Improved Privacy in Wallets with Observers

(Extended Abstract)

R. J. F. Cramer[1] and T. P. Pedersen[2]

[1] CWI, The Netherlands
[2] Aarhus University, Denmark***

Abstract. Wallets with observers were suggested by David Chaum and have previously been described in [Ch92] and [CP92]. These papers argue that a particular combination of a tamper-resistant-unit and a small computer controlled by the user is very suitable as a personal device in consumer transaction systems. Using such devices, protocols are constructed that, simultaneously, achieve high levels of security for organizations and anonymity for individual users. The protocols from [CP92] offer anonymity to users, under the assumption that the information stored by observers is never revealed to the outside world.
This paper extends [CP92] by defining additional requirements for the protocols which make it impossible to trace the behaviour of individuals in the system if one is also allowed to analyze a posteriori the information observers can collect. We propose two protocols satisfying our requirements, thus achieving a higher degree of privacy for individuals. This extra level of privacy is obtained at essentially no cost as the new protocols have the same complexity as those previously proposed.

1 Introduction

In [Cha83], the notion of blind signatures was proposed and used to construct an on-line electronic payment scheme for anonymous payments. Later, Even and Goldreich (see [EG84]) suggested an off-line electronic payment scheme in which the security relied on a tamper-resistant unit. However, such units inherently cannot offer proper protection of the individual's privacy as the unit in principle can send a lot of personal information to the counterpart during a transaction. An off-line payment system not depending on tamper-resistance and offering privacy was presented in [CFN90]. In this model (off-line and no tamper-resistant units) the same coin cannot be prevented from being spent several times, but [CFN90] solved this problem by guaranteeing that any person who spends a coin more than once would be identified.

Thus, as noted by Chaum in [Ch92], in off-line transactions a solution based on tamper-resistance can give good security but no satisfactory solution to the privacy aspect, whereas cryptographic solutions without tamper-resistant units although offering good privacy cannot prevent "double-spending". Therefore it

*** Research partly done while visiting CWI was suggested in [Ch92] to obtain the best from these two worlds by combining the two approaches. This is done in an electronic wallet consisting of a small computer, trusted by the individual, and a tamper-resistant unit, called an *observer*. These observers are issued by a special issuing authority (IA) and are trusted by this authority. As the observer may contain (sensitive) information about the user, it is important for privacy reasons that the observer cannot communicate with the outside world (neither during the transactions nor when the observer is not used). During transactions all communication between an observer and the outside world (e.g. the organizations individuals do business with) therefore passes through the user's computer. As part of the protocols, this computer must ensure that the data transmitted to and from the observer contain no (Shannon) information (except for agreed upon messages). This is called "prevention of inflow and outflow". In [CP92], protocols are presented which satisfy this requirement.

In order to be able to recognize observers, these are equipped with a digital signature set-up. This so-called *native signature scheme* is used to prove that the user has a genuine observer. However, the native scheme cannot be used directly, as it would identify the observer (and thereby the user) in each transaction. Instead the native scheme is used to obtain a *validator* at a Validator Issuing Center (VIC). A validator consists of a secret key and a public key, together with a signature by VIC on the public key. One of the central ideas behind validators is that VIC doesn't know which validator it issues (VIC makes a blind signature), so that the wallet can later use it to sign messages that cannot be traced back to the user. Furthermore, the user (more precisely the user's computer) and the observer must cooperate in order to sign messages with respect to the public key of the validator. Such a signature therefore signals to the receiver that the observer (which the receiver is assumed to trust) has acknowledged the validity of the message.

Each validator can be used to sign multiple messages. Any other party can be sure that two messages signed with the same validator originated from the same observer. By using different validators at different times, the user can prevent linking.

As the protocols of getting and using a validator are the central components of most applications of wallets with observers, we concentrate on these two protocols in this paper. Such a pair of protocols will be called a *validator scheme*.

In [CP92] a validator scheme is proposed where unlinkability of a users transactions is based on the assumption that the observer's contents are never revealed. However, assuming an environment of mutual distrust and cheating, observers could be designed in such a way that their contents can be accessed by IA (using a secret trapdoor) in case it has complete physical control over the observer (either by means of stealing, or because of maintenance procedures). In Section 3 we present protocols where the executions of the signing protocol (unconditionally) cannot be linked to the user even if the contents of the observer is known to all organizations (including IA and VIC). First, however, Section 2 describes the set-up in more details and gives the necessary definitions.

