Centrum voor Wiskunde en Informatica
REPORT*RAPPORT*

Restrictive blinding of secret-key certificates

S.A. Brands

Computer Science/Department of Algorithmics and Architecture

CS-R9509 1995

Report CS-R9509
ISSN 0169-118X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

Restrictive Blinding of Secret-Key Certificates

Stefan Brands

CWI

P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

Abstract

Many signature transporting mechanisms require a signer to issue triples, consisting of a secret key, a matching public key, and a certificate of the signer on the public key. Of particular interest are so-called restrictive blind signature issuing protocols, in which the receiver can blind the issued public key and the certificate but not a certain predicate of the secret key.

This paper describes the first generally applicable technique for designing efficient such issuing protocols, based on the recently introduced notion of secret-key certificates. The resulting three-move issuing protocols require the receiver to perform merely a single on-line multiplication, and the property of restrictive blinding can be proved with respect to a plausible intractability assumption. Application of the new issuing protocols results in the most efficient and versatile off-line electronic cash systems known to date, without using the blind signature technique developed by Chaum.

AMS Subject Classification (1991): 94A60
CR Subject Classification (1991): D.4.6
Keywords and Phrases: Cryptography, Certificates, Digital signatures, Signature transport.
Note: This paper will appear in the Proceedings of Eurocrypt '95.

1. Introduction

A restrictive blind signature issuing protocol enables an issuer to issue a triple, consisting of a secret key, a matching public key, and a certificate of the issuer on the public key, in such a way that (1) the public key and the certificate can be blinded by the receiver, whereas (2) a certain non-trivial "blinding-invariant" predicate of the secret key cannot. This notion was introduced in [1] for the case that the certificate is a

public-key certificate (that is, a digital signature on the public key), and one particular embodiment was provided. From this embodiment the most efficient privacy-protecting off-line cash system known to that date was constructed, and it was shown how to incorporate several extensions in functionality that had not been realized before. Oddly enough, it is unclear how to generalize the design technique underlying the embodiment in [1] in order to design other embodiments of restrictive blind signature issuing protocols.

Recently [3] the notion of secret-key certificates was introduced. These certificates can be used for secure management of cryptographic keys in much the same way as can public-key certificates. As a consequence it makes sense to issue secret-key certificates by means of a restrictive blind signature issuing protocol. Is there a methodology for designing such issuing protocols for the new type of certificates? This is the problem that we are concerned with in this paper.

The main result of this paper is a technique for converting secret-key certificate schemes into restrictive blind signature schemes. The new technique can be applied to any signature scheme of the so-called Fiat-Shamir type, if only it can be turned into an ordinary blind signature scheme (as defined by Chaum [8]) by applying a divertability technique due to Okamoto and Ohta [19]. No such generally applicable technique is known for public-key certificates.

The advantages of the resulting issuing protocols over the restrictive blind signature issuing protocol in [1] are threefold: each of the new protocols requires the receiver to perform only a single on-line multiplication (all other computations can be performed off-line), as opposed to several hundred in [1]; issuing protocols can be designed that can be used in conjunction with showing protocols that are as secure as the RSA assumption, instead of the Discrete Log assumption; and it can rigorously be proved that a single receiver cannot blind the blinding-invariant predicate of the secret key, assuming only a plausible intractability assumption—no such proof is known for the scheme in [1].

The new technique has a direct bearing on the design of efficient privacy-protecting mechanisms for signature transport; for details, the reader is referred to [4] and [5]. Since none of the new issuing protocols is a blind signature issuing protocol as defined

by Chaum in [8] and his later work, this radically falsifies the popular belief that efficient privacy-protecting off-line electronic cash systems must be based on withdrawal protocols that are blind signature issuing protocols.

This paper is organized as follows. In Sect. 2 background information needed to understand the subsequent exposition is provided. In Sect. 3 the general technique for restrictive blinding of secret-key certificates is explained on the basis of a particular signature scheme. In Sect. 4 the difference between restrictive blinding of secret-key certificates and the blinding technique of Chaum is discussed. Finally, references are provided in Sect. 5 to articles that explain how to apply the new technique to privacy-protecting signature transporting mechanisms.

2. Background

A summary is provided in this section of several basic notions; an understanding of these notions is an absolute prerequisite in order to understand the material in Sect. 3. Because in the next section the technique for designing restrictive blind secret-key certificate issuing protocols will be explained for explicitness in terms of the Guillou-Quisquater signature scheme [17], this particular scheme will serve throughout this section to illustrate the basic notions.

2.1 Digital Signatures

In a digital signature scheme, the objects of interest are pairs consisting of a message and a corresponding digital signature of a signer. A signature scheme consists of several items [16]. First, a verification algorithm that determines what exactly constitutes a digital signature on a message. This algorithm usually is deterministic and can hence be represented in terms of an equality relation. Secondly, a key generation algorithm that generates a key pair for the signer. And thirdly, a signature scheme that specifies an *issuing protocol* between the signer and a receiver.

It is the issuing protocol that we are most concerned with in this paper. The primary purpose of any issuing protocol is to provide a means to ensure that the signer can issue message-signature pairs in a one-to-one correspondence with executions of the issuing protocol. There is virtually no limit to the variety of different issuing protocols that may be used by the signer to issue a certain type of digital signature; it is only on

the basis of its intended purpose and (presumed) security that one particular issuing protocol will be more suitable than another. "Ordinary" signature issuing protocols are only intended to meet the primary purpose mentioned above, and need normally not be interactive. If other purposes are to be met as well, then interaction may need to be incorporated.

The most practical type of digital signature schemes known to date originates from a technique introduced by Fiat and Shamir [15]. For this reason we will henceforth refer to this type of scheme as a Fiat-Shamir type signature scheme. Signature schemes of the Fiat-Shamir type have in common that they are derived from three-move sound identification protocols that do not leak "useful" information about the secret key of the prover (without being zero-knowledge), by replacing the challenge of the verifier by a one-way hash of the message and the information sent by the prover in the first move; the verification relation of the underlying identification protocol determines the signature verification algorithm, and the modified protocol is the signature issuing protocol. As a by-product, the interaction is no longer needed, since the signer can compute the challenge by itself. Among the known Fiat-Shamir type signature schemes are the Fiat-Shamir scheme itself [15], the Feige-Fiat-Shamir scheme [14], the Schnorr scheme [21], the Guillou-Quisquater scheme [17], the Brickell-McCurley scheme [7], and the Okamoto schemes [18]. The ElGamal signature scheme [12] and the DSA [11] can also be seen as being of this type; as a simple exercise one may want to write down the underlying sound identification protocols, and apply the conversion of the challenge to a hash-value of the message and the information provided in the first move. (The subtle difference is that in both the ElGamal scheme and the DSA, which are identical except for the additional "mod q" operator in the verification relation of the latter, one does not *need* to hash in the information provided in the first move; taking $c := \mathcal{H}(m)$ seems to suffice.) Although no reductions from well-known problems to the security of any of these Fiat-Shamir type signature schemes are known, it is generally believed that they are secure.

Since the exposition in Sect. 3 will be in terms of the Guillou-Quisquater scheme, we will summarize this scheme here. The computations in the Guillou-Quisquater signature scheme are performed in a multiplicative group modulo n, denoted by $\mathbb{Z}_n^*$, with n being the product of two distinct large primes. The computations in the exponents

are performed modulo a number v. For convenience, but without loss of generality, we will always assume that v is a prime number that is not a proper divisor of the order $\varphi(n)$ of $\mathbb{Z}_n^*$; another suitable choice would have been to take v to be twice such a prime number. Furthermore, in expressions involving multiplications and divisions of numbers in $\mathbb{Z}_n^*$ the "mod n" operator will never be written down explicitly.

Before describing the three constituents of the Guillou-Quisquater signature scheme, we will state the RSA assumption [20], since this is part of what the scheme derives its presumed security from:

Assumption 1 *Let n denote the product of two distinct prime numbers, v a prime number that is co-prime with $\varphi(n)$ and h_0 an element of $\mathbb{Z}_n^*$. No probabilistic polynomial-time algorithm A, on given as input a triple (n, v, h_0) generated according to some appropriate probability distribution, can output $h_0^{1/v}$ with non-negligible probability of success. The success probability is taken over the coin tosses of A and the probability distribution of the input triple.*

An appropriate probability distribution may select n by generating uniformly at random two hard primes of length k, and multiplying them. From now on, the use of the indication "random" will refer to a probability distribution over the specified set that is polynomially indistinguishable from the uniform distribution, and that is independent of any other event. Without loss of generality, we will assume henceforth that n and v are generated independently of h_0 and are always of the correct form (meaning that, for instance, the probability that v is not prime is zero, instead of merely negligible), and that h_0 is generated at random from $\mathbb{Z}_n^*$.

The *key generation algorithm* for the Guillou-Quisquater signature scheme, on given as input a security parameter k, generates a public key $(n, v, h_0, \mathcal{H}(\cdot))$ and a corresponding secret key $x_0 = h_0^{1/v}$, for use by a probabilistic polynomial-time signer $\mathcal{S}_0$. The triple (n, v, h_0) is generated as specified in the RSA assumption, and $\mathcal{H}(\cdot)$ is a polynomial-size description of a hash-function that maps its inputs to $\mathbb{Z}_{2^t}$ for some appropriate t such that $2^t < v$. The hash-function is generated at random from a suitable family of collision-intractable hash functions. This family preferably is correlation-free, as defined by Okamoto [18].

A *digital signature* on a message m is defined to be a pair (r_0, c_0) such that $c_0 =$

$\mathcal{H}(m, r_0^v h_0^{-c_0})$. If the family of hash-function is sufficiently strong (in particular, correlation-free), then it should be infeasible to algebraically combine several message-signature pairs into a new such pair.

