Technical Report OUCS-2004-16

Attacks on Homage Anonymous Group Authentication Protocol

Authors:
Stewart Fleming, Sonil Gohil

Department of Computer Science, University of Otago

Status: Report of work done under Departmental summer bursary project December 2004-February 2005. Edited for consistency and with respect to further work done in preparing submission to Financial Cryptography conference

Attacks on Homage Anonymous Group Authentication Protocol

Department of Computer Science, University of Otago.

Sonil Gohil, Stewart Fleming

6th September 2004

Abstract

The Homage anonymous group authentication was originally proposed by Handley (2000). This report describes an implementation of the attacks on the protocol proposed by Jaulmes & Poupard (2001). All of the proposed attacks are shown to be valid, given the assumptions made by the authors.

We propose slight modifications to the original protocol that take all of Handley's assertions into account and eliminate the ability of the authority to cheat. We believe that this secures the basic protocol and we report on an implementation that demonstrates the protocol in action.

As an extension to the original protocol, we propose a number of options to incorporate biometric authentication into Homage without compromising anonymity and preserving efficiency. We critique the various options, showing which are and are not compatible with the basic objectives of Homage.

This report is an edited version of the original report submitted by Sonil Gohil (see separate Technical Report) to summarize the work done during summer bursary project in December 2003-January 2004. It incorporates the additional work that was done in refactoring the implementation and critiquing the biometric options that was done in preparation for publication in Financial Cryptography conference in 2005 (see separate Technical Report).

Introduction

The Homage protocol due to Handley (2000) allows an individual to be authenticated anonymously as a member of a group. The identity of the individual remains unknown to the authority when the user is being authenticated.

The anonymity provided by the protocol is important in many applications to prevent the leaking of personal information and to stop an authority from tracking down private and confidential information associated with the individual. It allows an organization to get the information that they require – that an individual belongs to one or more groups – without letting them have access to any information that would compromise the identity of the individual.

Homage satisfies the security properties required for anonymous group authentication. Non-members of the group are rejected and members are successfully authenticated without their identities being exposed. For this property, Homage does rely on non-transference of an individual's private key. In many situations, this is a concern and we address this issue later in this report.

Homage is also designed as a resource-efficient protocol. Interactions in the protocol require minimal computation and communication between the authority and the individual. All interactions are between the authority and one individual – either at registration or authentication.

Homage Group Authentication Protocol

The Homage protocol operates between a group authority and a number of members of that group. The authority issues certificates to group members that can be used to anonymously prove group membership.

Certificates are based on pseudonyms based on the public keys of group members, secret values known only to the group authority and random values specific to each group member. The group authority does not need to store any information specific to a group member, nor is any such information useful since the certificate requires knowledge of the private key of the group member to use.

The Homage protocol can be summarised as consisting of three separate activities: authorisation, testing the certificate and verification. Testing the soundness of the certificate was proposed so that the group member can be assured that the authority has not cheated in the generation of the certificate.

Definitions

Group Authority – an individual who controls a group and can issue certificates to users and authenticate them.

User – individual who is a member or prospective member of a group.

H(m) is a publicly-known secure hash function.

x is the private key held by the user,

$y = g^x \bmod p$ is the public key of the user; to be completely disassociated from the user, a pseudonym (g^c, g^{cp}) can be adopted and used.

z and w are two secret keys held by the authority such that $z \in Z^*_{p-1}$ and $w \in [1, p-2]$. (Jaulmes & Poupard (2001) make this claim on the basis of their personal communication with Handley and this confirms our own exchanges with the original author.)

p is a publicly-known prime modulus[1],

g is a publicly-known generator of Z^*_{p-1} as in the public key algorithm,

$u \in Z^*_{p-1}$, a public constant co-prime to $p-1$ and should be able to generate a large fraction, if not all of Z^*_{p-1}

$v = u^w (\bmod p-1)$ is a public key of the authority,

$a \in_R Z^*_{p-1}$ is a random number co-prime to p-1 chosen by the authority when registering a user,

$b \in_R \{1, p-1)$ and $c \in_R Z^*_{p-1}$ are random numbers chosen by the user at authentication time (different for each authentication attempt).

[1] Handley initially suggests in his paper that $\frac{p-1}{2}$ *should* be prime and then in personal communication defines that it *must* be.

$d = u^b (\mathrm{mod}\ p-1)$ is calculated by the user at authentication time and sent to the authority.