The results in this paper are also presented in [BCCFP92]. Furthermore, to avoid misunderstandings it must be mentioned that the protocols were already presented by David Chaum as part of the presentation of [CP92] at Crypto'92. However, the protocols are not included in that paper. The contribution of this paper is to introduce the notion of *shared information* and to present the criteria according to which the protocols were constructed.

2 Security of Protocols Involving Observers

This section discuss the basic principles of wallets with observers, and the security of the basic protocols is defined.

2.1 Tracing an Observer if its Contents Are Known.

When considering the privacy of the user we must assume that the observer stores whatever information can be collected from the transactions. In the validator issuing protocol of [CP92], observers know exactly which particular validator is being issued. When the validator is later used in a validator-based signature scheme to sign messages, the observer is able to link the execution of the validator-based signature scheme to that of the issuing protocol. Furthermore, the observer's native signatures, used in the issuing protocol, uniquely link VIC's views of the executions of the issuing protocol to the observer's views of that protocol. As the receiver of a validator-based signature knows the public key of the validator (and VIC's signature on it), the receiver's view of the execution of the validator-based signature scheme, can be linked to the observer's view of the same protocol. Consequently, if the contents of an observer are ever known to the outside world, its behaviour can be completely recovered (see Figure 1). Linking observers with individuals is just a practical matter. So in a scenario in which the contents of (some of) the observers may become available to the outside world, the anonymity of individuals in the system faces serious threats.

Moreover, this attack, which was based on the fact that the observer knows which particular validator is involved in each transaction, can be generalized:

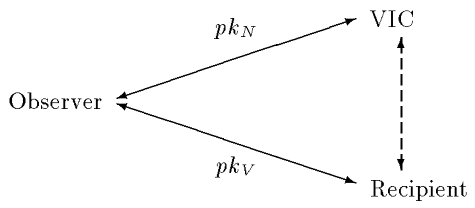

Fig. 1. Linking a validator based signature to the issuing of the validator using the public key of the native scheme, pk_N, and the public key of the validator, pk_V.

any transaction data that arises from the interaction between organizations and the wallet and is known by both observer and recipient can be used to establish the described links. Hence, when designing the protocols such data (called *shared information*) must be avoided.

2.2 Definitions

In view of the above attack we now describe additional criteria for the protocols, which, roughly speaking, require that the public key of validators be unconditionally hidden from observers and that the validator-based signature scheme prevents shared information (except for the message, m, to be signed). As the observer and the receiver in general must know the message, it should be chosen from a sparse set such that each message is likely to be signed by many different wallets. This prevents that m can be used to link transactions.[4]

Recall that a validator scheme consists of a pair of protocols. One, the validator issuing protocol, results in the user and her observer getting a validator, and the other, the validator signing protocol, can be used to make signatures with respect to the public key of the validator. We trust that this informal description is sufficient here.

First note that the validator issuing as well as the signing protocol are three-party protocols involving observer and user plus either VIC or the receiver of the signature. In this extended abstract we leave out a precise definition of our model. Briefly, each participant is modeled by an interactive probabilistic polynomial time Turing machine as in [GMR89]. In some places dishonest participants may have unlimited computing power, but this should be clear from the context. The user will be called Alice and her observer will be denoted by OA.

In order to capture what a participant learns from such a protocol we use the notion of views introduced in [GMR89].

Definition 1. For any execution of a protocol and for any participating party, X, which has input i_x, the view of X, $\text{VIEW}_X(i_x)$, is the set of all the messages that X sees plus all the random choices that X makes.

A validator scheme will be said to protect the privacy of the user, if whenever the user follows her protocol it is impossible for an unlimited powerful entity to find any information regarding which signature corresponds to which execution of the validator issuing protocol, even with all the information of VIC, the recipient of the signature, and the observer. The following definition expresses this informally in terms of views.

Definition 2. Let a validator scheme be given, and let R denote the recipient of the signature. Let a message, m, be given. The scheme prevents *shared information* if whenever Alice follows the issuing and signing protocols (when signing m) then no matter what unlimited powerful VIC, OA and R do in the protocols,

[4] Replay attacks can be prevented by incorporating random bits chosen by the recipient into the signature, see Section 3.5.

the (distribution of the) view of R (when receiving a signature on m) is independent of the combined views of VIC and OA (the views of the observer in both protocols).