In the *signature issuing protocol*, $\mathcal{S}_0$ issues a signature on a message m to a probabilistic polynomial-time receiver $\mathcal{R}_0$ by generating at random a number $w_0 \in \mathbb{Z}_n^*$, and computing $r_0 := x_0^{c_0} w_0$ and $c_0 := \mathcal{H}(m, w_0^v)$. It is generally believed that executions of this issuing protocol do not help a probabilistic polynomial-time attacker to forge signatures, and so that the primary purpose of an issuing protocol is met by this particular protocol.

2.2 Blind Signature Issuing Protocols

Besides meeting the primary purpose of an "ordinary" signature issuing protocol, in a *blind* signature issuing protocol (a notion introduced by Chaum [8]) an additional property must be satisfied. Namely, the receiver in a blind signature issuing protocol must be able to retrieve a pair in such a way that it is uncorrelated to the view of the signer in the issuing protocol. The computations required of the receiver to achieve this property are commonly referred to as "blinding."

Efficient blind signature issuing protocols are known for a variety of digital signatures. Chaum [9] described a blind signature issuing protocol for issuing one particular type of digital signatures, called RSA signatures [20]. Okamoto and Ohta [19] proposed a general technique that applies to a variety of digital signatures of the Fiat-Shamir type; each of these types of signatures can be issued by means of a blind signature issuing protocol.

The technique of Ohta and Okamoto amounts to *not* removing the interaction when applying the technique of Fiat and Shamir for converting a three-move sound identification protocol into a signature issuing protocol; by having the verifying party (receiver) determine the challenge, one may hope that it can blind the issued message-signature pair. Indeed, as shown by Ohta and Okamoto, for many schemes of the Fiat-Shamir type this works if only a certain random self-reducibility property holds. The technique does not seem applicable to the ElGamal signature scheme and the DSA.

For the Guillou-Quisquater signature issuing protocol, the modification of Ohta and Okamoto results in the following blind signature issuing protocol:

Step 1. $\mathcal{S}_0$ generates at random a number $w_0 \in \mathbb{Z}_v$, and sends $a_0 := w_0^v$ to $\mathcal{R}_0$.

Step 2. $\mathcal{R}_0$ generates at random a number $t_1 \in \mathbb{Z}_n^*$ and a number $t_2 \in \mathbb{Z}_v$, computes $c_0' := \mathcal{H}(m, t_1^v h_0^{t_2} a_0)$ for a message m of its choice, and sends the challenge $c_0 := c_0' + t_2 \bmod v$ to $\mathcal{S}_0$.

Step 3. $\mathcal{S}_0$ sends the response $r_0 := x_0^{c_0} w_0$ to $\mathcal{R}_0$.

$\mathcal{R}_0$ accepts if and only if $r_0^v h_0^{-c_0} = a_0$. (Note that the protocol described up to this point is identical to the Guillou-Quisquater *identification* protocol, with $\mathcal{R}_0$ generating its challenge from $\mathbb{Z}_v$ and in a particular way.) If $\mathcal{R}_0$ accepts, it computes $r_0' := r_0 t_1 h_0^{c_0' + t_2 \operatorname{div} v}$. As can easily be shown, this is a blind signature issuing protocol, with (r_0', c_0') being a Guillou-Quisquater signature on m.

Although it should be easier to forge signatures when the signer uses the new issuing protocol instead of the previous one, since the message can be chosen depending on the information sent by the signer in the first move and the challenge can be freely chosen, the resulting signature scheme is generally believed to be unforgeable. For the blind Guillou-Quisquater signature issuing protocol this belief can be expressed as follows:

Assumption 2 *For any $l \geq 0$, no probabilistic polynomial-time receiver can determine with non-negligible probability of success $l+1$ distinct pairs, consisting of a message and a corresponding Guillou-Quisquater signature, by performing l executions of the blind Guillou-Quisquater signature issuing protocol with an honest signer.*

2.3 Secret-Key Certificate Schemes

Public-key certificates are a well-known cryptographic tool for secure key management. The idea is to have a chosen party, called the (certificate) issuer, certify the public keys of other parties by digitally signing these public keys with respect to its own public key. By widely disseminating the public key of the issuer through a variety of media, anyone can verify that it is genuine. Because a public-key certificate is a digital signature of the issuer on a public key, certificates on public keys of other parties can be publicly verified off-line by using the public key of the issuer. Public-key certificates can be used statically (*e.g.*, in public-key directories) or dynamically, for signature transporting mechanisms.

As with public-key certificates, with *secret-key certificates* [3] the objects of interest are triples consisting of a secret key, a corresponding public key, and a certificate of an issuer on the public key. However, contrary to public-key certificates, in a secret-key certificate scheme the certificate is *not* a digital signature on the public key; the publicly verifiable relation between a public key and a certificate thereon is such that anyone can generate (in isolation) pairs consisting of a public key and a matching certificate, with a distribution that is indistinguishable from the distribution that applies when the issuing protocol is conducted with the issuer. On the other hand, as with public-key certificates, triples consisting of a secret key, a corresponding public key, and a secret-key certificate on the public key can only be retrieved by performing an issuing protocol with the issuer; in effect, the certificate is a digital signature on the secret key. The certificate tells a verifying party that the public key is authentic *if* the presumed owner of the public key knows a corresponding secret key.

Since secret keys are usually not intended to be revealed, how is one to go about in verifying the validity of a secret-key certificate? The answer lies in the fact that a party that can successfully perform a cryptographic action with respect to its public key (such as digital signing or proving knowledge of a corresponding secret key) ordinarily needs to know a corresponding secret key. In fact, there is no point in using a public-key certificate scheme if the cryptographic actions that are to be performed with respect to a certified public key can be performed *without* knowing a corresponding secret key. Consequently, secret-key certificate schemes preserve the functionality intended to be offered by public-key certificate schemes; see [3] for further details.

Regardless of the type of certificate, we will refer to a pair consisting of a public key and a matching certificate as a *certified public key*, and to a triple consisting of a secret key, a corresponding public key, and a matching certificate as a *certified key pair*. A *certificate scheme* consists of several items. Similar to a digital signature scheme, a verification algorithm is needed that determines what exactly constitutes a certificate on a public key. We also need a key generation algorithm that generates key pairs for the issuer, and a certificate issuing protocol. In addition, we need a key generation algorithm for the receiver of a certified key pair; this algorithm specifies which key pairs are considered valid for certification. Finally, in case of a secret-key certificate scheme there must exist a simulator that simulates certified public keys with the same

probability as that by which they are generated by the issuing protocol.

Of course, secret-key certificate schemes and public-key certificate schemes differ in many aspects. An important disadvantage of the former is that they are much harder to design than the latter, especially when the issuer must be prevented from learning the secret keys of the issued triples. On the positive side, with a secret-key certificate scheme the information that is listed in a public-key directory cannot be of help to attack the signature scheme of the issuer; the whole directory could have been generated by an attacker itself. Perhaps the most important advantage of secret-key certificates over public-key certificates is one that is the subject of our studies in this paper: they seem much more suitable for the construction of restrictive blind signature schemes.

In [3] a particular class of embodiments of secret-key certificate schemes was described, based on signature schemes of the Fiat-Shamir type; each of these embodiments can be proved to be as secure as the Fiat-Shamir type signature scheme from which it has been derived, and triples can be issued without the issuer learning the secret keys. According to this construction, a secret-key certificate scheme can be derived from the Guillou-Quisquater signature, as follows. On input a security parameter k, the *key generation algorithm* generates a public key $(n, v, h, g, \mathcal{H}(\cdot))$ and a corresponding secret key (x, y) for the certificate issuer $\mathcal{S}$. Here, $(n, v, h, \mathcal{H}(\cdot))$ and x are generated as described by the key generation algorithm for the Guillou-Quisquater signature scheme, and g and y are generated according to the same distribution as that by which h and x are generated. In particular, $h = x^v$ and $g = y^v$.

A *secret-key certificate* of $\mathcal{S}$ on a public key h_i of a receiver $\mathcal{R}_i$, for some $i \in \mathbb{N}$, is a pair (r, c) such that

$$c = \mathcal{H}(h_i, r^v (h\, h_i)^{-c}).$$

A secret key of $\mathcal{R}_i$ corresponding to its public key h_i is a pair $(s_{0i}, s_{1i}) \in \mathbb{Z}_v \times \mathbb{Z}_n^*$ such that

$$h_i = g^{s_{0i}} s_{1i}^v.$$

(Other choices can be made as well; see [3, 5].)

In [3] an "ordinary" *issuing protocol* is described that enables $\mathcal{S}$ to issue a certified key pair in such a way that it cannot learn a secret key corresponding to the public

key that $\mathcal{R}_i$ ends up with. In Sect. 3 we will develop a new issuing protocol for this particular secret-key certificate scheme, to demonstrate the new technique for designing restrictive blind signature schemes.

3. Restrictive Blinding of Secret-Key Certificates

We now get to the heart of the matter. In a *restrictive* blind signature issuing protocol, the objects of interest are triples, consisting of a secret key, a matching public key, and a certificate on the public key; contrast this to the "ordinary" blind signature issuing protocols discussed in Subsection 2.2, which are concerned with pairs. Similar to ordinary blind signature issuing protocols, the receiver in a restrictive blind signature issuing protocol should be able to ensure that the public key and the certificate of such a triple are uncorrelated to the view of the signer in the issuing protocol. However, the receiver should *not* be able to modify a certain non-trivial predicate of the secret key while blinding about. This is what these issuing protocols derive their name from: on the one hand, the receiver should be able to perform blinding operations, on the other hand the blinding operations at the receiver's disposal should be restricted such as to prevent blinding of the predicate intended by the issuer to be blinding-invariant.

One can distinguish between restrictive blind public-key certificate issuing protocols and restrictive blind secret-key certificate issuing protocols, depending on the type of certificates used. Restrictive blind public-key certificate issuing protocols can be thought of as being a generalization of the withdrawal protocol in the untraceable off-line cash system of Chaum, Fiat and Naor [10].