Registration

A prospective group member presents her public key[2] to the group authority to gain authorisation to become a member of the group. The group authority chooses a random value $a \in_R Z^*_{p-1}$ and then calculates $\alpha_1 = (gy^z)^a (\mathrm{mod}\ p)$ and $\alpha_2 = a^w (\mathrm{mod}\ p-1)$. The pair (α_1, α_2) is the certificate issued to the group member. The certificate is used in anonymous authentication. In order to be authenticated, a group member must provide values based on a valid certificate and demonstrate knowledge of the private key on which the certificate is based. Registration requires 4 modular exponentiations.

Verification

To verify that the group authority has not cheated in the issuing of the certificate, the group member should check that it has been computed correctly. The group member goes through the protocol with the group authority as for authentication (see below), but the final challenge is omitted. Instead, if the group authority can tell the group member what $g^{cx} = y^c$ is, the group member accepts that the certificate has been correctly computed.

The proposal in the original protocol for the need for testing the certificate at this point is because the first round of proof is the only point at which the authority can cheat, since zero knowledge is revealed in the second round of proof. This is one of the known ways for an authority to cheat and is eliminated by this verification step – the user can verify the certificate without revealing their identity and if the certificate is not correctly formed, they do not trust the authority.

Authentication

A user who wants to authenticate with the authority to prove that they are a group member interacts with the authority as follows:

Chooses $b \in_R \{1..p-1\}$, $c \in_R Z^*_{p-1}$ and calculates:

$$\beta_1 = \alpha_1^{cd} \bmod p$$
$$\beta_2 = \alpha_2 v^b (\mathrm{mod}\ p-1)$$
$$\beta_3 = y^c (\mathrm{mod}\ p)$$

Sends β_1, β_2, β_3 to the authority, who then calculates:

$$\gamma_1 = \beta_2^{1/w} (\mathrm{mod}\ p-1)$$
$$\gamma_2 = \beta_1^{1/\gamma_1} (\mathrm{mod}\ p)$$
$$\gamma_3 = \left(\frac{\gamma_2}{\beta_3}\right) (\mathrm{mod}\ p-1)$$

[2] Note that the public key referred to here is in effect a pseudonym based on the real primary key, so that there is no direct association between the identity of the group member and the public key presented.

and sends H(γ_3) to the user[3]. The user verifies that $H(\gamma_3) \overset{?}{=} H(y^c \bmod p)$ and then enters a multi-round challenge/response protocol (Figure ??) to demonstrate their knowledge of their private key x. Once the user has confirmed that values of H(γ_3) agree, they are confident that the authority is genuine; once the user has completed t rounds of the zero-knowledge proof of discrete logarithm, the authority is confident that the user is authentic, with a less than 1 in 2^t chance of the user cheating. Authentication thus requires t + 4 modular exponentiations to complete.

Table 1 defines who knows what, and when they know it. Note from the table that the authority does not get any information at authentication time that can identify the user, nor do they get any information that was passed at registration time. This is the key to anonymity of the protocol.

	Authority	**User**
Setup	w, can vary; z, must not change, for all users in the same group.	x, private key
Registration	y, α_1, α_2	α_1, α_2
Authentication	β_1, β_2, β_3; γ_1, γ_2, γ_3	β_1, β_2, β_3, H(γ_3)

Table 1. Who knows what and when they know it in Homage.

Attacks on Homage

Three attacks on Homage were proposed by Jaulmes & Poupard (2001). As there was some confusion with the presentation and preparation of final drafts of the original paper (Camp, 2001) and some indication of contrary findings from the original author (Handley, personal communication 2003), we decided to investigate the attacks through implementation of the various schemes that were suggested. We then considered how to strengthen the protocol as necessary in order to deflect these attacks. Note that there is no suggestion that the basic assumption underlying the protocol is invalid *i.e.* that being able to compute discrete logarithms is hard.

In this section, we first discuss the proposed attacks and illustrate with examples that they are valid. Then we show how to deflect the attacks, either based on close reading of the constraints on the protocol, or by detection by the user and/or authority and, where necessary through strengthening of the protocol.

Attack 1 – Choice of Modulus p to allow subgroup identification

This attack works by placing members in subgroups at registration time and choosing the parameter a for their certificates based on the subgroup in which the user has been placed. When a user tries to authenticate, the authority calculates the multiplicative order of β_1 in modulus p which reveals the subgroup to which the user belongs.

[3] Forging a proof of membership is possible if we know z^th roots modulo p. By sending H(γ_3) instead of γ_3, we avoid this possibility.

More specifically, if there is a modulus p such that $\frac{p-1}{2}$ has many factors (*i.e.* $p = 2 * \prod_{i=1}^{\eta} q_i$), then the authority can have 2^{η} subgroups. The authority calculates $a = a' * \prod_{i=1}^{\eta} q_i^{v_i}$, where v is a binary vector of form (v_1, …, v_η). The multiplicative order of β_1 in modulus p is the product of the q_i's not used in a.

