bmod p$
 - $\gamma_3 = \left(\frac{\gamma_2}{\beta_3}\right)^{\frac{1}{z}} \bmod p$,
- Alice finally proves to Bob that she knows the discrete logarithm x of γ_3 in basis β_3 . The author of *Homage* proposes in [11] a zero-knowledge protocol to achieve this; notice that this protocol is exactly the Schnorr scheme [19] with challenges chosen in $\{0, 1\}$.

Completeness of the authentication protocol. If both parties are honest in their computations, i.e. follow the previously described protocol, we should have the following:

- $d = u^b \bmod p - 1$
so $d^w = u^{wb} = v^b \bmod p - 1$
- $\beta_1 = \alpha_1^{cd} = (gy^z)^{acd} \bmod p$
- $\beta_2 = \alpha_2 v^b = \alpha_2 d^w = (ad)^w \bmod p - 1$
- $\beta_3 = g^c \bmod p$
- $\gamma_1 = \beta_2^{\frac{1}{w}} = ((ad)^w)^{\frac{1}{w}} = ad \bmod p - 1$
- $\gamma_2 = \beta_1^{\frac{1}{ad}} = ((gy^z)^{acd})^{\frac{1}{ad}} = (gy^z)^c \bmod p$
- $\gamma_3 = (\frac{\gamma_2}{\beta_3})^{\frac{1}{z}} = \left(\frac{(gy^z)^c}{g^c}\right)^{\frac{1}{z}} = (y^{zc})^{\frac{1}{z}} = y^c = g^{xc} = \beta_3^x \bmod p,$

and finally the discrete logarithm of γ_3 in basis β_3 must be the user's secret key x .

In other words, this means that the triplet $(\beta_1, \beta_2, \beta_3)$ must satisfy the equation:

$$\beta_1 = (\beta_3^{1+xz})^{\beta_2^{\frac{1}{w}}} \bmod p$$

Consequently, the knowledge of four numbers $\beta_1, \beta_2, \beta_3$ and x satisfying the above equation enables their owner to be authenticated.

2.4 Soundness of a certificate

As explained in [11], a dishonest group manager can break the anonymity of authentication if he issues bad certificates, i.e. certificates that are not computed using the formula $((gy^z)^a \bmod p, a^w \bmod p - 1)$ or correctly computed but using bad secret keys w and z . As a consequence, when Alice receives her certificate (α_1, α_2) , she has to make sure that Bob has not cheated. To do this, it is suggested in [11] that she goes through the verification protocol with Bob. The only difference is that, at the end, Bob sends Alice the number γ_3 he has computed and Alice verifies that $\gamma_3 = y^c \bmod p$ (see section 2.3).

Furthermore, this verification of the soundness of the certificate should be conducted in a way such that Bob has no way of guessing who Alice is. To achieve this, the author of *Homage* suggests the use of a third party acting as an anonymizer.

Note. Handley seems to have felt the danger of revealing γ_3 and replaced it by an hash value $H(\gamma_3)$ in the final paper [12]. Consequently, Alice's verification becomes $H(\gamma_3) = H(y^c \bmod p)$. The following attack is fixed by such a modification, even if the principle of looking for z^{th} roots modulo p remains the main way to forge certificates.

3 How to forge valid proofs of membership

3.1 Basic idea

Assume we know a pair (n, m) such that $m = n^z \pmod p$ and that we want to construct a valid proof of membership. Remember that we know a pair (u, v) such that $v = u^w \pmod p$. We choose a random x and we compute the three values β_1 , β_2 and β_3 that will be sent to Bob as follows:

- randomly choose $b \in \mathbb{Z}_{\varphi(p-1)}$ and $c \in \mathbb{Z}_{p-1}$,
- compute $\delta = n^c \times m^{x^c} \pmod p$,
- compute $\beta_1 = \delta^{u^b} \pmod p$,
- compute $\beta_2 = v^b \pmod p - 1$,
- compute $\beta_3 = \frac{\delta}{m^{x^c}} = n^c \pmod p$.