We now proceed to demonstrate that a *restrictive* blind secret-key certificate issuing protocol can be designed for any Fiat-Shamir type signature scheme for which an *ordinary* blind signature issuing protocol can be constructed by applying the blinding technique of Okamoto and Ohta. A formal description of the general construction would be rather unwieldy, and so we will explain the technique on the basis of secret-key certificates derived from Guillou-Quisquater signatures (which we described in Subsection 2.3); this should make it much easier to understand why the construction works. Exactly the same design technique applies to secret-key certificates based on at least any of the following Fiat-Shamir type signature schemes: Fiat-Shamir [15], Brickell-McCurley [7], Feige-Fiat-Shamir [14], Okamoto [18] (several schemes), and

Schnorr [21].

3.1 Restrictive Blinding for the Guillou-Quisquater Secret-Key Certificate Scheme
In the restrictive blind issuing protocol, $\mathcal{R}_i$ will receive a certified key pair (s_{0i}, s_{1i}), h_i', (r', c'), with the blinding-invariant predicate of the secret key being equal to $s_{0i} \bmod v$. The apostrophes on the public key and the certificate serve to emphasize that they will be uncorrelated to the view of $\mathcal{S}$. We will assume that $\mathcal{S}$ initially provides $\mathcal{R}_i$ with a number s_{0i}, generated according to some appropriate probability distribution over $\mathbb{Z}_v$, and will denote $g^{s_{0i}}$ by h_i. Note that the pair $(s_{0i}, 1)$ is a secret key corresponding to public key h_i; one can think of h_i as the public key that is to be blinded to h_i'. For technical reasons, related to the proof of Proposition 8, we will assume that s_{0i} is generated independently of h. For all practical purposes this is not a restriction.

The issuing protocol is as follows (see Fig. 1):

Step 1. $\mathcal{S}$ generates at random a number $w \in \mathbb{Z}_n^*$, and sends $a := w^v$ to $\mathcal{R}_i$.

Step 2. $\mathcal{R}_i$ generates at random two numbers $s_{1i}, t_1 \in \mathbb{Z}_n^*$, and a number $t_2 \in \mathbb{Z}_v$. $\mathcal{R}_i$ computes $h_i' := h_i s_{1i}^v$, $c' := \mathcal{H}(h_i', t_1^v (h\, h_i)^{t_2} a)$, and sends $c := c' + t_2 \bmod v$ to $\mathcal{S}$.

Step 3. $\mathcal{S}$ sends $r := (xy^{s_{0i}})^c w$ to $\mathcal{R}_i$.

$\mathcal{R}_i$ accepts if and only if $r^v (h\, h_i)^{-c} = a$. If this verification holds, $\mathcal{R}_i$ computes $r' := r\, t_1 (h\, h_i)^{c' + t_2 \operatorname{div} v} s_{1i}^{c'}$.

Observe that we have applied the technique of Okamoto and Ohta to the Guillou-Quisquater signature issuing protocol, with $\mathcal{S}$ performing the protocol with respect to a combined public key that is the product of its own public key and the "not-yet-blinded" public key of $\mathcal{R}_i$, and $\mathcal{R}_i$ blinding not only c and r but also this combined public key. In an (informal) nutshell, this explains the general technique.

Why should this construction work? For this to become clear, we will prove that our exemplary scheme described above is a restrictive blind secret-key certificate scheme; pay particular attention to Proposition 8, as its proof in essence provides the answer to this question.

Following Feige, Fiat and Shamir [14], we will denote by $\overline{\mathcal{Z}}$ a party $\mathcal{Z}$ that follows the issuing protocol, by $\widehat{\mathcal{Z}}$ a probabilistic polynomial-time party $\mathcal{Z}$ that may deviate

from the issuing protocol in an arbitrary way, and by $\tilde{\mathcal{Z}}$ a party $\mathcal{Z}$ with unlimited computing power that may deviate from the issuing protocol in an arbitrary way.

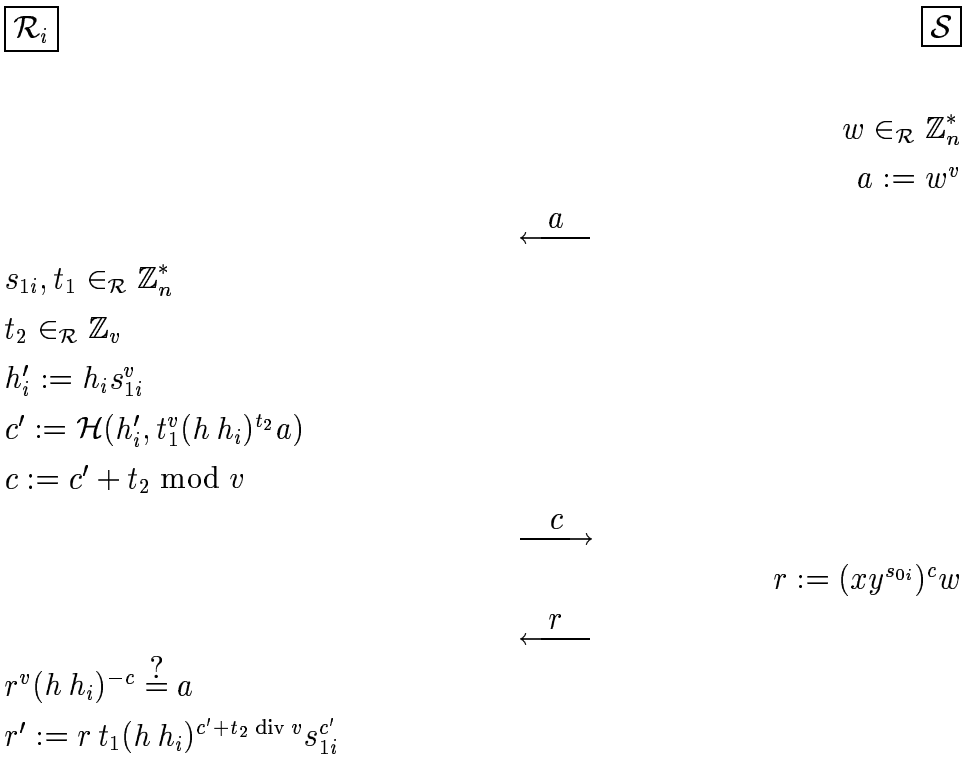

Fig. 1: The issuing protocol

3.2 Proof of Correctness

The secret-key certificate scheme defined in Subsection 2.3, with the issuing protocol of Subsection 3.1 substituted for the ordinary issuing protocol, defines a new certificate scheme. To prove that our new scheme is indeed a secret-key certificate scheme (correctness), we must prove that $\mathcal{R}_i$ in the issuing protocol receives a certified public key. Moreover, since we have changed the issuing protocol it is not immediately clear whether certified public keys can still be generated with indistinguishable probability distribution, without cooperation of $\mathcal{S}$; we will prove that property next.

Proposition 1 *If $\overline{\mathcal{R}_i}$ accepts, then*

$$(s_{0i}, s_{1i}),\ h'_i,\ (r', c')$$

is a certified key pair.

Proof It is clear that (s_{0i}, s_{1i}) is a secret key corresponding to h'_i, and so we must show that (r', c') is a secret-key certificate on h'_i. In Step 2 of the issuing protocol, $\overline{\mathcal{R}_i}$ computes $c' := \mathcal{H}(h'_i, t_1^v (h\, h_i)^{t_2} a)$. Therefore it suffices to prove that $(r')^v (h\, h'_i)^{-c'} = t_1^v (h\, h_i)^{t_2} a$ for the assignments made by $\overline{\mathcal{R}_i}$. This can be seen as follows:

$$\begin{aligned}
(r')^v (h\, h'_i)^{-c'} &= (r\, t_1 (hh_i)^{c'+t_2 \text{ div } v} s_{1i}^{c'})^v (h\, h_i s_{1i}^v)^{-c'} \\
&= (r\, t_1 (h\, h_i)^{c'+t_2 \text{ div } v})^v (h\, h_i)^{-c'} \\
&= r^v t_1^v (hh_i)^{v\,(c'+t_2 \text{ div } v)} (h\, h_i)^{-c'} \\
&\overset{(\star)}{=} ((h\, h_i)^c a)\, t_1^v (hh_i)^{v(c'+t_2 \text{ div } v)} (h\, h_i)^{-c'} \\
&= (h\, h_i)^{(c'+t_2 \bmod v)+v\,(c'+t_2 \text{ div } v)} a t_1^v (h\, h_i)^{-c'} \\
&= (hh_i)^{c'+t_2} t_1^v (h\, h_i)^{-c'} a \\
&= t_1^v (h\, h_i)^{t_2} a.
\end{aligned}$$

The substitution in $(\star)$ is allowed because $\overline{\mathcal{R}_i}$ accepts only if $r^v (h\, h_i)^{-c} = a$. □

Proposition 2 *The new certificate scheme is a secret-key certificate scheme.*

Proof We will construct a polynomial-time simulation algorithm A that generates certified public keys with the same probability as that according to which they are generated in the issuing protocol between $\overline{\mathcal{S}}$ and $\overline{\mathcal{R}_i}$. On input the public key $(n, v, h, g, \mathcal{H}(\cdot))$ of $\mathcal{S}$, A generates at random two numbers $t_1, t_2 \in \mathbb{Z}_n^*$, computes $h_i := h^{-1} t_1^v$, $c := \mathcal{H}(h_i, t_2^v)$ and $r := t_1^c t_2$, and outputs the pair h_i, (r, c). The output of A is a certified public key:

$$\begin{aligned}
c &= \mathcal{H}(h_i, t_2^v) \\
&= \mathcal{H}(h_i, (r\, t_1^{-c})^v) \\
&= \mathcal{H}(h_i, r^v (t_1^v)^{-c}) \\
&= \mathcal{H}(h_i, r^v (h\, h_i)^{-c}).
\end{aligned}$$

Since v is co-prime to $\varphi(n)$, and the numbers t_1, t_2 in Step 1 are chosen at random from $\mathbb{Z}_n^*$, the output distribution of A is identical to the distribution that applies when certified key pairs are issued to $\overline{\mathcal{R}_i}$ by $\overline{\mathcal{S}}$. □

3.3 Proof of the Primary Purpose

We go on to prove that the primary purpose of issuing protocols, namely the one-to-one correspondence between executions of the issuing protocol and certified key pairs, holds for the new certificate scheme. The one-to-one correspondence should hold even if multiple receivers, each of which may perform the issuing protocol for a different number s_{0i}, conspire. A conspiracy should be thought of as being one probabilistic polynomial-time Turing machine, composed of the "participating" receivers; this is easy to formalize, and will be left out here.