Consider the following arrangement (Table 2) that demonstrates the operation of this attack.

	User	**Authority**

Setup	x = 19	w = 13, z = 3, u = 9, g=23
		p = 521 = 1 + 2 * (2 * 2 * 5 * 13)

Registration		Choose q_4 = 13, place user in subgroup 2 (0, 0, 0, 1) α_1 = 377, α_2 = 39
Authentication	Choose b = 15, c = 7, β_1 = 388	Calculate multiplicative order of β_1 in modulus p which is 20 = (2 * 2 * 5); hence prime used in a must be 13 and the user must be in subgroup 2.

Table 2. Setup parameters to demonstrate subgroup attack.

Attack 2 – Choice of Modulus p based on characteristic order

This is the most serious of the proposed attacks, as it allows a cheating authority to recover the identity of the user at authentication time with time complexity linear in the number of users.

For this attack to be valid, the value $\frac{q-1}{2}$ must have more than one prime factor, for example: $q = 2r_1r_2 + 1$, where r_1 and r_2 are both primes. Now, the authority chooses the parameter u of multiplicative order r_1 in modulus p – 1. Consequently (with high probability), the parameter $v = u^w \pmod{p-1}$ is also of order r_1. The authority maintains a list of all α_2 that have been issued in association with the identities (public keys) of the user to whom they belong. When a user authenticates, the authority calculates $\zeta = \frac{\beta_2}{\alpha_2} \pmod{p-1}$ for all α_2 that have been issued. With high probability (Jaulmes & Poupard refer to "overwhelming probability", but we have found some ambiguity in our simulations), the authenticating user will have ζ of multiplicative order r_1 in modulus p – 1, while other users will have ζ of order r_1r_2 or $2r_1r_2$.

Consider the following situation where r_1 = 19, r_2 = 29 and so q = 1003 and p = 2207. The parameter u is of multiplicative order r_1 in p – 1. Other parameters are w = 10, g = 23 and z = 15 and the key parameters for users are as shown in Table 3.

Name	**x**	**a**	α_1	α_2
Peggy	19	13	1544	1553
Anna	29	17	633	479
Alice	31	21	637	1109
Sue	17	15	1110	813
Marie	37	9	1800	655
Linda	41	25	1849	1063

Table 3. User private key and certificates to demonstrate attack 2.

When these users come to authenticate, they need to select parameters b and c at random. In the following table (Table 4), we list these parameters and tabulate ζ and its multiplicative order for each user. Note from the diagonal column that the characteristic order r_1 shows up and identifies the user.

Name	**(b,c)**	**Peggy**	**Anna**	**Alice**	**Sue**	**Marie**	**Linda**
Peggy	(6, 11)	**(501, 19)**	(2025, 551)	(73, 551)	(1885, 551)	(1363, 551)	(1809, 551)
Anna	(14, 27)	(753, 551)	**(613, 19)**	(1325, 551)	(429, 551)	(2975, 551)	(77, 551)
Alice	(18, 7)	(1715, 551)	(195, 551)	**(677, 19)**	(1979, 551)	(311, 551)	(1939, 551)
Sue	(12, 33)	(1929, 551)	(1377, 551)	(1285, 551)	**(1723, 19)**	(1721, 551)	(83, 551)
Marie	(26, 49)	(1831, 551)	(1667, 551)	(2009, 551)	(141, 551)	**(1731, 19)**	(195, 551)
Linda	(14, 35)	(2127, 29)	(1213, 551)	(693, 551)	(579, 551)	(1335, 551)	**(613, 19)**

Table 4. Characteristic order of ζ on authentication for authority selecting modulus based on known prime factors.

Attack 3 – Choice of different secret key z to identify subgroups

The third form of attack proposed concerns the choice of different values of secret key z by the authority in order to identify sub-groups at authentication. It is interesting to note that Handley also considers this attack and dismisses it as infeasible. We suspect that this difference of opinion is based on the different expression of constraints on the parameters of Homage between drafts of the original paper. Regardless, we consider the attack, implement a simulation and demonstrate by example that it is possible.

Consider the following parameters for the system. Our test user has a private key x=53. When the user registers, the authority chooses a=13 and z=17, issuing certificates α_1= 827 and α_2 = 685. Registration and authentication proceed as in Table 5.

	User	**Authority**
Setup	x = 53	p=2207, g=19, w=5, u=77
Registration		Choose a=13, place user in subgroup corresponding to z=17. α_1= 827 and α_2 = 685
Authentication	b=5, c=13; c is relatively prime to p – 1 β_1 = 1788, β_2 = 965, β_3 = 1820	Calculate γ_3 for each subgroup; send $H(\gamma_3)$ to authenticating user.