When Bob verifies the certificate, he obtains:

- $\gamma_1 = \beta_2^{\frac{1}{v}} = v^{\frac{b}{w}} = u^b \pmod p - 1$
- $\gamma_2 = \beta_1^{\frac{1}{\gamma_1}} = \left(\delta^{u^b}\right)^{\frac{1}{u^b}} = \delta \pmod p$
- $\gamma_3 = \left(\frac{\gamma_2}{\beta_3}\right)^{\frac{1}{z}} = \left(\frac{\delta}{m^{x^c}}\right)^{\frac{1}{z}} = m^{\frac{x^c}{z}} = n^{x^c} \pmod p$

Finally, it is easy to prove to Bob that we possess the discrete logarithm x of $\gamma_3 = (n^c)^x$ in basis $\beta_3 = n^c$ since we have chosen such an x . Consequently, we can forge valid proof of membership for any secret key x provided we know a pair $(n, n^z \pmod p)$, i.e. a z^{th} root modulo p .

3.2 Finding z^{th} roots modulo p

Preliminary remark. Let us first remark that the pair $(1, 1)$ is a suitable couple (n, m) and that one of the pairs $(-1, -1)$ and $(-1, 1)$ is also suitable. However if these pairs are used to forge proof of membership Bob can observe that γ_3 and β_3 are equal to 1 or -1 and can consequently easily detect the forgery. Anyway, this has to be added in the protocol.

Verification of soundness leaks z^{th} roots. In section 2.4, we have seen that, in the original protocol [11], when Alice receives her certificate, she goes with Bob through a verification protocol where, at the end, Bob sends her the γ_3 he has computed.

Suppose now that, instead of following the protocol, Alice proceeds as follows in order to make Bob compute and reveal a z^{th} root of a randomly chosen integer $m \in \mathbb{Z}_p^*$. Firstly she randomly chooses $b \in \mathbb{Z}_{\varphi(p-1)}$ and $\delta \in \mathbb{Z}_p^*$. Then she sends β_1 , β_2 and β_3 to Bob, where:

- $\beta_1 = \delta^{u^b} \pmod p$,
- $\beta_2 = v^b \pmod p - 1$,

- $\beta_3 = \frac{\delta}{m} \bmod p$.

Following the procedure described in section 2.3, Bob computes

- $\gamma_1 = \beta_2^{\frac{1}{w}} = v^{\frac{b}{w}} = u^b \bmod p - 1$,
- $\gamma_2 = \beta_1^{\frac{1}{\gamma_1}} = \left(\delta^{u^b}\right)^{\frac{1}{u^b}} = \delta \bmod p$,
- $\gamma_3 = \left(\frac{\gamma_2}{\beta_3}\right)^{\frac{1}{z}} = \left(\frac{\delta}{\frac{\delta}{m}}\right)^{\frac{1}{z}} = m^{\frac{1}{z}} \bmod p$.

and, as proposed in [11], he sends γ_3 to Alice so that she can verify he has not cheated. Consequently, Alice obtains a z^{th} root of m and just tells Bob she is satisfied. Then, using the cheating strategy we have described above, she can be authenticated for any secret key x .

Note. Replacing γ_3 by $H(\gamma_3)$, as proposed in [12], is a simple fix to this way of finding z^{th} roots modulo p .

4 How to break the anonymity of authentication

We now demonstrate how a dishonest authority can break the anonymity of authentication. We propose two attacks; the first one shows that the modulus p should be a safe prime and the second one shows how a dishonest authority can recover the identity of any authenticating user, even if p is a safe prime.

Attack 1. The basic idea of the first attack is to choose a modulus p such that $(p - 1)/2$ has more than one prime factor: $p = 1 + 2 \times \prod_{i=1}^{\eta} q_i$. Then, the authority can distinguish 2^η different subgroups of users:

- first the authority associates to each subgroup a binary vector $v = (v_1, \dots, v_\eta)$ of length η ,
- when a user registers, the authority chooses a random number a' and computes $a = a' \times \prod_{i=1}^{\eta} q_i^{v_i} \bmod p - 1$ according to the subgroup the user belongs to; then certificates are regularly computed,
- finally, when a user wants to be authenticated, the order of β_1 is a product of some of the q_i s that reveals the vector v and consequently the subgroup of the user.