We first need two lemmas. In the proof of the first lemma we will construct a simulator A that moves to a third step only after any l executions of the issuing protocol have been simulated. To ensure that A always halts in polynomial time in a defined state, we can let A halt if some polynomial amount of time has expired without any requests for executions of the issuing protocol. To keep the proof simple we will not incorporate such a notion of timing, but will instead implicitly assume its presence; this detail can easily be filled in. The same remark pertains also to the simulators used in later proofs. For a definition of witness hiding, see Feige and Shamir [13].

Lemma 3 *If the blind Guillou-Quisquater signature issuing protocol is witness hiding, then no conspiracy can compute $g^{1/v}$ with non-negligible probability of success.*

Proof Suppose that a conspiracy can misuse l executions of the secret-key certificate issuing protocol to extract with non-negligible probability of success $g^{1/v}$. We will construct a polynomial-time algorithm A for extracting the witness of $\mathcal{S}_0$ in the blind Guillou-Quisquater signature issuing protocol.

Algorithm A, on given as input a public key $(n, v, h_0, \mathcal{H}(\cdot))$ of $\overline{\mathcal{S}_0}$, performs the following steps:

Step 1. (Simulate the key generation for $\mathcal{S}$.) Set $g := h_0$. Generate at random an element $x \in \mathbb{Z}_n^*$, and compute $h := x^v$. The simulated public key of $\overline{\mathcal{S}}$ is $(n, v, h, g, \mathcal{H}(\cdot))$.

Step 2. For each receiver in the conspiracy, simulate the actions that $\overline{\mathcal{S}}$ would perform. For a receiver $\widehat{\mathcal{R}_i}$, perform hereto the simulation as follows:

- (Generation of the blinding-invariant part of the secret key.) Generate a number $s_{0i} \in \mathbb{Z}_v$ for $\widehat{\mathcal{R}_i}$, according to the probability distribution by which $\overline{\mathcal{S}}$ generates the blinding-invariant parts.
- (The issuing protocol.)

  Step 1. Receive a_0 from $\overline{\mathcal{S}_0}$, and pass $a := a_0^{s_{0i}}$ on to $\widehat{\mathcal{R}_i}$.

  Step 2. Receive c from $\widehat{\mathcal{R}_i}$, and pass $c_0 := c$ on to $\overline{\mathcal{S}_0}$.

  Step 3. Receive r_0 from $\overline{\mathcal{S}_0}$, and pass $r := r_0^{s_{0i}} x^c$ on to $\widehat{\mathcal{R}_i}$.

Continue this simulation until l executions of the issuing protocol have been performed.

Step 3. Check if the conspiracy has $g^{1/v}$ on its tapes. If not, then halt.

Step 4. Output $g^{1/v}$.

By definition of the key generation of $\overline{\mathcal{S}_0}$, and that of A, the public key in Step 1 is simulated with the same probability distribution as that by which $\overline{\mathcal{S}}$ generates its public key. The response that is computed by A in the simulated issuing protocol is the same as the response that $\overline{\mathcal{S}}$ would compute:

$$\begin{aligned} r^v &= (r_0^{s_{0i}} x^c)^v \\ &= (r_0^v)^{s_{0i}} (x^v)^c \\ &\overset{(\star)}{=} (h_0^{c_0} a_0)^{s_{0i}} h^c \\ &= (g^{s_{0i}})^{c_0} a_0^{s_{0i}} h^c \\ &= h_i^c \, a \, h^c \\ &= (h \, h_i)^c a, \end{aligned}$$

where the substitution in $(\star)$ is allowed because the response of $\overline{\mathcal{S}_0}$ in the blind Guillou-Quisquater signature issuing protocol is always correct. From this it easily follows that the views provided by A in the simulated issuing protocol have the same distribution as those provided by $\overline{\mathcal{S}}$ in the issuing protocol, regardless of the probability distribution

by which the receivers in the conspiracy generate their challenges. Hence, Step 4 is reached by supposition with non-negligible probability.

To complete the proof, observe that each execution of the simulated issuing protocol constitutes exactly one execution of the blind Guillou-Quisquater signature issuing protocol with $\overline{\mathcal{S}_0}$. For the output of A in Step 4 we have $g^{1/v} = h_0^{1/v}$, which is the secret key of $\overline{\mathcal{S}_0}$. Since A performs only polynomially many executions of the protocol with $\overline{\mathcal{S}_0}$, this contradicts the assumption that the blind Guillou-Quisquater signature issuing protocol is witness hiding. □

Note that a similar result can be proved *unconditionally* for the restrictive blind secret-key certificate schemes that can be derived by our technique from the signature schemes of McCurley-Brickell [7] and Okamoto [18], since these schemes are known to be witness hiding. Furthermore, the result also holds if a conspiracy can "wire-tap" executions of the issuing protocol with honest receivers; in Step 2 of the simulation obviously the same simulation can be used for these honest receivers.

The proof of the following lemma is trivial, and is therefore omitted.

Lemma 4 *If Assumption 2 is true, then the blind Guillou-Quisquater signature issuing protocol is witness hiding.*

We are now prepared to prove the one-to-one correspondence. In the following, a conspiracy is said to be able to *forge a certified key pair* if it can compute with non-negligible probability of success $l + 1$ distinct certified key pairs by performing l executions of the issuing protocol with $\overline{\mathcal{S}}$, for some $l \geq 0$.

Proposition 5 *If Assumption 2 is true, then no conspiracy can forge a certified key pair.*

Proof Suppose that a conspiracy can misuse any l executions of the issuing protocol to extract with non-negligible probability of success l^* *distinct* certified key pairs, with $l^* > l$. We will construct a polynomial-time algorithm A for breaking Assumption 2.

Algorithm A, on given as input a public key $(n, v, h_0, \mathcal{H}(\cdot))$ of $\overline{\mathcal{S}_0}$, performs the following steps:

Step 1. (Simulate the key generation for $\mathcal{S}$.) Set $h := h_0$. Generate at random an element $y \in \mathbb{Z}_n^*$, and compute $g := y^v$. The simulated public key of $\overline{\mathcal{S}}$ is $(n, v, h, g, \mathcal{H}(\cdot))$.

Step 2. For each receiver in the conspiracy, simulate the actions that $\overline{\mathcal{S}}$ would perform. For a receiver $\widehat{\mathcal{R}_i}$, perform hereto the simulation as follows:

- (Generation of the blinding-invariant part of the secret key.) Generate a number $s_{0i} \in \mathbb{Z}_v$ for $\widehat{\mathcal{R}_i}$, according to the probability distribution by which $\overline{\mathcal{S}}$ generates the blinding-invariant parts.
- (The issuing protocol.)

  Step 1. Receive a_0 from $\overline{\mathcal{S}_0}$, and pass $a := a_0$ on to $\widehat{\mathcal{R}_i}$.

  Step 2. Receive c from $\widehat{\mathcal{R}_i}$, and pass $c_0 = c$ on to $\overline{\mathcal{S}_0}$.

  Step 3. Receive r_0 from $\overline{\mathcal{S}_0}$, and pass $r := r_0 y^{s_{0i}c}$ on to $\widehat{\mathcal{R}_i}$.

Continue this simulation until l executions of the issuing protocol have been performed.

Step 3. Check if the conspiracy has l^* distinct certified key pairs on its tapes. If not, then halt.

Step 4. For each of the l^* distinct certified key pairs, (s_{0i}, s_{1i}), h_i', (r', c'), compute $c_0 := c'$, $r_0 := r'(y^{s_{0i}} s_{1i})^{-c'}$ and $m := h_i'$, and output $m, (r_0, c_0)$.

By definition of the key generation of $\overline{\mathcal{S}_0}$, and that of A in Step 1, the public key in Step 1 is generated with the same probability distribution as that by which $\overline{\mathcal{S}}$ generates its public key. The response that is computed by A in the simulated issuing protocol is the same as the response that $\overline{\mathcal{S}}$ would compute:

$$\begin{aligned} r^v &= (r_0 y^{s_{0i}c})^v \\ &= r_0^v (y^v)^{s_{0i}c} \\ &\overset{(\star)}{=} (h_0^{c_0} a_0)(g^{s_{0i}})^c \\ &= (h^c a)\, h_i^c \\ &= (h\, h_i)^c a, \end{aligned}$$

where the substitution in $(\star)$ is allowed because the response of $\overline{\mathcal{S}_0}$ in the blind Guillou-Quisquater signature issuing protocol is always correct. From this it easily follows that the views provided by A in the simulated issuing protocol have the same distribution as those provided by $\overline{\mathcal{S}}$ in the issuing protocol, regardless of the probability distribution by which the conspiracy generates its challenges. Hence, Step 4 is reached by supposition with non-negligible probability.