Table 5. Registration and authentication for attack based on selection of z for different subgroups.

Deflection of Attacks

In this section, we describe how each of the attacks can be detected, where possible and deflected by the user (if possible).

Attack 1 – based on the assumption that an authority can detect subgroups by choosing a prime modulus that has many factors. Since the authority must publish p and that it is constrained in the protocol that $\frac{p-1}{2}$ must be prime, this form of cheating is easy to detect by checking that α_2 is co-prime to p. From the example above, α_2 and p share a common factor 13.

Attack 2 – based on the assumption that p can be selected so that it is based on a value q that is itself based on two prime factors. The problem with this attack is that both u and v are publicly known and their respective multiplicative orders can be verified. If u is of order $\frac{q-1}{2}$, where $q = \frac{p-1}{2}$, then it is possible for the user to detect the attack and not to trust the authority. Noting the restriction on the selection of parameter u, which should be co-prime to p – 1, this attack is deflected.

Attack 3 – although we are unsure exactly how the authority selects a different z value for different groups of users based on the information sent at authentication time, we have found no way for the user to detect this attack.

Since we found that all of the attacks were valid, but that only one of them could not be detected given the constraints of the original protocol, we found that we needed to make modifications to the original Homage protocol. We accept that the proposed protocol modifications due to Jaulmes & Poupard (see appendix)have merit, but we do not accept that their form of solution is necessary to address the only fault in the protocol that we agree to be present – the ability for the authority potentially to cheat using different z values for different users. In the next section, we describe what we did to modify the protocol and the additional assurance of security that it provides.

Modified Homage Protocol

The modification that we have made to the protocol addresses attack 3 above and is achieved by modifying the process for issuing certificates and by including a zero-knowledge proof prior to the first round of authentication.

Homage Parameters

Public information

- *p* is a public prime integer such that $q = \frac{p-1}{2}$ is also prime
- g is a public generator of Z^*_{p-1} and should be able to generate a large fraction, if not all of Z^*_{p-1}
- $u \in Z^*_{p-1}$, a public constant co-prime to $p-1$
- $v = u^w \pmod{p-1}$ is a public key of the authority,

- $y = g^x \bmod p$ is the public key of the user; to completely disassociate from the user, a pseudonym can be adopted and used.
- $H(x)$ is a publicly-known secure hash function,
- α_1 and α_2 form the certificate issued by the group authority; they require knowledge of private key x to use; α_1 is issued in the form $\alpha_1 = E_1 I_1 (\bmod p)$ along with $g_1 = y^a \bmod p$, where $a \in_R Z^*_{p-1}$, $E_1 = g^a \bmod p$ and $I_1 = g_1^z (\bmod p)$
- $g_2 = g_1^{r_1} \bmod p$, where $r_1 \in_R \{1..q\}$ is calculated by the user in proving knowledge of z at authentication,

Information private to the authority

- z and w are two secret keys held by the authority such that $z \in Z^*_{p-1}$ and $w \in [1, p-2]$. (Jaulmes & Poupard (2001) make this claim on the basis of their personal communication with Handley and this confirms our own exchanges with the original author.)

Information private to the group member

- $x \in Z_{p-1}$ is the private key of the group member.

These parameters are summarised in Table 6 to indicate who knows what and at what point in the protocol they know it (contrast this with the same information shown in Table 1 for the original protocol).

	Authority	**User**
Setup	w, can vary; z, must not change.	x, private key
Registration	y, α_1, α_2	Knows that authority knows z and that it is the same z that I_1 is based on.
		g_1, I_1, E_1, r_2, α_1, α_2
Authentication	β_1, β_2, β_3; γ_1, γ_2, γ_3, g_2, r_3 Key to anonymity of Homage is that nothing that can reveal the identity of the user or anything known to the authority at registration time is made known at authentication time.	x_1, s, $H(\gamma_3)$

Table 6. Who knows what and when they know it in modified Homage protocol.

Registration

During registration, the group member presents their public key (or pseudonym) to the group authority. The variation here on the original is to divide the first half of the certificate α_1 into g_1, E_1 and I_1. This is necessary since, in the authentication phase, we force the authority to prove knowledge of the z that was used to create the certificate[4]. Since z is only used as an exponent, the security of the protocol overall still depends on the difficulty of finding discrete logarithms. Table 7 shows the modified flow of information between the user and the group authority.

Group Authority		**Group Member**
$g_1 \equiv y^a (\bmod p)$	$\rightarrow g_1, E_1, I_1, r_2$	$\alpha_1 = E_1 * I_1 \bmod p$
$E_1 \equiv g^a (\bmod p)$		
$I_1 \equiv g_1^z (\bmod p)$		
	$\leftarrow r_3$	$r_3 = \in_R \{0, B\}$
	$\rightarrow s$	
$\alpha_2 \equiv a^w (\bmod p-1)$	$\rightarrow \alpha_2$	Member accepts proof of z, now has certificate in both halves α_1 and α_2

Table 7. Registration with modified protocol.