This definition basically says that the receiver of the signature may not be able to obtain any information which can be linked to the observer. Namely, if such information exists the receivers view of the signature scheme can be linked to the observer, and the observer can easily link its view of the protocol to the corresponding validator issuing protocol (see also Section 2.1).

2.3 Requirements for the Validator Scheme

In light of the above definitions we now list the properties that the validator issuing and the signature protocol must have in order to be secure as well as protect the privacy of the user. The validator issuing protocol must have the following properties:

1. It must not be feasible for a cheating user to obtain a validator for which she knows the secret key herself (i.e., VIC must be ascertained that a valid observer is taking part).
2. VIC's signature scheme and the application of it in the validator issuing protocol must be secure, such that Alice (even if she breaks the observer) is unable to forge signatures.
3. No (Shannon) information flows between the observer and VIC (if Alice follows the protocol). This is true even if OA and VIC both have unlimited computing power (i.e., no inflow/outflow).
4. Only Alice knows the public key of the validator and VIC's signature on it (both should be unconditionally hidden from VIC and OA, see also requirement 5 to the validator signing scheme below).
5. OA and Alice each get a part of the secret key of the validator from VIC, such that they can only use the secret key by cooperating.

It is in properties 4 and 5 that our issuing scheme differs from that given in [CP92]. These properties ensure that the observer and Alice share the control over the validator, while the observer doesn't know the public key of the validator. The first requirement ensures that Alice is not able to acquire a validator for herself. If she were able to do so, then she could dispense with the observer and run all the other protocols herself without the help of an observer (see also point 3 below).

The signature scheme based on validators must have the following properties (we often refer to the recipient as the verifier):

1. If Alice and OA cooperate they can make a signature on a given message with respect to the public key of their validator.
2. If Alice follows the protocols, then it is not feasible for OA, VIC and the verifier to execute the validator issuing protocol and the signing protocol (several times) in such a way that they are able to learn the secret key of Alice.

3. If OA and VIC both follow the validator issuing protocol, then it is not feasible for Alice to execute this protocol and subsequent signing protocols in such a way that she can sign a message which the observer is not willing to sign.
4. In order to avoid replay, the verifier will be interactive in the protocol.
5. The validator scheme must prevent shared information (Definition 2).

The first four requirements are related to the security of the validator signing protocol, while the last refers to the privacy obtained by the validator scheme.

3 The Protocols

This section presents a validator scheme, which satisfies the requirements outlined above. The protocols work for every group of prime order q. For the sake of concreteness we take a prime order subgroup of $\mathbb{Z}_p^*$, where p is a prime. All protocols are based on the certified discrete logarithm problem. More precisely, it is assumed to be hard to compute discrete logarithms modulo a prime, p, even if the factorization of $p-1$ is given.

3.1 The Basic Signatures

The protocols in this paper use the signature scheme presented in [CP92]. We now briefly describe this scheme. Let two primes p and q be given such that q divides $p-1$, and let $g \in \mathbb{Z}_p^*$ be an element of order q. The group generated by g is denoted by G_q.

The secret key is $x \in \mathbb{Z}_q^*$ and the public key is

$$(p, q, g, h),$$

where $h = g^x$. Let $m \in G_q$ be a message. The signature on m consists of $z = m^x$ plus a proof that

$$\log_g h = \log_m z.$$

This proof bears some resemblance to two parallel executions of instances of Schnorr's scheme [S91] and works as follows:

1. The prover chooses $w \in \mathbb{Z}_q$ at random and computes $(a, b) = (g^w, m^w)$. This pair is sent to the verifier.
2. The verifier chooses a random challenge $c \in \mathbb{Z}_q$ and sends it to the prover.
3. The prover sends back $r = w + cx \bmod q$.
4. The verifier accepts the proof if and only if $g^r = ah^c$ and $m^r = bz^c$.

We shall often refer to this protocol as the *basic proof* (of equality of discrete logarithms).