We next show (i) that the output of A consists of l^* messages with corresponding Guillou-Quisquater signatures, and (ii) that all these pairs are distinct with at least overwhelming probability. Property (i) follows from

$$\begin{aligned} c_0 &= c' \\ &\overset{(\star)}{=} \mathcal{H}(h_i', (r')^v (h\, h_i')^{-c'}) \\ &= \mathcal{H}(h_i', (r_0\, (y^{s_{0i}} s_{1i})^{c'})^v (h\, g^{s_{0i}} s_{1i}^v)^{-c'}) \\ &= \mathcal{H}(h_i', r_0^v h^{-c'}) \\ &= \mathcal{H}(m, r_0^v h_0^{-c_0}), \end{aligned}$$

where the substitution in $(\star)$ is allowed by definition of a secret-key certificate.

To prove property (ii), consider any two certified key pairs,

$$(s_{0i}, s_{1i}),\ h_i,\ (r, c)$$

and

$$(s_{0i}^*, s_{1i}^*),\ h_i^*,\ (r^*, c^*).$$

Suppose that the two corresponding pairs, as computed by A in Step 4, are identical; they are m, $(r\,(y^{s_{0i}} s_{1i})^{-c}, c)$ and m^*, $(r^*(y^{s_{0i}^*} s_{1i}^*)^{-c^*}, c^*)$. We will prove that if these two pairs are identical, then the two certified key pairs are identical. Applying the definition of a secret-key certificate to the two certified key pairs, we have

$$c = \mathcal{H}(h_i,\ r^v (h\, h_i)^{-c})$$

and

$$c^* = \mathcal{H}(h_i^*,\ (r^*)^v (h\, h_i^*)^{-c^*}).$$

Since $h_i = h_i^*$ and $c = c^*$ by equality of the two corresponding pairs, it follows that

$$\mathcal{H}(h_i,\ r^v (h\, h_i)^{-c}) = \mathcal{H}(h_i,\ (r^*)^v (h\, h_i)^{-c}).$$

Because $\mathcal{H}(\cdot)$ is collision-intractable, $r^v (h\, h_i)^{-c} = (r^*)^v (h\, h_i)^{-c}$ with overwhelming probability, and hence $r = r^*$ with overwhelming probability. This leaves us with the possibility that $(s_{0i},\, s_{1i})$ differs from $(s^*_{0i},\, s^*_{1i})$. Suppose that $s_{0i} \neq s^*_{0i} \bmod v$ (the other possibility can be taken care of in a similar way). Let $e, f \in \mathbb{N}$ be such that $(s_{0i} - s^*_{0i})\, e = 1 + f\, v$; such a pair exists because $s_{0i} \neq s^*_{0i} \bmod v$, and can be computed efficiently by applying the extended Euclidean algorithm. Then from $g^{s_{0i}} s_{1i}^v = h_i = h_i^* = g^{s^*_{0i}} (s^*_{1i})^v$ it follows that

$$\begin{aligned} ((s^*_{1i}/s_{1i})^e)^v &= ((s^*_{1i}/s_{1i})^v)^e \\ &= g^{(s_{0i} - s^*_{0i})\, e} \\ &= g^{1+f\, v} \\ &= g\, (g^f)^v, \end{aligned}$$

and so $(s^*_{1i}/s_{1i})^e / g^f = g^{1/v}$. Since the certified key pairs in Step 4 are known by the conspiracy (and so e, f can be computed by the conspiracy), while its view in the simulation is exactly the same as in the issuing protocol, this means that the conspiracy has been able to determine $g^{1/v}$. According to Lemma 4, the blind Guillou-Quisquater signature scheme is witness hiding if Assumption 2 is true, and so by Lemma 3 we have a contradiction with Assumption 2. So the certified key pairs are equal, and hence property (ii) holds.

To complete the proof, observe that an execution of the simulated issuing protocol constitutes exactly one execution of the blind Guillou-Quisquater signature issuing protocol with $\overline{\mathcal{S}_0}$. Because of this one-to-one correspondence, A performs in total l executions of the blind Guillou-Quisquater signature issuing protocol. This contradicts Assumption 2. □

In combination with Proposition 1, which states that an honest receiver receives a certified key pair when it performs an execution of the issuing protocol, this result tells us that there is a one-to-one correspondence between executions of the issuing protocol and certified key pairs. In other words, the primary purpose of issuing protocols is satisfied.

3.4 Proof of the Property of Restrictive Blinding

We next turn to proving that the issuing protocol is a *restrictive blind* signature issuing protocol. We hereto need to prove (1) that the public key and the certificate can be blinded by the receiver, whereas (2) the blinding-invariant predicate of the secret key cannot. We start with the first of these two properties.

Lemma 6 *For any certified public key and any possible view of $\tilde{\mathcal{S}}$ in an execution of the issuing protocol in which $\overline{\mathcal{R}_i}$ accepts, there is exactly one set of random choices that $\overline{\mathcal{R}_i}$ could have made in that execution of the issuing protocol such that $\overline{\mathcal{R}_i}$ would end up with a triple that encompasses that particular certified public key.*

Proof The response r of $\tilde{\mathcal{S}}$ is such that $r^v(h\,h_i)^{-c} = a$, since $\overline{\mathcal{R}_i}$ accepts. We correspondingly define the following set:

$$\begin{aligned}\text{Views}\,(\tilde{\mathcal{S}}) \;=\; \{(a,c,r) \mid\; & a, r \in \mathbb{Z}_n^* \text{ and } c \in \mathbb{Z}_v \text{ such that} \\ & r^v(h\,h_i)^{-c} = a\}.\end{aligned}$$

Furthermore, we introduce the set

$$\text{Choices}\,(\mathcal{R}_i) \;=\; \{(s_{1i}, t_1, t_2) \mid s_{1i}, t_1 \in \mathbb{Z}_n^* \text{ and } t_2 \in \mathbb{Z}_v\}.$$

Consider any certified public key h_i, (r', c'). We will show that for all $\tilde{\mathcal{S}}$-view $\in$ Views $(\tilde{\mathcal{S}})$ there is exactly one triple $(s_{1i}, t_1, t_2) \in$ Choices $(\mathcal{R}_i)$ such that $\tilde{\mathcal{S}}$-view corresponds to an execution of the issuing protocol in which $\overline{\mathcal{R}_i}$ receives the pair h_i, (r', c'). We must take into account that $\tilde{\mathcal{S}}$ can make smart choices for its public key $(n, v, h, g,$ $\mathcal{H}(\cdot))$ and s_{0i}.

Suppose that $\tilde{\mathcal{S}}$-view corresponds to the issuing of the certified public key h_i', (r', c'). We will successively determine uniquely the numbers s_{1i}, t_1, t_2 that must have been chosen by $\overline{\mathcal{R}_i}$. First, s_{1i} is uniquely determined from s_{0i}, h_i' as $s_{1i} = (h_i'\,g^{-s_{0i}})^{1/v}$. Note that s_{1i} exists and is uniquely defined, since v is co-prime to $\varphi(n)$. Next, t_2 is determined from c, c' according to $c = c' + t_2 \bmod v$. Note that t_2 exists and is uniquely defined because $\mathbb{Z}_v$ is a field. Finally, the choices for s_{1i} and t_2, together with r, r' and c', uniquely determine t_1 as $t_1 = r'(r\,t_1(h\,h_i)^{c'+t_2 \operatorname{div} v} s_{1i}^{c'})^{-1}$.

For these choices of the three variables all the assignments and verifications in the execution of the issuing protocol would be satisfied by definition, except maybe for the

assignment $c' := \mathcal{H}(h_i', t_1^v(h\,h_i)^{t_2}a)$ that must have been made by $\overline{\mathcal{R}_i}$. To prove that this assignment hold as well, we notice that by definition of a certified public key we have that $c' = \mathcal{H}(h_i', (r')^v(h\,h_i')^{-c'})$. Therefore, the proof is complete if $(r')^v(h\,h_i')^{-c'} = t_1^v(h\,h_i)^{t_2}a$ for the choices for s_{1i}, t_1 and t_2 made above. This can be derived exactly as in the proof of Proposition 1, considering that the substitution in $(\star)$ is allowed here because $\tilde{\mathcal{S}}$-view $\in$ Views $(\tilde{\mathcal{S}})$. □

Proposition 7 *If $\mathcal{R}_i$ follows the issuing protocol then the issued certified public key is perfectly blinded.*

Proof This is an immediate consequence of Lemma 6 and the fact that $\overline{\mathcal{R}_i}$ generates triples (s_{1i}, t_1, t_2) uniformly at random from Choices $(\mathcal{R}_i)$. □

We now turn to proving the second property, for which we need a new assumption. If we modify the Guillou-Quisquater *identification* protocol by having the prover generate h_0 at random in each execution of the protocol, instead of fixing it initially, the proof of soundness for the Guillou-Quisquater identification still applies. In particular, if the prover can compute correct responses with respect to two different challenges of the verifier, then it "knows" $h_0^{1/v}$. Applying the general technique of Fiat and Shamir for converting identification schemes into signature schemes to this modified "identification" protocol (correspondingly taking c equal to $\mathcal{H}(m, h_0, a_0)$), and leaving out the message m, it follows that it should be infeasible to determine in isolation pairs h_0, (r_0, c_0), without knowing $h_0^{1/v}$, such that $c_0 = \mathcal{H}(h_0, r_0^v h_0^{-c_0})$. Of course, this should also hold if we instead consider satisfying $c_0 = \mathcal{H}(h^{-1}h_0, r_0^v h_0^{-c})$, for a predetermined and randomly chosen $h \in \mathbb{Z}_n^*$. (This variation is considered merely for a technical reason, which will become clear in the proof of the next proposition; alternatively, we could have redefined the verification relation for the issuing protocol.) Substituting $h\,h_i$ for h_0, we get the following assumption.