Registration requires only 4 modular exponentiations.

Verification

Verification is necessary to ensure that the certificate has been correctly computed. This is because the only possibility for an authority to cheat is in the computation of the certificate and at the first round of proof. The protocol for verification of the certificate is the same as that for authentication below, including proof of knowledge of z as the first stage and certificate authentication, but not including the challenges by which the user proves knowledge of γ_3.

Authentication

Proof of Knowledge of z

Since the authority has issued the group member with a certificate based on secret keys w and z, we require the authority to prove their knowledge of the same secret key as the first stage of authentication. We do this using a zero-knowledge proof with the following sub-protocol between authority and group member (Table 8)..

[4] This method for issuing certificates is due entirely to Gohil and is the mechanism that enables the zero-knowledge proof of z to take place as the first stage in authentication.

Authority		**Member**
	$g_2 \leftarrow$	Chooses $r_1 \in_R \{1..q\}$ and calculates: $g_2 \equiv g_1^{r_1} (\bmod p)$ $I_2 \equiv I_1^{r_1} (\bmod p) \equiv \left(g_1^{r_1}\right)^z (\bmod p)$
Choose $r_2 \in_R \{1..q\}$ $x_1 = g_2^{r_2} \bmod p$	$\rightarrow x_1$	
	$r_3 \leftarrow$	Choose $r_3 \in_R \{1..q\}$
$s = r_2 + (r_3 * z) \bmod q$	$\rightarrow s$	$g_2^s (\bmod p) \overset{?}{=} x_1 * I_2^{r_3} (\bmod p)$

Table 8. Proof of equality of two discrete logarithms (originally due to Chaum & Pedersen (1992), with the form here taken from sub-protocol A in Jaulmes & Poupard (2001).

There is no way for the authority to be able to cheat by using a different z to identify sub-groups of users since they do not get any clue as to which z to use at the initiation of the protocol. The authority can work out *a* from g_2 only if they can compute discrete logarithms.

Authentication of the certificate

Authentication of the certificate proceeds as described in the original protocol (Table 9):

Group Authority		**Group Member**
$v = u^w (\bmod p-1)$ is publicly known.		Chooses $b \in_R \{1..p-1\}$, $c \in_R Z_{p-1}^*$ and calculates: $\beta_1 = \alpha_1^{cd} \bmod p$ $\beta_2 = \alpha_2 v^b (\bmod p-1)$ $\beta_3 = y^c (\bmod p)$
	$\beta_1, \beta_2, \beta_3 \leftarrow$	
$\gamma_1 = \beta_2^{1/w} (\bmod p-1)$ $\gamma_2 = \beta_1^{1/\gamma_1} (\bmod p)$ $\gamma_3 = \left(\frac{\gamma_2}{\beta_3}\right) (\bmod p-1)$	$\rightarrow H(\gamma_3)$	Verify $H(\gamma_3) \overset{?}{=} H(y^c \bmod p)$

Table 9. Authentication of the user with the modified protocol.

The authentication protocol above is also used to verify the certificate, in an anonymous fashion.

Group member then proves knowledge of private key in γ_3 since $\gamma_3 \equiv \beta_3^x$. Using zero-knowledge proof of equality of discrete logarithm, the user can only be authenticated if they can prove knowledge of x. Sending H(γ_3) is necessary to avoid leaking zth roots modulo p, as noted by Jaulmes & Poupard and corrected by Handley in the final draft of the original paper (personal communication, 2003).

Authentication requires only t + 4 modular exponentiations, where t is the security parameter used in the challenge/response protocol (Table 14). The final round of the protocol below is used during authentication only and consists of t rounds of challenges to ensure that the group member has less than 1 in 2^t chance of cheating.

Security Review

We now review the security of the protocol as far as is known, to demonstrate that our modified protocol is no longer vulnerable to the proposed attacks and to indicate the security of the zero-knowledge proofs that are used.

Modified Protocol And Proposed Attacks

As the protocol explicitly constrains prime modulus p to be of the form $p = 2q + 1$ with $q = 2r + 1$ also prime, $\frac{p-1}{2}$ has only one factor and therefore is not vulnerable to attack 1.

For attack 2, consider the users registration parameters as in (Table 10) and system parameters w = 10, z = 15, g=23 as before, but this time we select prime modulus based on r = 41, q = 83 and p = 167.