As an example, if p is 2048 bits long, the authority can choose $(p - 1)/2$ with $\eta = 10$ prime factors and consequently distinguish more than 1000 different subgroups! However, such an attack can be easily avoided, just verifying that the second part α_2 of any certificate is relatively prime with $p - 1$. Anyway, we think that using a safe prime as public modulus is a good precaution.

Attack 2. We now describe a much more serious attack; let us assume that the modulus p is a safe prime $p = 2q + 1$ with q a prime integer and that $q = 2r_1r_2 + 1$ with r_1 and r_2 two large prime numbers. The set \mathbb{Z}_{p-1}^* of invertible elements modulo $p - 1$ has $\varphi(p - 1) = \varphi(2q) = q - 1 = 2r_1r_2$ elements. It is well known that the multiplicative order of each element is a divisor of $\varphi(p - 1) = 2r_1r_2$ and that there are $\varphi(d)$ elements of order d . We obtain the following repartition for the elements of \mathbb{Z}_{p-1}^* :

multiplicative order ω	1	2	r_1	r_2	$2r_1$	$2r_2$	r_1r_2	$2r_1r_2$
number of elements of \mathbb{Z}_{p-1}^* of order ω	1	1	$r_1 - 1$	$r_2 - 1$	$r_1 - 1$	$r_2 - 1$	$(r_1 - 1) \times (r_2 - 1)$	$(r_1 - 1) \times (r_2 - 1)$

Consequently, a randomly chosen element in \mathbb{Z}_{p-1}^* has overwhelming probability $\left(1 - \frac{1}{r_1}\right) \times \left(1 - \frac{1}{r_2}\right) \approx 1 - \frac{4}{\sqrt{p}}$ to be of large order $q - 1$ or $(q - 1)/2$. Furthermore, if the factorization of $q - 1$ is unknown, we do not know any efficient algorithm to compute the order of elements in \mathbb{Z}_{p-1}^* .

Let us consider an authority who chooses u of order r_1 in \mathbb{Z}_{p-1}^* ; consequently, $v = u^w \bmod p - 1$ is also of order r_1 if $w \not\equiv 0 \pmod{r_1}$, i.e. with overwhelming probability.

Then, following the protocol, the authority can recover the identity of any authenticating users. Just notice that $\beta_2 = \alpha_2 v^b \bmod p - 1$ is of order r_1r_2 or $2r_1r_2$ with very high probability but $\beta_2/\alpha_2 = v^b$ is of order r_1 . Consequently, if the authority wants to test if β_2 comes from a user who received α'_2 as second part of his certificate, he just checks if the order of β_2/α'_2 is of “pathological” order r_1 . Indeed, if α'_2 corresponds to the good user, β_2/α'_2 is equal to v^b that is of order r_1 whereas if α'_2 belongs to another user, β_2/α'_2 is of order r_1r_2 or $2r_1r_2$ with very high probability.

Consequently, the authority recovers the identity of the user in time complexity linear in the number of registered users: he just has to check the order of β_2/α'_2 for all α'_2 he has delivered.

Notice that if the authority can choose the modulus p and the parameter u as he wants, this attack cannot be detected. Furthermore, we have considered the case of a safe prime $p = 2q + 1$ such that $(q - 1)/2$ has two large prime factors but the attack can of course be applied in many other situations where the description is more difficult but the attack always as efficient.

5 How to repair the protocol

The main flaw of the *Homage* protocol comes from the need of a certificate verification phase. A modification that avoids the extraction of z^{th} roots described in section 3.2 simply consists of making Alice prove with a zero-knowledge protocol that she knows the discrete logarithm of γ_3 in basis β_3 before Bob reveals γ_3 .

Consequently, Bob is convinced that Alice already knows γ_3 and that he does not disclose any information when he gives it over to her.