Assumption 3 *There exists a probabilistic polynomial-time Turing machine M (the knowledge extractor) such that for any probabilistic polynomial-time Turing machine A with work tape WT and random tape RT, and any sufficiently large k, if A on input $(n, v, h, \mathcal{H}(\cdot))$ outputs with nonnegligible probability of success a pair $h_i, (r, c)$ such that*

$$c = \mathcal{H}(h_i, r^v(h\,h_i)^{-c}),$$

then $M(A, WT, RT, (n, v, h, \mathcal{H}(\cdot))) = (h\, h_i)^{1/v}$ *with non-negligible probability. The probability is taken over the coin tosses of* A *and the distribution of the input tuple, and the distribution of the input tuple is as specified for the Guillou-Quisquater signature scheme.*

This assumption is all the more plausible considering that we allow an attacker to forge pairs only *from scratch*, and that the knowledge extractor should succeed with only non-negligible probability.

In the following proposition, we will study the issuing protocol with respect to an "isolated" receiver.

Proposition 8 *If the RSA assumption and Assumption 3 are true, then* $\widehat{\mathcal{R}_i}$ *cannot retrieve with non-negligible probability of success a certified key pair* (s^*_{0i}, s_{1i}), h_i, (r, c) *for which* s^*_{0i} *differs modulo* v *from its blinding-invariant part* s_{0i}.

Proof Suppose that $\widehat{\mathcal{R}_i}$ can misuse l executions of the issuing protocol to extract with non-negligible probability of success a certified key pair such that $s^*_{0i} \neq s_{0i} \bmod v$. We will construct a polynomial-time algorithm A for breaking the RSA assumption.

Algorithm A, on given as input a triple (n, v, h_0) generated as specified in the RSA assumption, performs the following steps:

Step 1. (Simulate the initial key generation.) Set $g := h_0$. Generate at random an element $x \in \mathbb{Z}^*_n$ and an element $s_{0i} \in \mathbb{Z}_v$ (according to the probability distribution by which $\overline{\mathcal{S}}$ generates the blinding-invariant parts), and compute $h := x^v g^{-s_{0i}}$. Generate $\mathcal{H}(\cdot)$ in the same way as described in the key generation for the Guillou-Quisquater signature scheme. The simulated public key of $\overline{\mathcal{S}}$ is $(n, v, h, g, \mathcal{H}(\cdot))$.

Step 2. Simulate for $\widehat{\mathcal{R}_i}$ the actions that $\overline{\mathcal{S}}$ would perform, as follows:

- (Generation of the blinding-invariant part of the secret key.) Use s_{0i} as the invariant part of the secret key for $\widehat{\mathcal{R}_i}$.
- (The issuing protocol.)

  Step 1. Generate at random an element w of $\mathbb{Z}^*_n$. Compute $a := w^v$, and send a to $\widehat{\mathcal{R}_i}$.

Step 2. Receive c from $\widehat{\mathcal{R}_i}$.

Step 3. Compute $r := x^c w$, and send r to $\widehat{\mathcal{R}_i}$.

Continue this simulation until l executions of the issuing protocol with $\widehat{\mathcal{R}_i}$ have been performed.

Step 3. Check if $\widehat{\mathcal{R}_i}$ has on its tapes a certified key pair (s_{0i}^*, s_{1i}), h_i', (r', c') such that $s_{0i}^* \neq s_{0i} \bmod v$. If not, then halt.

Step 4. Run the knowledge extractor M on all tapes of A and $\widehat{\mathcal{R}_i}$, and the tuple $(n, v, h, \mathcal{H}(\cdot))$. Compute $e, f \in \mathbb{N}$ such that $(s_{0i} - s_{0i}^*)\, e = 1 + f\, v$, by applying the extended Euclidean algorithm (such a pair exists since $s_{0i} \neq s_{0i}^* \bmod v$). Denoting the output of M by d, compute $(d\,/x\, s_{1i})^e g^{-f}$, and output the outcome.

Note that we have made use of the knowledge extractor M of Assumption 3 in Step 4 of this simulation, and have viewed $\widehat{\mathcal{R}_i}$ and A as one Turing machine (this can easily be formalized, and is omitted here).

By definition of the key generation of A in Step 1, the public key in Step 1 is simulated with the same probability distribution as that by which $\overline{\mathcal{S}}$ generates its public key. The response that is computed by A in the simulated issuing protocol is the same as the response that $\overline{\mathcal{S}}$ would compute:

$$\begin{aligned} r^v &= (x^c w)^v \\ &= (x^v)^c w^v \\ &= (h g^{s_{0i}})^c a \\ &= (h\, h_i)^c a. \end{aligned}$$

From this it easily follows that the view of $\widehat{\mathcal{R}_i}$ that is provided by A in the simulated issuing protocol has the same distribution as that provided by $\overline{\mathcal{S}}$ in the issuing protocol, regardless of the probability distribution by which $\widehat{\mathcal{R}_i}$ generates its challenges and despite of the tricky way in which A generates h. Hence, Step 4 is reached by supposition with non-negligible probability.

By Assumption 3, the output d of M in Step 4 is equal to $(h\, h_i')^{1/v}$ with non-negligible probability, and in that case we have

$$((d\,/x\, s_{1i})^e)^v = ((d\,/x\, s_{1i})^v)^e$$

$$
\begin{aligned}
&= (d^v / x^v s_{1i}^v)^e \\
&= (h\, h_i' / x^v s_{1i}^v)^e \\
&= (x^v g^{-s_{0i}} g^{s_{0i}^*} s_{1i}^v / x^v s_{1i}^v)^e \\
&= (g^{s_{0i}^* - s_{0i}})^e \\
&= g^{(s_{0i}^* - s_{0i})\, e} \\
&= g^{1+f\, v} \\
&= g\, (g^f)^v.
\end{aligned}
$$

Since $g = h_0$, it follows that $(d\,/x\, s_{1i})^e g^{-f} = h_0^{1/v}$ and so the output of A in Step 4 is equal to $h_0^{1/v}$ with non-negligible probability. This contradicts the RSA assumption. Hence, if Assumption 3 and the RSA assumption are true then it cannot be the case that $s_{0i}^* \neq s_{0i} \bmod v$ with non-negligible probability of success. □

Observe that the proof of this strong result makes use of the fact that the simulator can determine the public key of $\overline{\mathcal{S}}$ in terms of s_{0i}, *without this changing the distribution of the views for* $\mathcal{R}_i$. As a quick comparison with the proof of Proposition 2 will reveal, this proof technique owes its existence to the simulatability of certified public keys that is inherent to secret-key certificates.

3.5 Sequential Executions of the Issuing Protocol

Obviously, in applications of practical interest there are *multiple* receivers, as has been emphasized throughout by the use of a subscript i. We can distinguish between multiple executions of the issuing protocol with respect to the same number s_{0i}, or with respect to different s_{0i}'s. In the former case, Proposition 8 still holds, independent of whether the executions of the issuing protocol are performed sequentially or in parallel by $\mathcal{S}$; but this case is hardly of practical interest. So we will now study the latter case, in which issuing protocols are executed by $\mathcal{S}$ with respect to different blinding-invariant numbers.

When $\mathcal{S}$ sees to it that it performs executions of the issuing protocol with respect to *different* blinding-invariant numbers only *sequentially*, Proposition 8 provides fairly solid evidence that not even a *conspiracy* of receivers will be able to blind the presumed blinding-invariant predicate of one of their secret keys. The motivation for this is

that essentially no cooperation is possible between different receivers while performing executions of the issuing protocol. This seems to justify the following conjecture.

Conjecture 1 *If $\mathcal{S}$ makes sure that it never performs two executions of the issuing protocol in parallel in case they involve different blinding-invariant numbers, then the defined certificate issuing scheme is a restrictive blind signature scheme.*

It is stressed that it cannot hurt if $\mathcal{S}$ performs executions of the issuing protocol in parallel that pertain to the *same* blinding-invariant number.

In many practical applications the simple measure of not running executions of the issuing protocol in parallel, in case they pertain to different blinding-invariant parts, will certainly suffice. Namely, consider two such executions of the issuing protocol. $\mathcal{S}$ starts by sending a number a in the first execution. As soon as $\mathcal{S}$ has received a challenge number c for this first execution it starts the second execution; the execution of the first protocol can be completed while the second is still running. Even though the protocol executions overlap in part in this manner, they clearly are performed sequentially. The only way in which the first receiver can hold up the second is by not immediately responding with a challenge; but $\mathcal{S}$ can simply refuse to provide a corresponding response if the delay between sending out a and receiving c is too large, and proceed anyways. A semaphore, or flag, can keep track of whether an execution of the issuing protocol is still "active."

What constitutes a "fair" permitted delay depends on the infrastructure of the particular network that is being used, and on the computation speed of receivers. Receivers that accidentally exceed the allowed delay can simply try again in a new execution of the protocol (feedback). In this respect it is important to note that the receiver in the issuing protocol (and in all other issuing protocols that result from the presented technique) needs to perform only a single on-line (modular) multiplication, to compute the second argument of c'; the other computations can be performed off-line. (The computation of the hash-value of the two arguments, and of the addition modulo v for c, can be done comparatively very fast so that we can ignore it in our argument.) A rough lower bound for a fair delay time (which may even vary per receiver) is to double the time needed for the information to travel through the network from $\mathcal{S}$ to $\mathcal{R}_i$ and back, and add to that the time needed by the receiver to perform the on-line

multiplication; the chosen value for the allowed delay time need usually not exceed this lower bound by much. In case $\mathcal{S}$ can rely on the security of certain parts of the network infrastructure, significant reductions can be obtained. For example, if receivers can perform the issuing protocol by inserting a smart card into one of many secured terminals (or pointing a hand held with infrared communication channel), and $\mathcal{S}$ trusts these terminals, then only the time for the information to travel between smart card and terminal need be considered; this will be negligible.