Name	x	a	α_1	α_2
Peggy	19	13	141	77
Anna	29	17	22	161
Alice	31	21	127	9

Table 10. User private keys and certificates to demonstrate neutralisation of attack 2.

For these users at authentication time, we tabulate below $\zeta = \frac{\beta_2}{\alpha_2} (\bmod\, p-1)$ and its multiplicative order (for a subset of these users). Note that no characteristic multiplicative order is now seen.

Name	(b, c)	Peggy	Anna	Alice
Peggy	(6, 11)	(21, 41)	(75, 41)	(75, 41)
Anna	(14, 27)	(99, 41)	(69, 41)	(17, 41)
Alice	(18, 7)	(11, 41)	(63, 41)	(131, 41)

Table 11. Multiplicative order of ζ for non-cheating authority.

For attack 3, the authority must guess the correct value of z for the subgroups into which they have divided users. The chance of selecting the correct group by chance is $\frac{1}{n}$ if there are *n* subgroups. The only piece of information on which the authority can base this guess at authentication time in our modified protocol is $g_2 \equiv g_1^{r_1} \pmod{p}$ as opposed to β_1, β_2, β_3 in the original protocol.

1. Protocol – the modified protocol is now immune to the attacks proposed by Jaulmes & Poupard (2001); attacks 1 and 2 are easily detected by the user so the authority cannot get away with cheating and attack 3 is deflected by our inclusion of the zero-knowledge proof as the first step in authentication and the modified method of issuing certificates. Sub-protocol A is a standard zero-knowledge proof and is sound and complete (Feige et al, 1988; Goldwasser et al, 1989; Chaum & Pedersen, 1992). Sub-protocol B is also a standard proof and is zero-knowledge.

2. Cheating by the authority – the ability of the authority to cheat by issuing certificates in an incorrect form is removed by the maintenance of the verification phase as per the original protocol; selecting unsafe prime modulus is easily detected; selecting different z values for sub-groups is no longer possible due to zero-knowledge proof of z at first stage of authentication.

3. Cheating by the user – a user can only forge certificates by computing a discrete logarithm, which we assume is hard, or by computing g^z, which is theoretically possible, but equally hard.

Biometric Extensions

We describe three possible extensions to Homage to incorporate biometric extensions.

Option 1

For this extension to the Homage protocol, to incorporate biometric data, we make the following definitions:

- *B* is a bit-string that represents a biometric template,
- $B' = encrypt(B, y)$, where y is the user's public key (or pseudonym),
- $T = encrypt(B, x)$, where x is the user's private key,
- $T' = encrypt(T, \beta_3^x)$, calculated at authentication,
- $encrypt(A, b)$ is some secure encryption function that encrypts *A* with key *b* so that $encrypt(encrypt(B, x), y) \neq encrypt(encrypt(B, y), x)$

Creating the biometric template *B* can be done as in many biometric devices, to use raw biometric data and use a one-way transformation to generate a template. Biometric data is captured at enrolment to generate a template and later compared during verification where biometric data is captured from the user, a new template is generated and compared with the original that was enrolled.

Generation of biometric data - in our scheme, when the template *B* is generated, it is immediately encrypted with the user's public key to generate *B'* and with the private key to generate *T*. This has an added benefit of securing the biometric data since that data can never be revoked. If biometric data exists in raw form, or even as an unprotected template, there are significant security and privacy risks to the user.

Issuing of keys –users are issued with their public and private keys by the key authority at the same time that biometric data is captured. We do not need to identify the user; all that we require is that the biometric data is captured in the presence of the key authority so that we know it came from the same user to which the keys are issued. The user then submits *T* and *B'* to the key authority who verifies that the biometric has been encrypted with keys x and y respectively.

Communication between key authority and group authority – the key authority is in the role of a trusted 3rd party and should be separate from the group authority. The key authority sends a list of {*y, B'*} to the group authority and a separate list of *T*. This should be done in such a way as to ensure that the group authority cannot associate any *T* with the corresponding *B'* or *y*. Randomization of the list order will suffice.

Registration with the group authority – the user sends public key *y* and *B'* to confirm that they have registered with the key authority and generated a valid, encrypted template. Registration then proceeds in the normal way as described above to generate certificates.

Authentication with the group authority – the user authenticates with the authority as above via proof of z, calculation of parameters β_1, β_2, β_3 and verification of $H(\gamma_3)$. They send *T* to the authority and *y, B, T,* $c \in_R Z_{p-1}$ to a tamper-proof biometric authentication device.

The authority checks that *T* is in the list received from the key authority, in addition to performing the normal authentication checks. If not, then either the user has not authenticated with the key authority, or they have used a private key belonging to some other user to encrypt their biometric data. If the authority is convinced that the user is valid, they then send *T*, β_3 to the authentication device.