However, a dishonest authority can use different secret keys z to compute certificates and then distinguish users during authentication. More precisely, the authority chooses various keys z_i for different subgroups of users and computes “correct” certificates with the formula $(\alpha_1, \alpha_2) = ((gy^{z_i})^a \bmod p, a^w \bmod p-1)$. Then, during authentication, he can check which z_i is used and consequently to which subgroup the user belongs. The repartee suggested in [11] is to use an anonymizer between users and the authority for the certificate verification phase. However, we believe that such a protection is not good since a user who wants to verify the correctness of his certificate, even using an anonymizer, is probably the user who last registered...! Consequently, we definitively think that the verification phase is the weak point of the protocol.

We are now going to propose a different approach based on classical zero-knowledge techniques and that seems to avoid many difficulties and that makes the first step in the direction of a provably secure variant of *Homage*.

How to avoid verification. The idea to avoid verification is to provide with each certificate (α_1, α_2) a proof that it has been correctly computed, i.e. that the authority knows an integer a such that $\alpha_1 = (gy^z)^a \bmod p$ and $\alpha_2 = a^w \bmod p-1$, and that the correct secret keys w and z have been used. This is achieved with the following modifications:

- the prime modulus p is a safe prime $p = 2q + 1$ with $q = 2r + 1$ a safe prime as well,
- g is an element of \mathbb{Z}_p^* of order q ,
- u is an element of \mathbb{Z}_q^* of order r ,
- another basis h of order q is chosen such that the discrete logarithm of h in basis g is unknown to the authority,
- the secret key z is committed with a public key $Y = g^z h^{z'}$ mod p where $z' \in \mathbb{Z}_q$ is a randomly chosen value,
- a commitment scheme is selected and we note $\text{commit}(x)$ the commitment of data x ; as an example of such a scheme, we can use Pedersen’s protocol [16] using parameters g , h and p .

Then the authority proves the knowledge of secret values w , z , z' and a such that the following four relations are valid:

$$\begin{aligned} \alpha_1 &= (gy^z)^a \bmod p, \alpha_2 = a^w \bmod p-1, \\ v &= u^w \bmod p-1 \text{ and } Y = g^z h^{z'} \bmod p \end{aligned}$$

We can use well known proofs derived from the Schnorr scheme [19] such as the proof of equality of discrete logarithm of Chaum and Pedersen [5] and we obtain the following protocol:

- the prover (the authority), chooses a random value $t \in \mathbb{Z}_q$ and computes

- $C_1 = \text{commit}(t)$
 - $C_2 = \text{commit}(t \times a \bmod q)$
 - $C_3 = (gy^z)^{ta} \bmod p$
 - $C_4 = (ta)^w \bmod q,$
- the prover sends C_1, C_2, C_3 and C_4 to the verifier (the registering user) who answers a challenge e randomly chosen in $\{0, 1\}$,
- if $e = 0$, the prover opens the commitment C_1 and reveals τ ; then he proves with subprotocol A (see fig.1) that $\log_\tau(C_4/\alpha_2) = \log_u(v)$; furthermore, the verifier checks that $C_3 = \alpha_1^\tau \bmod p$,
- if $e = 1$, the prover opens the commitment C_2 and reveals τ' ; then he proves that $\log_{\tau'}(C_4) = \log_u(v)$ (with subprotocol A, fig.1) and that $\log_{y^{\tau'}}(C_3/g^{\tau'})$ is equal to the secret key z committed into $Y = g^z h^{z'}$ (with subprotocol B, fig.2).

The probability of success of a cheating prover is smaller than $1/2$ so this elementary round has to be repeated n times in order to make the cheating probability smaller than $1/2^n$.