If requests for executing the issuing protocol "arrive" like a Poisson process, this strategy in effect is the M/D/1 model with feedback from queueing theory. Of course, the feedback may be purposely limited by $\mathcal{S}$, to shut out receivers that frequently exceed the permitted delay. Efficiency improvements can be made by letting $\mathcal{S}$ use k secret keys, instead of a single one. This allows k times as many receivers to be served in the same time span, since executions with respect to independently generated secret keys of $\mathcal{S}$ can obviously be performed in parallel without danger. In effect, we then switch to the M/D/k model (with feedback). The trade-off is that the unlinkability of views to certified public keys does not hold with respect to different public keys of $\mathcal{S}$.

3.6 Parallel Executions of the Issuing Protocol

Conjecture 1 is false if $\mathcal{S}$ performs executions of the issuing protocol *in parallel* when *different* blinding-invariant numbers are involved. Let s_{0i} be the blinding-invariant number for $\mathcal{R}_i$, and $s_{0j} \neq s_{0i} \bmod v$ that for $\mathcal{R}_j$; the corresponding "not-yet-blinded" public keys are h_i and h_j. In its simplest form (leaving out additional computations that need to be performed to completely blind the certified key pair, in order to prevent unduly obscuring of the description), the attack on the two parallel executions of the issuing protocol is the following:

(Step 1 for $\mathcal{R}_i$) $\mathcal{S}$ generates at random a number $w_i \in \mathbb{Z}_v$, and sends $a_i := w_i^v$ to $\mathcal{R}_i$.

(Step 1 for $\mathcal{R}_j$) $\mathcal{S}$ generates at random a number $w_j \in \mathbb{Z}_v$, and sends $a_j := w_j^v$ to $\mathcal{R}_j$.

(Cooperation between $\mathcal{R}_i$ and $\mathcal{R}_j$) $\mathcal{R}_i$ and $\mathcal{R}_j$ compute $h_k := g^{s_{0k}}$ for an arbitrary number s_{0k} of their choice (s_{0k} need not be in $\mathbb{Z}_v$; any number in $\mathbb{N}$ will do). They then compute $c_k := \mathcal{H}(h_k, a_i a_j)$.

(Step 2 for $\mathcal{R}_i$) $\mathcal{R}_i$ sends $c_i := c_k\,(s_{0j} - s_{0k})(s_{0j} - s_{0i})^{-1} \bmod v$ to $\mathcal{S}$.

(Step 2 for $\mathcal{R}_j$) $\mathcal{R}_j$ sends $c_j := c_k\,(s_{0k} - s_{0i})(s_{0j} - s_{0i})^{-1} \bmod v$ to $\mathcal{S}$.

(Step 3 for $\mathcal{R}_i$) $\mathcal{S}$ sends $r_i := (xy^{s_{0i}})^{c_i} w_i$ to $\mathcal{R}_i$.

(Step 3 for $\mathcal{R}_j$) $\mathcal{S}$ sends $r_j := (xy^{s_{0j}})^{c_j} w_j$ to $\mathcal{R}_j$.

$\mathcal{R}_i$ and $\mathcal{R}_j$ accept if and only if

$$r_i^v (h\,h_i)^{-c_i} = a_i \quad \text{and} \quad r_j^v (h\,h_j)^{-c_j} = a_j.$$

If this verification holds, then $\mathcal{R}_i$ and $\mathcal{R}_j$ compute

$$r_k := r_i r_j h^{-((c_i+c_j) \operatorname{div} v)} g^{-((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)} g^{c_k s_{0k} \operatorname{div} v}.$$

Here, and in the proof below, we make abundant use of parenthesis in the expressions in the exponent, in order to prevent confusion about the priority of the involved operators.

Proposition 9 *If $\mathcal{R}_i$ and $\mathcal{R}_j$ accept, then*

$$(s_{0k}, 1),\ h_k,\ (r_k, c_k)$$

is a certified key pair.

Proof It is clear that $h_k = g^{s_{0k}} 1^v$, and so we must show that $h_k, (r_k, c_k)$ is a certified public key, *i.e.*, that

$$c_k = \mathcal{H}(h_k, r_k^v (h\,h_k)^{-c_k}).$$

Since $\mathcal{R}_i$ and $\mathcal{R}_j$ compute c_k according to $c_k := \mathcal{H}(h_k, a_i a_j)$, this follows from:

$$\begin{aligned}
a_i a_j &= r_i^v (h\,h_i)^{-c_i} r_j^v (h\,h_j)^{-c_j} \\
&= (r_i r_j)^v h^{-(c_i+c_j)} h_i^{-c_i} h_j^{-c_j} \\
&= (r_i r_j)^v h^{-(c_i+c_j)} g^{-(c_i s_{0i} + c_j s_{0j})} \\
&= (r_i r_j)^v h^{-((c_i+c_j) \bmod v) - v\,((c_i+c_j) \operatorname{div} v)} \\
&\quad\ g^{-((c_i s_{0i} + c_j s_{0j}) \bmod v) - v\,((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)} \\
&= \left(r_i r_j h^{-((c_i+c_j) \operatorname{div} v)} g^{-((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)}\right)^v \\
&\quad\ h^{-((c_i+c_j) \bmod v)} g^{-((c_i s_{0i} + c_j s_{0j}) \bmod v)} \\
&\overset{(\star)}{=} \left(r_i r_j h^{-((c_i+c_j) \operatorname{div} v)} g^{-((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)}\right)^v
\end{aligned}$$

$$
\begin{aligned}
& h^{-c_k} g^{-((c_i s_{0i} + c_j s_{0j}) \bmod v)} \\
\overset{(\star\star)}{=} \; & (r_i r_j h^{-((c_i+c_j) \operatorname{div} v)} g^{-((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)})^v \\
& h^{-c_k} g^{-(c_k s_{0k} \bmod v)} \\
= \; & (r_i r_j h^{-((c_i+c_j) \operatorname{div} v)} g^{-((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)})^v \\
& h^{-c_k} g^{-c_k s_{0k} + v\,((c_k s_{0k}) \operatorname{div} v)} \\
= \; & (r_i r_j h^{-((c_i+c_j) \operatorname{div} v)} g^{-((c_i s_{0i} + c_j s_{0j}) \operatorname{div} v)} g^{((c_k s_{0k}) \operatorname{div} v)})^v \\
& h^{-c_k} g^{-c_k s_{0k}} \\
= \; & r_k^v h^{-c_k} (g^{s_{0k}})^{-c_k} \\
= \; & r_k^v (h\, h_k)^{-c_k}.
\end{aligned}
$$

The substitution in $(\star)$ is allowed because $2^t < v$ and so

$$
\begin{aligned}
(c_i + c_j) \bmod v \; = \; & (c_k (s_{0j} - s_{0k})(s_{0j} - s_{0i})^{-1} \bmod v \\
& + c_k (s_{0k} - s_{0i})(s_{0j} - s_{0i})^{-1} \bmod v) \bmod v \\
= \; & (c_k (s_{0j} - s_{0k})(s_{0j} - s_{0i})^{-1} \\
& + c_k (s_{0k} - s_{0i})(s_{0j} - s_{0i})^{-1}) \bmod v \\
= \; & c_k (s_{0j} - s_{0i})(s_{0j} - s_{0i})^{-1} \bmod v \\
= \; & c_k \bmod v \\
= \; & c_k.
\end{aligned}
$$

Likewise, since

$$
\begin{aligned}
(c_i s_{0i} + c_j s_{0j}) \bmod v \; = \; & (s_{0i}((c_k s_{0j} - c_k s_{0k})(s_{0j} - s_{0i})^{-1} \bmod v) \\
& + s_{0j}((c_k s_{0k} - c_k s_{0i})(s_{0j} - s_{0i})^{-1} \bmod v)) \bmod v \\
= \; & (s_{0i}(c_k s_{0j} - c_k s_{0k})(s_{0j} - s_{0i})^{-1} \\
& + s_{0j}(c_k s_{0k} - c_k s_{0i})(s_{0j} - s_{0i})^{-1}) \bmod v \\
= \; & (s_{0j} c_k s_{0k} - s_{0i} c_k s_{0k})(s_{0j} - s_{0i})^{-1} \bmod v \\
= \; & c_k s_{0k} \bmod v
\end{aligned}
$$

the substitution in $(\star\star)$ is allowed. □

In case the application really demands that $\mathcal{S}$ can securely run executions of the issuing protocol in parallel, without any restrictions, we must therefore alter the issuing

protocol. The following minor adjustment is believed to suffice for this purpose. Rather than revealing s_{0i} to $\mathcal{R}_i$ *before* the execution of the issuing protocol, $\mathcal{S}$ does not make it known until it has received c of $\mathcal{R}_i$ in Step 2; instead, $\mathcal{S}$ only makes $h_i = g^{s_{0i}}$ known initially. Furthermore, $\mathcal{S}$ chooses s_{0i} at random from $\mathbb{Z}_v$: computing s_{0i} from h_i then requires breaking the *Discrete Log* assumption in $\mathbb{Z}_n^*$. This modification does not in any way prevent $\mathcal{R}_i$ from performing the necessary computations in Step 2, since only h_i needs to be known to $\mathcal{R}_i$.

Observe that the attack described above requires the attacking receivers to compute their respective challenges in terms of s_{0i} and s_{0j}, and so this particular attack no longer works for the adjusted issuing protocol.

Is the modified issuing protocol secure under parallel executions? I strongly believe so, although I have not been able to come up with a proof with respect to some reasonable intractability assumption. If Conjecture 1 is true then any successful attack must be such that each of the two challenges c_i, c_j depends on each of a_i, a_j. Based on this observation, an argument for the security of the modified protocol can be given. Because of the hand-waving nature of this argument, it will not be included here; the interested reader can find it in the appendix.