Now let $\mathcal{H}$ be a one-way hash function mapping arbitrary inputs into the set $\mathbb{Z}_q$ (as in the Fiat-Shamir scheme, see [FS86]). Given this function and the above protocol the signature on m is

$$\sigma(m) = (z, c, r).$$

It is correct if

$$a = g^r h^{-c} \qquad \text{and} \qquad b = m^r z^{-c}$$

satisfy $c = \mathcal{H}(m, z, a, b)$.

The reader is referred to [CP92] for a discussion of the security of this signature scheme. In that paper it is argued that it is hard to forge signatures, and that the signer does not give away any information about his secret key except m^x. This is made more precise in the following assumption:

Assumption 1. *When the signer signs a message, he does not make it computationally easier to compute any function of x, than if he just gives away m^x.*

Assumption 2. *If it is possible to make a signature (z, c, r) on a message m with respect to the public key $h = g^x$ then $z = m^x$.*

The second assumption can be justified as follows. If $z \neq m^x$, then for every pair (a, b) there is at most one $c \in \mathbb{Z}_q$ for which there exists an $r \in \mathbb{Z}_q$ such that the signature is valid. Hence, if $z \neq m^x$ the forger must hope that the image of $\mathcal{H}$ is exactly this c. However, if the output of $\mathcal{H}$ is "hard to control" this seems extremely unlikely.

3.2 Blind Signatures

To get a blind signature on the message $m \in G_q$ one chooses a random $t \in \mathbb{Z}_q^*$ and asks the signer to sign $m_0 = mg^t$. Let $z_0 = {m_0}^x$. The signer then proves that $\log_g h = \log_{m_0} z_0$ in such a way that the messages are blinded:

1. The signer chooses a random $w \in \mathbb{Z}_q$ and computes $(a_0, b_0) = (g^w, m_0^w)$. This pair is sent to the verifier.
2. The verifier chooses $u \in_{\mathcal{R}} \mathbb{Z}_q^*, v \in \mathbb{Z}_q$ at random and computes

   $$a = a_0^u g^v \qquad \text{and} \qquad b = (b_0/a_0^t)^u m^v.$$

   (If both parties follow the protocol, $a = (g^{uw+v})$ and $b = (m^{uw+v})$.) The verifier computes $z = z_0/h^t$, the challenge $c = \mathcal{H}(m, z, a, b)$ and the blinded challenge $c_0 = c/u \bmod q$. The verifier sends c_0 to the signer.
3. The signer sends back $r_0 = w + c_0 x \bmod q$.
4. The verifier accepts if

   $$g^{r_0} = a_0 h^{c_0} \qquad \text{and} \qquad {m_0}^{r_0} = b_0 z_0^{c_0}.$$

   The verifier computes $r = ur_0 + v \bmod q$ and

   $$\sigma = (z, c, r).$$

The following two propositions from [CP92] show that this really is a blind signature:

Proposition 3. *If the verifier accepts in the above protocol, then σ is a correct signature on m.*

Proposition 4. *The signer gets no information about m and σ if the verifier follows the protocol.*

The following assumption makes precise which signatures the verifier can obtain when executing the blind signature protocol:

Assumption 3. *On input m_0 to the above blind signature protocol, the verifier can only obtain a blind signature on a message $m \in G_q$ if he knows $s, t \in \mathbb{Z}_q$ such that $m = m_0^s g^t$.*

This assumption can be justified as follows. Assumption 2 implies that

$$z = m^x \qquad \text{and hence} \qquad \log_m b = \log_g a.$$

Thus the verifier must be able to compute m^x from m_0^x. This is presumably difficult unless m is a product of elements for which the x'th power is known. Furthermore, under Assumption 2 it can be shown that no matter how the verifier computes b it satisfies

$$b = m^{uw+v},$$

where u and v are defined by

$$u = c/c_0 \bmod q \qquad \text{and} \qquad v = r - ur_0 \bmod q.$$

It seems to be hard to compute a pair (m, b) satisfying this before getting r_0 from the signer unless m is of the prescribed form.

3.3 Signatures by OA

In [CP92] it is shown how this signature scheme can be used by the observer such that it cannot send any information to or receive any information from the verifier (i.e., without causing inflow of outflow). This scheme can therefore be used as the native signature scheme.

3.4 Validator Issuing

We now turn to the problem of creating a validator. The public key is going to be a number $h \in G_q$ such that the corresponding secret key, $\log_g h$, is of the form $s + t \bmod q$, where only OA knows t and only Alice knows s. Furthermore, Alice will obtain a (blind) signature from VIC on h.