The authentication device first compares the values of *T* sent by the authority and the user to ensure that they are the same. Then it checks that $\beta_3 = g^c \bmod p$ to ensure that the same parameter *c* has been sent to the authentication device as to the authority during authentication of the certificate. Then it does a biometric authentication with the user by capturing data, generating a template B1 and ensuring that $B_1 \equiv B$.

Authentication device		**Authority**
$\beta_3^x = y^c \bmod p$ $T' = encrypt(T, \beta_3^x)$	**T'→**	$T_1' = encrypt(T, \gamma_3)$ $T_1' \overset{?}{=} T'$

Table 12. Biometric authentication of a user.

If the authority verifies that $T_1' \overset{?}{=} T'$ (Table 12), then they are assured that:

- The user has previously registered with the key authority,
- The private key x corresponds to the authenticating user,
- The same user that is authenticating now was the same one that enrolled and encrypted the biometric data B with the key authority.

The disadvantage with this approach is that information regarding the user is now made available to the authority at authentication time. If there is any collusion with the key authority, the group authority can assemble a dictionary of associations between public keys and encrypted biometric data and hence can recover the identity of the user.

Option 2

We now discuss an option for biometric authentication and non-transferability of the private key based on the mechanism of wallet databases plus observers. This mechanism uses the invention of the electronic wallet that consists of a computer controlled by the user (a smart-card in this case) and a tamper-proof mechanism embedded within it, known as the observer. The two parts are arranged so that the observer can only communicate with the card and not the outside world (Figure 1).

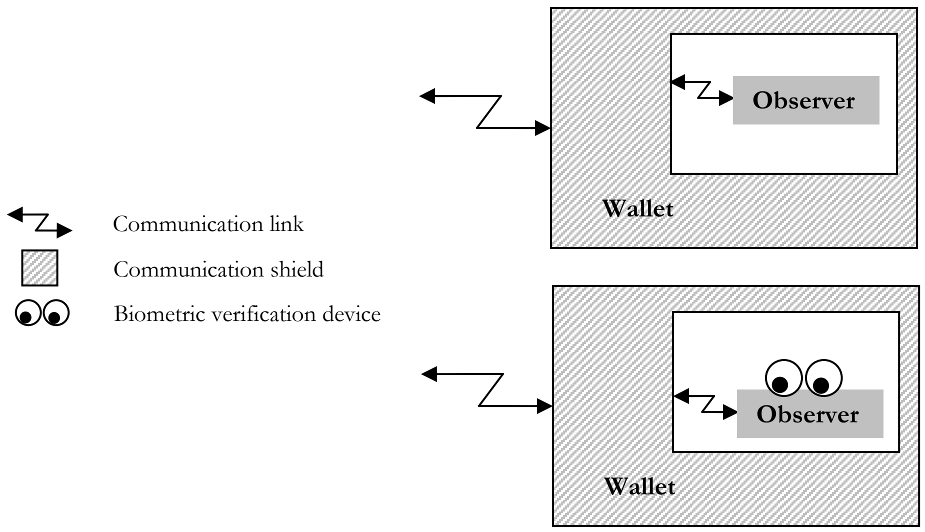

Figure 1. Wallet databases plus observers (i) basic mechanism due to Chaum & Pedersen (1992) and (ii) mechanism with embedded biometric verification due to Bleumer (1998).

Bleumer's extension to this mechanism is to include a biometric verification facility within the tamper-proof observer. This device is used to verify identity of the controlling user and operates as follows:

- On registration, biometric credentials are obtained from the user and enrolled on the card along with the user's private key.
- On authentication, the biometric device compares a biometric sample taken from the authenticating user and compares it with the enrolled user. If there is no match, the device will not engage in further authentication.

This method has several advantages. Firstly, it provides a mechanism for biometric authentication. Secondly, it protects the identity of the user since no information needs to be sent to the authority at authentication time, other than that the user passed biometric verification. The procedure for obtaining a private key and pseudonym is as outlined in Chaum & Pedersen (1992, p11-13).

Appendix A – Zero-Knowledge Proofs

The following zero-knowledge proofs are used in the Homage protocol.

For the following proof of equality of discrete logarithms (Table 13):

$p = 2q + 1$, a safe prime integer;
B some integer;
$g_1, g_2 \in Z_p^*$ of order q;
$s \in Z_q$ is secret data.