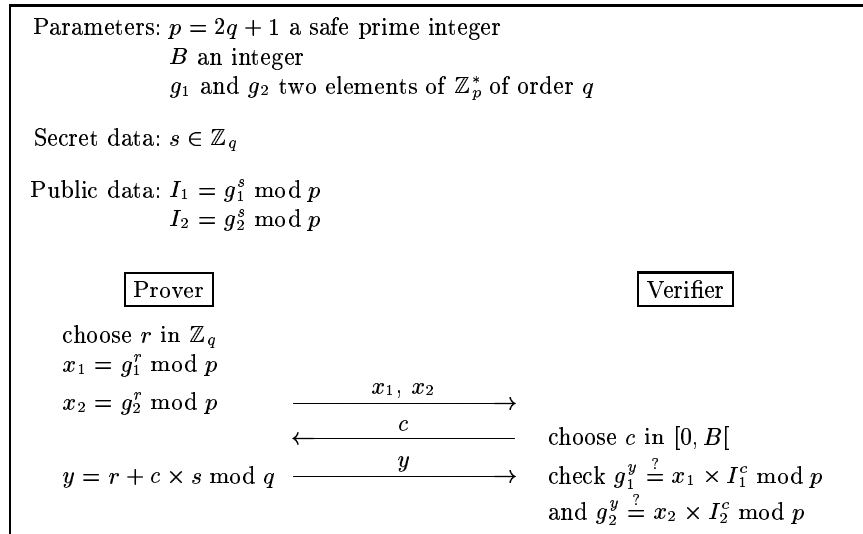


Fig. 1. Subprotocol A: proof of equality of discrete logarithms

Security analysis. Subprotocols A and B are complete and sound (see for example [10] and [9] for definitions); a prover who is accepted with probability larger than $1/B$ during the execution of subprotocol A can be used by an extractor which computes the common discrete logarithm s of I_1 and I_2 in basis g_1

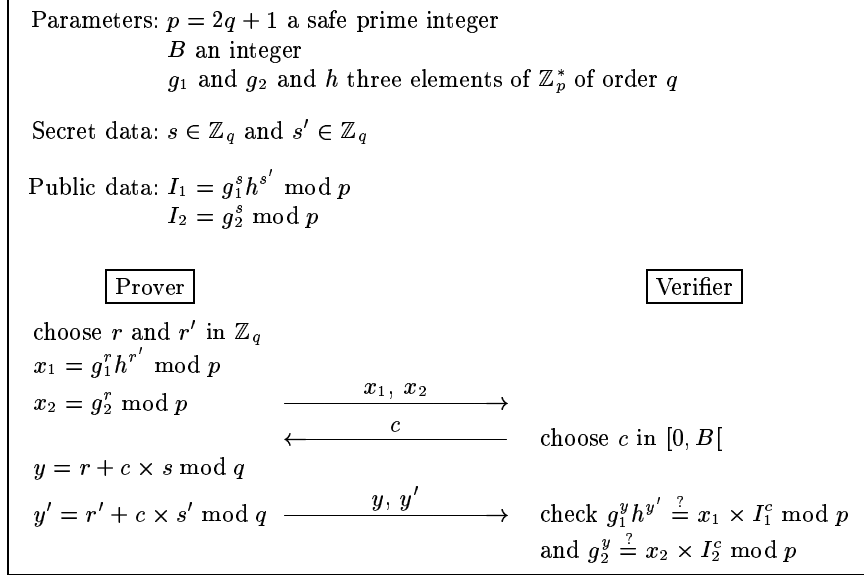


Fig. 2. Subprotocol B: proof of equality of partial discrete logarithms

and g_2 respectively. Similarly, s and s' can be computed using a prover accepted with probability $> 1/B$ in subprotocol B.

Those two protocols, like the Schnorr scheme, are not zero-knowledge when B is not polynomial because the communications cannot be simulated. Anyway, this does not lead to any known attack. Furthermore, if we want a zero-knowledge protocol, we can use a polynomial value for B and repeat the protocol ℓ times; the probability of success is smaller than $1/B^\ell$ and the simulation has polynomial complexity $O(\ell \times B)$. Another solution to obtain constant-round protocols is to have the verifier commit to his challenges by using, for instance, Pedersen's commitment scheme [16].

It is not difficult to prove that the protocol we propose is complete and zero-knowledge. The soundness is proved in the following way: if a prover is able to answer the two challenges $e = 0$ and $e = 1$ for the same commitments C_1, \dots, C_4 , we can extract w, z, z' and a such that $\alpha_1 = (gy^z)^a \bmod p$, $\alpha_2 = a^w \bmod p - 1$, $v = u^w \bmod p - 1$ and $Y = g^z h^{z'} \bmod p$. Consequently, if a prover is accepted with probability $> 1/2^n$, he can be used by an extractor to compute the four secret values.

Notes.