Before this protocol can start, the native public key and the certificate on it are sent to Alice and VIC. VIC has a private key x and a public key $H = G^x$ where $G \neq g$ and G is a generator of G_q such that Alice does not know $\log_g G$ (hence, (G, H) will replace (g, h) when the blind signature scheme is used to issue a validator).

VIC will use the blind signature scheme when signing the public part of the validator. The resulting protocol to acquire a validator is:

1. Alice chooses $j, s \in \mathbb{Z}_q$ at random, computes $\alpha := g^s G^j$ and sends this to OA (s is Alice's part of the secret key, and G^j blinds it).
2. OA chooses a random $t \in \mathbb{Z}_q$ and sends $\beta := g^t$ to Alice ($s + t$ is the secret key).
3. Alice computes the public key $h := g^s \beta$ and chooses $k \in \mathbb{Z}_q$ uniformly at random. She then computes the blinded public key $B := hG^{j+k}$ and sends k to OA.
4. OA computes $B := \alpha\beta G^k$ and a signature, $\sigma(B)$, on B using its native signature scheme. OA sends the signature to Alice.
5. Alice verifies the signature received from OA and sends B and $\sigma(B)$ to VIC.
6. VIC verifies the signature to ensure that B was created in cooperation with an observer. He then chooses $w \in \mathbb{Z}_q$ at random, constructs $(a_0, b_0) := (G^w, B^w)$, and computes $V_0 := B^x$. VIC sends a_0, b_0, and V_0 to Alice.
7. Alice chooses $u \in \mathbb{Z}_q^*$ and $v \in \mathbb{Z}_q$ at random and computes $a := a_0^u G^v$ and $b := \left(b_0 / a_0^{j+k}\right)^u h^v$. She computes $V := V_0 / H^{j+k}$ and the corresponding challenge $c := \mathcal{H}(h, V, a, b)$. Alice then computes $c_0 := c/u \bmod q$ and sends c_0 to VIC.
8. VIC computes $r_0 := w + c_0 x$ and sends it to Alice.
9. Alice verifies that

$$G^{r_0} \stackrel{?}{=} a_0 H^{c_0} \qquad \text{and} \qquad h^{r_0} \stackrel{?}{=} b_0 V_0^{c_0}.$$

   She then computes $r := v + ur_0$. The tuple (V, c, r) is the signature on h.

Proposition 5. *If* OA*, Alice and* VIC *all follow the prescribed protocol then Alice gets a correct signature on h.*

Proof. By straightforward computations. □

This protocol can in a natural way be split in two parts. In the first part (corresponding to step 1–4) Alice and OA choose a public key $h = g^{s+t}$, where Alice knows s and OA knows t. In the second part VIC signs this public key.

The first part can very well be performed off-line before the user contacts VIC. Furthermore, the computation of expressions of the form $a^d b^e \bmod p$, where $a, b \in G_q$ and $d, e \in \mathbb{Z}_q$ requires only a little more work than a single exponentiation (less than $2|q|$ multiplications in G_q), see [BC90, B92]. Hence, the on-line computation needed by the user is just a little more than 5 exponentiations in G_q, whereas VIC requires 3 exponentiations and a signature verification (which again requires approximately 2 exponentiations, if the scheme from Section 3.3 is used as the native scheme).

The next proposition says that the key selection part is secure for Alice independently of the computing power of OA (so that requirement 4 of the validator issuing protocol is satisfied)

Proposition 6. *If Alice follows the protocol then no matter what an unlimited powerful cheating observer and center do, they cannot (even together) obtain any Shannon information about the validator.*

Proof. Given a view of OA and a view of VIC it is sufficient to show that for all $h \in G_g$ and for all signatures on h there is exactly one possible choice of s, j, k, u, v by Alice such that OA and VIC would get these views when issuing a validator on h.