Public data: $I_1 = g_1^s \bmod p$ and $I_2 = g_2^s \bmod p$

Verifier		**Prover**
	xx_1, x_2 $\leftarrow$	Choose $r \in Z_q$ and calculate
		$x_1 = g_1^r \bmod p$, and
		$x_2 = g_2^r \bmod p$
Choose $c \in_R \{0, B\}$	$\rightarrow$ c	$y = r + c * s (\bmod q)$
	y $\leftarrow$	
Check that		
$g_1^y \overset{?}{=} x_1 * I_1^r \bmod p$ and		
$g_2^y \overset{?}{=} x_2 * I_2^r \bmod p$		

Table 13. Proof of equality of discrete logarithms.

Authority		**Group Member**
	$\gamma_4 \leftarrow$	$t \in_R Z_q^*$
		$\gamma_4 = \beta_3^t \bmod p$
$c \in_R \{0,1\}$	$\rightarrow c$	
$c = 0 : \gamma_4 \overset{?}{=} \beta_3^t \bmod p$	$c = 0 : t \leftarrow$	
$c = 1 : \gamma_3 \overset{?}{=} \gamma_4^{\left(\frac{x}{t}\right)} \bmod p$	$c = 1 : \left(\frac{x}{t}\right) \bmod p \leftarrow$	

Table 14. Challenge/response protocol for group member to prove knowledge of discrete logarithm γ_3 in basis β_3.

Appendix B – Alternative Modified Protocol

The alternative protocol proposed by Jaulmes & Poupard removes the verification phase from the original protocol and provides a proof with each certificate that it has been correctly computed at registration. A commitment scheme is used to commit the private key z of the authority. This is committed with a public key $Y = g^z h^j (\bmod p)$, where j is random, h is of order q and the discrete logarithm of h in basis g is unknown to the authority. Opening a commitment involves revealing the commited value and the random value used in the commitment.

When certificates are computed, the authority chooses a random value t and calculates and sends to the user the following values:

$$C_1 = commit(t)$$

$$C_2 = commit(t * a (\bmod q))$$

$$C_3 = commit(a_1^t (\bmod p))$$

$$C_4 = (ta)^w (\bmod q)$$

A challenge-based system is used and based on the answers to a challenge, either C_1 or C_2 is opened and the user and authority prove knowledge of their respective private keys.

Pedersen's protocol (Pedersen, 1991) provides two options for selecting parameters g and h – either by using a trusted centre (third party), or via a coin-flipping protocol. If a third party is to be used, then they have to be involved when the commitment of t is made or opened. This seems unreasonable, given that the goals of the Homage protocol are to be simple and resource-efficient.

Essentially what is being done here – group member and authority proving knowledge of their private keys – is the same as in our proposed modifications. The zero-knowledge scheme of proving the private key z of the authority at authentication time seems to us to be a simpler method of securing the protocol.

References

1. Bleumer, G. *Biometric yet Privacy-Protecting Person Authentication*. in *Information Hiding 1998*. 1998: Springer-Verlag.

2. Boneh, D. *The decision Diffie-Hellman problem*. in *Third Algorithmic Number Theory Symposium*. 1998: Springer-Verlag.

3. Chaum, D. and T.P. Pedersen. *Wallet Databases with Observers*. in *Advances in Cryptology: Proceedings of CRYPTO '92*. 1992: Springer-Verlag.

4. Halevi, S. and S. Micali, *Practical and Provably-Secure Commitment Schemes from Collision-Free Hashing.*

5. Handley, B. *Resource-Efficient Anonymous Group authentication*. in *Financial Cryptography, 4th International Conference FC2000*. 2000. Anguilla, British West Indies: Springer-Verlag.

6. Impagliazzo, R. and S.M. More. *Anonymous Credentials with Biometrically-Enforced Non-Transferability*. in *Workshop on Privacy in the Electronic Society (WPES '03)*. 2003. Washington DC, USA: ACM.

7. Information Technology Library, N., *Security Requirements for Cryptographic Modules. Federal Information Standards Publication FIPS PUB 140-2*. 2001, National Institute of Standards and Technology: Gaithersburg, MD.

8. Jaulmes, E. and G. Poupard. *On The Security of Homage Group Authentication Protocol*. in *Financial Cryptography, 5th International Conference FC2001*. 2001. Grand Cayman, British West Indies: Springer-Verlag.

9. Pedersen, T.P. *Non-Interactive and Information-Theoretic secure Verifiable Secret Sharing*. in *Advances in Cryptology: Proceedings of CRYPTO '91*. 1991: Springer-Verlag.

Acknowledgements

We are grateful to the assistance of our colleagues in the preparation of this report, particularly Mike Atkinson and Michael Albert who listened patiently as we explained our assumptions and difficulties, and to Chris Handley for locating drafts of Ben Handley's original paper and putting us in touch by e-mail. I am also indebted to my partner Donna for deciphering my handwriting and preparing many of the equations in the drafts of this paper. The project was supported by a Department of Computer Science summer bursary grant.