1. The basis h is used for the computation of public key Y in order to avoid revealing something like “ $Y = g^z \bmod p$ ”, i.e. a z^{th} root.
2. A practical way to be sure that the authority has not chosen a second basis h of known discrete logarithm in basis g is to derive the basis from a publicly verifiable pseudo-random generation algorithm.
3. The modulus p is a safe prime $p = 2q + 1$ so elements of \mathbb{Z}_p^* have order 1 (the unity), 2 (the element -1), q or $2q$. A simple way to generate an element g of order q is to choose a random element $x \in [2, p - 2]$; if x is a quadratic residue, let $g = x$ and otherwise let $g = x^2 \bmod p$. This avoids residual problems described in section 4 (attack 1).
4. The computation of a prime modulus p such that $q = (p-1)/2$ and $(q-1)/2$ are also prime integers can be done in reasonable time. Such modulus have already been used in protocols such as [20].

Acknowledgments

We would like to thank Giovanni Di Crescenzo for helpful comments on the bibliography related to group authentication.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In *Crypto '2000*, LNCS 1880. Springer-Verlag, 2000.
2. G. Ateniese and G. Tsudik. Some Open Issues and New Directions in Group Signature. In *Financial Cryptography '99*, LNCS. Springer-Verlag, 1999.
3. D. Boneh and M. Franklin. Anonymous Authentication With Subset Queries. In *Proceedings of 6th ACM-CCS*, pages 113–119. ACM press, 1999.
4. J. Camenisch and M. Michels. A Group Signature Scheme with Improved Efficiency. In *Asiacrypt '98*, LNCS 1514. Springer-Verlag, 1998.
5. D. Chaum and T. P. Pedersen. Wallet Databases with Observers. In *Crypto '92*, LNCS 740, pages 89–105. Springer-Verlag, 1992.
6. D. Chaum and E. van Heyst. Group Signatures. In *Eurocrypt '91*, LNCS 547, pages 257–265. Springer-Verlag, 1992.
7. L. Chen and T. P. Pedersen. New Group Signature Schemes. In *Eurocrypt '94*, LNCS 950, pages 140–155. Springer-Verlag, 1995.
8. T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *IEEE Transactions on Information Theory*, volume IT-31, no. 4, pages 469–472, july 1985.
9. U. Feige, A. Fiat, and A. Shamir. Zero-Knowledge Proofs of Identity. *Journal of Cryptology*, 1:77–95, 1988.
10. S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM journal of computing*, 18(1):186–208, february 1989.
11. B. Handley. Resource-Efficient Anonymous Group Identification. In *Preproceedings of Financial Cryptography 2000*, 2000.

12. B. Handley. Resource-Efficient Anonymous Group Identification. In *Financial Cryptography 2000*, LNCS. Springer-Verlag, 2001 (?). (*Personal communication from the author in January 2001*).
13. J. Kilian and P. Petrank. Identity Escrow. In *Crypto '98*, LNCS 1462, pages 169–185. Springer-Verlag, 1998.
14. NIST. Digital Signature Standard (DSS). Federal Information Processing Standards PUBLication 186, november 1994.
15. T. Okamoto. A digital multisignature scheme using bijective public-key cryptosystems. *ACM transactions on computer systems*, 6(4):432–441, 1988.
16. T.P. Pedersen. Non-Interactive and Information-Theoretic secure Verifiable Secret Sharing. In *Crypto '91*, LNCS 576, pages 129–140. Springer-Verlag, 1992.
17. A. De Santis, L. di Crescenzo, and G. Persiano. Communication-Efficient Group Identification. In *Proceedings of the 5th ACM-CCS*. ACM press, 1998.
18. A. De Santis, L. di Crescenzo, G. Persiano, and M. Yung. On Monotone Formula Closure of SZK. In *Proceedings of the 35th FOCS*, pages 454–465. IEEE, 1994.
19. C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *Crypto '89*, LNCS 435, pages 235–251. Springer-Verlag, 1990.
20. A. Young and M. Yung. Auto-Recoverable Auto-Certifiable Cryptosystems. In *Eurocrypt '98*, LNCS 1403, pages 17–31. Springer-Verlag, 1998.