Except for the random bits the view of OA consists of

$$\alpha,\ \beta,\ k,\ B, \sigma(B)$$

and the view of VIC of

$$B,\ \sigma(B),\ (a_0, b_0, V_0),\ c_0,\ r_0.$$

Let h and the signature (V, c, r) on h be given. Then k is determined by the observer's view and s and j are determined by

$$s = \log_g(h/\beta) \qquad \text{and} \qquad j = \log_G(B/h) - k.$$

Finally, u and v are determined by

$$u = c/c_0 \bmod q \qquad \text{and} \qquad v = r - ur_0.$$

It can be shown that these values of u and v satisfy

$$a_0^u g^v = g^r h^{-c} \qquad \text{and} \qquad \left(b_0/a_0^{j+k}\right)^u h^v = h^r V^{-c}$$

as they will in an execution of the protocol. □

Supposing the native signature scheme prevents inflow and outflow this proof also implies that observer and center cannot use the protocol as a subliminal channel of information, so that requirement 3 of the validator issuing protocol is satisfied.

Finally we have to consider the security for the organization. In other words we want to show that no polynomially bounded user $\tilde{A}$ can get a validator of a public key for which she knows the corresponding secret key. Assume therefore that OA and VIC both follow the prescribed protocol, and recall that B is the blinded public key, which OA signs.

First note that Assumption 3 implies that $\tilde{A}$ cannot get a signature on a number h for which $\log_g h$ is known unless she knows a pair (d, e) such that $B = g^d G^e$. The following proposition shows that $\tilde{A}$ cannot know such a pair unless it is easy to compute $\log_g G$.

Proposition 7. *Let π denote the probability that $\tilde{A}$ can find a pair (e, d) such that $B = g^d G^e$ (over the random coins of* OA, VIC *and $\tilde{A}$). If π is greater than the inverse of some polynomial for p and q sufficiently large, then there is a probabilistic polynomial time algorithm for finding $\log_g G$.*

Proof. The idea is only sketched. Recall, that none of the participants need to know $\log_g G$. Hence, they may all be cooperate in order to find this logarithm. Consider the following algorithm, which uses the methods of $\tilde{A}$, OA and VIC.

1. $\tilde{A}$ computes α.
2. OA chooses $t \in \mathbb{Z}_q$ at random and computes $\beta = g^t$.
3. $\tilde{A}$ computes k, gets a signature on $B = \alpha\beta G^k$ from OA and a blind signature on B from VIC.
4. $\tilde{A}$ tries to find (d, e) such that $B = g^d G^e$.
   If this fails: stop.
5. Rewind $\tilde{A}$ to after Step 1.
6. Choose $\beta' = g^a G^b$ (a and b chosen at random).
7. Given this, $\tilde{A}$ computes k', gets a signature on $B' = \alpha\beta' G^{k'}$ from OA and a blind signature on B' from VIC.
8. $\tilde{A}$ tries to find (d', e') such that $B' = g^{d'} G^{e'}$.
   If this fails: stop.
9. If $d - t = d' - a$ then stop.
   Output $f := ((d - t) - (d' - a))/((e' - b - k') - (e - k)) \bmod q$.

If Step 9 is reached the algorithm will output $f = \log_g G$ with probability at least $1 - 1/q$ (namely, if $d - t \neq d' - a$). Next it is shown that this step is reached with probability at least $\frac{1}{8}\pi^3$, thus completing the proof.

For a given $\alpha \in G_q$ let p_α denote the probability that $\tilde{A}$ finds the pair (d, e) given α is chosen in the first step (over the random coins of all three parties). Let $Prob[\alpha]$ denote the probability that $\tilde{A}$ chooses α. Then

$$\pi = \sum_\alpha p_\alpha Prob[\alpha].$$

Let E denote the set $\{\alpha \in G_q \,|\, p_\alpha \geq \pi/2\}$. Step 9 will be reached with probability

$$\begin{aligned}\sum_\alpha p_\alpha^2 Prob[\alpha] &\geq \sum_{\alpha \in E} p_\alpha^2 Prob[\alpha] \\ &\geq \frac{\pi^2}{4} Prob[p_\alpha \geq \pi/2].\end{aligned}$$

Finally observe that

$$Prob[p_\alpha \geq \pi/2] \geq \pi/2.$$

□

Note that this shows that requirements 1 and 6 for the validator issuing protocol are satisfied.

Regarding requirement 2 the reader is referred to [CP92] for a discussion of the basic proof and the security of the blind signature protocol. It is argued that it is difficult to forge a signature after a number of executions of the blind signature protocol.

3.5 Validator Signing

It is now shown how Alice and OA can sign a message using the validator obtained above.

It is assumed that Alice initially sends the public key h and the signature on it to the recipient, who then verifies the signature. A message $m \in G_q$ is now signed as follows

1. OA chooses a random $w \in \mathbb{Z}_q$, computes $a_0 := g^w$, $b_0 := m^w$ and $z_0 := m^t$ and sends a_0, b_0 and z_0 to Alice.
2. The verifier chooses $\rho \in \{0,1\}^{|q|}$ at random and sends it to Alice.
3. Alice chooses u and v at random and computes $a := a_0^u g^v$, $b := b_0^u m^v$ and $z := z_0 m^s$. She then computes $c := \mathcal{H}(m, z, a, b, \rho)$ and $c_0 := c/u$. She sends the challenge c_0 to OA.
4. OA computes the response $r_0 := w + c_0 t$ and sends this to Alice.
5. Alice checks that $g^{r_0} = a_0 \beta^{c_0}$ (she knows β from the validator issuing protocol) and $m^{r_0} = b_0 z_0^{c_0}$ (up to this point OA and Alice have used the basic proof. OA provides m^t and proves that $\log_g g^t = \log_m z_0$).
   Alice computes $r := u r_0 + v + cs$ and sends (z, c, r) to the verifier.
6. The verifier computes

$$a = g^r h^{-c} \qquad \text{and} \qquad b = m^r z^{-c}.$$

   and checks that $c = \mathcal{H}(m, z, a, b, \rho)$

By using the same technique as in the blind signature protocol the user can actually obtain a signature on a message of the form $m^l g^n$ for some $l, n \in \mathbb{Z}_q$ while the observer believes it signs m. To prevent this a hash-value of m should be used when b_0 and z_0 are computed.

First note that if all three parties follow the protocol then the verifier will end up with a correct signature on m (requirement 1 for the validator based signature scheme).

The purpose of the random string, ρ, is to prevent Alice from just replaying an old signature on m (see footnote 1). Hence requirement 4 to the validator signing protocol is satisfied.

The following proposition shows that Alice does not compromise her privacy by executing this protocol (requirement 5 for the validator based signature scheme).

Proposition 8. *The validator scheme satisfies Definition 2.*

Proof. Assume Alice follows the protocols. OA, VIC and the verifier may deviate arbitrarily from the protocols, and they may have unlimited computing power.

Let $\text{VIEW}^1_{\text{OA}}$ and $\text{VIEW}^2_{\text{OA}}$ be the views of OA in the validator issuing and signing protocols, respectively. Let furthermore VIEW_{VIC} be the view of VIC in the validator issuing protocol and VIEW_V be the view of the verifier in a signing protocol.

It is sufficient to show that for every triple $(\mathrm{VIEW}^1_{\mathrm{OA}}, \mathrm{VIEW}^2_{\mathrm{OA}}, \mathrm{VIEW}_{\mathrm{VIC}})$ belonging together there is exactly one sequence (s, j, k, u, v) such that this triple could correspond to the given VIEW_V. This can be done by a straightforward extension of the proof of Proposition 6 and is omitted here. □

In [OO90] a general construction of blind signatures based on divertible proofs is presented. Our blind signature uses the same principles, but it has the further (necessary) property that the observer can see the message without compromising the constraints from Definition 2 (the construction of [OO90] does not seem to allow the observer to see m without introducing shared information).

We finally show that it is secure for OA as well as Alice to execute this protocol (requirements 2 and 3 for the validator signing protocol).

Proposition 9. *This protocol satisfies:*

1. *If the basic proof system is secure for the prover then* OA *reveals no more information about its secret key than* m^t.
2. *The verifier alone just gets a random signature on* m.
3. *If* OA *and the verifier share all their information, Alice tells them no more than if she gives them* m^s *and executes the basic proof that* $\log_m m^s$ *equals* $\log_g g^s$.

Proof. The main ideas will be sketched.

The first property is obvious as OA just sends m^t to Alice and executes the basic proof system with her. The second follows from the fact that if Alice follows her protocol, then all possible signatures on m are equally likely.

As for the third assume that OA and VIC get $Z = m^s$ and a chance to execute the basic proof with Alice that it is correct. It can be shown that from such an execution they can generate both of their views from the signature protocol with the correct distribution.

First, the view of the observer can be generated by simply running OA.

Next, observe that OA actually proves knowledge of t such that $\beta = g^t$ (recall that β is the first message which OA sends to Alice in the validator issuing protocol). Hence, it is possible to obtain t by a machine with access to OA, and we can therefore assume that t is known from now on.

As part of the view of OA we got $z_0 = m^t$. In order to generate the view of the verifier we get $Z = m^s$ from Alice and execute the basic protocol proving that $\log_g(h/g^t)$ equals $\log_m Z$. Given (a, b) from Alice the challenge in this proof system is calculated as $c = \mathcal{H}(m, Zz_0, a, b, \rho)$, where the verifier supplies ρ. Alice then responds with some R, and we can compute the correct r corresponding to the signature as $r = R + ct \bmod q$. It is easy to see that this r satisfies

$$g^r = ah^c \qquad \text{and} \qquad m^r = b(Zz_0)^c.$$

Hence, the combined view of OA and V can be generated with almost the same distribution (the generated view is statistically indistinguishable from the real view). □

4 Conclusion and Open Problems

This paper has defined privacy in wallets with observers in such a way that the user is unconditionally protected even if the contents of the observer is later revealed to the organizations.

Furthermore, we have presented protocols for issuing a validator and using a validator to sign messages satisfying this definition. These protocols are just as efficient as previously proposed protocols offering less privacy.

However, the suggested scheme relies on quite strong, but trustworthy, assumptions. It would be interesting to prove (some of) these assumptions or construct another validator scheme based on weaker assumptions.

Acknowledgements

It is a pleasure to thank David Chaum for supporting and stimulating research on wallets with observers and we are grateful for the invitations to CWI which made this work possible. We also kindly acknowledge Jan-Hendrik Evertse's support and many helpful comments during the preparation of earlier work [Cr92] on which this paper is based. Last but not least we thank Stefan Brands, Niels Ferguson and Birgit Pfitzmann for many interesting discussions about wallets with observers.

References

[B92] J. Bos: Practical Privacy, PhD. thesis, Eindhoven University of Technology, The Netherlands, March 1992.

[BC90] J. Bos and M. Coster: Addition Chain Heuristics, Proceedings of Crypto '89, Santa Barbara, August 1989, pp. 400–407.

[BCCFP92] S. Brands, D. Chaum, R. Cramer, N. Ferguson, T. Pedersen: Transaction Systems with Observers. Survey report about wallets with observers, to appear as CWI-technical report.

[Cha83] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology - proceedings of CRYPTO 82*, pp.199–203, 1983.

[Ch92] D. Chaum: Achieving Electronic Privacy, Scientific American, August 1992, pp. 96–101.

[CFN90] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Advances in Cryptology - proceedings of CRYPTO 88*, Lecture Notes in Computer Science, pages 319–327. Springer-Verlag, 1990.

[CP92] D. Chaum and T.P. Pedersen: Wallet Databases with Observers, Proceedings of Crypto '92, Abstracts, Santa Barbara, August 1992, pp. 3.1–3.6.

[Cr92] R.J.F. Cramer: Shared Information in the Moderated Setting, Master's thesis, R.U. Leiden/CWI, August 1992.

[DH76] W. Diffie and M.E.Hellman: New Directions in Cryptography, IEEE Transactions on Information Theory, IT-22(6), November 1976, pp. 644–654.

[EG84] S. Even and O. Goldreich: Electronic Wallet, Proceedings of Crypto'83, Plenum Press 1984, pp. 383–386.

[FS86] A. Fiat and A. Shamir: How to Prove Yourself: Practical Solutions to Identification and Signature Problems, Proceedings of Crypto '86, Santa Barbara, August 1986, pp. 186–194.

[GMR89] S. Goldwasser, S. Micali and C. Rackoff: The Knowledge Complexity of Interactive Proof-Systems, SIAM Journal of Computation, 18(1): 186–208, 1989.

[OO90] T. Okamoto and K. Ohta: Divertible Zero Knowledge Interactive Proofs and Commutative Random Self-Reducibility, Proceedings of Eurocrypt '89, pp. 134–149.

[S91] C.P. Schnorr: Efficient Signature Generation by Smart Cards, J. of Cryptology (1991) 4, pp. 161–174.