3.7 Encoding specific information into the blinding-invariant part

If Conjecture 1 is true, then the non-adjusted version of the issuing protocol is for sequential executions *regardless of the distribution of* s_{0i}, since none of the results that we have proved depends on the distribution of s_{0i}. Therefore, $\mathcal{S}$ can use the entire number s_{0i} to encode specific information in, representing for instance a quantitative credential [5]; $\mathcal{S}$ in effect then generates s_{0i} according to a highly degenerate distribution.

In the modified protocol not the entire blinding-invariant number s_{0i} can be used to represent specific information, since it should be infeasible for the receivers to compute $\log_g h_i$. It is immediately clear, though, that $\mathcal{S}$ can use part of the bits of s_{0i} (in general, a predicate of s_{0i}) for this purpose, as long as the Discrete Log problem in $\mathbb{Z}_n^*$ remains intractable with respect to the chosen distribution for s_{0i}. But we can do even better than this. Observe that the only purpose of the modification to the issuing protocol is to ensure that s_{0i} cannot be computed from h_i *in the time period between receiving*

a and returning c. If we use the queueing measures described in Subsection 3.5, this may leave the attacking receivers in a realistic application with perhaps no more than a fraction of a second to compute a discrete logarithm with respect to g, and so the probability distribution for s_{0i} can be highly degenerate. In other words, $\mathcal{S}$ can use almost all the bits of s_{0i} to encode specific information in.

This concludes the exposition of the general technique for designing restrictive blind secret-key certificate issuing protocols. Although the description has been based on the Guillou-Quisquater signature scheme, enough handles have been provided throughout to easily apply the technique to any of the other Fiat-Shamir type signature schemes that can be subjected to the technique of Ohta and Okamoto for designing ordinary blind signature issuing protocols.

4. Relation to Blind Signature Issuing Protocols

Contrary to restrictive blind issuing protocols for public-key certificates, restrictive blind issuing protocols for secret-key certificates are *not* a particular case of "ordinary" blind signature issuing protocols. Consider a triple consisting of a secret key, a matching public key, and a certificate on the public key. The receiver in the certificate issuing protocol can completely blind the public key and the certificate, but not part of the secret key. If the certificate would be a public-key certificate, then the protocol would indeed be a particular case of an ordinary blind signature scheme; the public key is the message and the certificate is the signature on the message, and the pair is blinded.

However, if the certificate is a secret-key certificate, it is by definition *not* a digital signature on the public key (the extreme opposite is true: pairs consisting of a public key and a matching secret-key certificate can be generated by anyone with exactly the same probability distribution); the *secret key* is the message, and the certificate is the signature on the message. But the message *cannot* be blinded, by the very definition of a restrictive blind signature issuing protocol; only the signature can.

5. Conclusion

A variety of privacy-protecting signature transporting mechanisms can be obtained by combining the new restrictive blind signature issuing protocols with an appropriate

showing protocol between the receiver of the issued triple and a third party. One particularly interesting such signature transporting mechanism is an untraceable off-line electronic cash system, first studied by Chaum, Fiat and Naor [10]. In [1] I introduced the most efficient and versatile untraceable off-line cash system known to that date. As will be appreciated, the new issuing protocols can be combined fairly straightforwardly with the techniques developed in [1] for achieving prior restraint of double-spending with fall-back to traceability after the fact. By basing the restrictive blind secret-key certificate issuing protocol on the Schnorr signature scheme, the resulting cash system is considerably more efficient than the system in [1]. The interested reader is referred to [4], and to [2] for practical optimizations. In [6] the application to Internet payments is discussed.

In [5] general techniques are described for designing showing protocols that can be combined with restrictive blind signature issuing protocols in order to design general privacy-protecting signature transporting mechanisms (also known as credential mechanisms, first studied by Chaum [9]). Here, again, the use of restrictive blind issuing protocols instead of a cut-and-choose issuing protocol enables significant improvements in terms of efficiency, functionality, and provability of security properties of the credential mechanisms. Instead of using Chaum's technique of encoding different types of credentials by using different signatures schemes, in the new credential mechanisms credentials are encoded into the blinding-invariant parts of secret keys; this enables the holder of a set of credentials to prove a variety of predicates of his credentials without providing additional information.

As the reader may have noticed, we have never used the fact that $\mathcal{S}$ may know the factorization of the modulus n. The sole reason for not having done so is that in Fiat-Shamir type signature schemes that are based on the Discrete Log assumption, such as the Schnorr signature scheme, the signer also does not have such trapdoor information at its disposal, and our goal was to explain a generally applicable technique. However, if we allow $\mathcal{S}$ to know (and make use of) the factorization of n, a powerful technique becomes available, which enables the issuer in the credential mechanisms of [5] to update credentials without needing to know their current values. This updating technique is not possible in credential mechanisms based on cut-and-choose issuing protocols.

In sum, the demonstrated technique for designing efficient secret-key certificate issuing protocols has a direct bearing on the design of efficient and versatile privacy-protecting credential mechanisms.

References

1. Brands, S., "Untraceable Off-Line Cash in Wallet with Observers," Advances in Cryptology – CRYPTO '93, Lecture Notes in Computer Science, no. 773, Springer-Verlag, pp. 302–318. An extended pre-print appeared as: "An efficient off-line electronic cash system based on the representation problem," Centrum voor Wiskunde en Informatica (CWI), Report CS-R9323, March 1993. Available by anonymous ftp from: ftp.cwi.nl:/pub/CWIreports/AA/CS-R9323.ps.Z.

2. Brands, S., "Off-line Cash Transfer by Smart Cards," Proceedings of the First Smart Card Research and Advanced Application Conference, Lille (France), Oct. 1994, pp. 101–117. See also: Centrum voor Wiskunde en Informatica (CWI), Report CS-R9455, September 1994. Available by anonymous ftp from: ftp.cwi.nl:/pub/CWIreports/AA/CS-R9455.ps.Z.

3. Brands, S., manuscript (1993) part (i): "Secret-Key Certificates," submitted for publication. Preprint available on request.

4. Brands, S., manuscript (1993) part (iii): "Off-Line Electronic Cash Based on Secret-Key Certificates," To appear in: Proceedings of the Second International Symposium of Latin American Theoretical Informatics (LATIN '95), Valparaíso, Chili, April 3–7, 1995. See also: Centrum voor Wiskunde en Informatica (CWI), Report CS-R9506, Januari 1995. Available by anonymous ftp from: ftp.cwi.nl:/pub/CWIreports/AA/CS-R9506.ps.Z.

5. Brands, S., manuscript (1993) part (v): "Privacy-protecting Digital Credentials Based on Restrictive Blinding," submitted for publication. Preprint available on request.

6. Brands, S., "Electronic Cash on the Internet," To appear in: Proceedings of the Internet Society 1995 Symposium on Network and Distributed System Security, San Diego, California, Februari 16-17, 1995.

7. Brickell, E., McCurley, K., "An Interactive Identification Scheme Based on Discrete

Logarithms and Factoring," Journal of Cryptology, Vol. 5, No. 1 (1992), pp. 29–39.

8. Chaum, D., "Blind Signatures for Untraceable Payments," Advances in Cryptology – CRYPTO '82, Lecture Notes in Computer Science, Springer-Verlag, pp. 199–203.

9. Chaum, D., "Security without identification: transaction systems to make big brother obsolete," CACM Vol. 28, No. 10, October 1985, pp. 1030–1044.

10. Chaum, D., Fiat, A., Naor, M., "Untraceable electronic cash," Advances in Cryptology – CRYPTO '88, Lecture Notes in Computer Science, no. 403, Springer-Verlag, pp. 319–327.

11. NIST, "Specifications for a digital signature standard (DSS)," Federal Information Processing Standards Pub. (draft), Aug. 19, 1991.

12. ElGamal, T., "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, Vol. IT-31, No. 4, July 1985, pp. 469–472.

13. Feige, U., Shamir, A., "Witness Indistinguishable and Witness Hiding Protocols," Proceedings of the 22nd Annual ACM Symposium on the Theory of Computing, 1990, pp. 416–426.

14. Feige, U., Fiat, A., Shamir, A., "Zero-Knowledge Proofs of Identity," Journal of Cryptology, Vol. 1, No. 2 (1988), pp. 77–94.

15. Fiat, A., Shamir, A., "How to prove yourself: practical solutions to identification and signature problems," Advances in Cryptology – CRYPTO '86, Lecture Notes in Computer Science, Springer-Verlag, pp. 186-194.

16. Goldwasser, S., Micali, S., Rivest, R., "A digital signature scheme secure against adaptive chosen message attack," SIAM Journal on Computing, Vol. 17 No. 2 (1988), pp. 281–308.

17. Guillou, L., Quisquater, J.-J., "A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory," Advances in Cryptology – EUROCRYPT '88, Lecture Notes in Computer Science, no. 330, Springer-Verlag, pp. 123-128.

18. Okamoto, T., "Provably Secure and Practical Identification Schemes and Corre-

sponding Signature Schemes," Advances in Cryptology – CRYPTO '92, Lecture Notes in Computer Science, no. 740, Springer-Verlag, pp. 31–53.

19. Okamoto, T., Ohta, K., "Divertible Zero-Knowledge Interactive Proofs and Commutative Random Self-Reducibility," Advances in Cryptology – EUROCRYPT '89, Lecture Notes in Computer Science, no. 434, Springer-Verlag, pp. 481–496.

20. Rivest, R., Shamir, S., Adleman, L., "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, Feb. 1978, pp. 120-126.

21. Schnorr, C, "Efficient Signature Generation by Smart Cards," Journal of Cryptology, Vol. 4, No. 3 (1991), pp. 161-174.

THE SECURITY OF THE MODIFIED ISSUING PROTOCOL FOR PARALLEL EXECUTIONS

As announced in Subsection 3.6 an argument will be provided here for the security of the modified issuing protocol under parallel executions. Although the line of arguing presented below may seem to be very restricted, some study will reveal that most attacks that come to mind amount to the attack considered by the argument.

Consider an "algebraic" attack on the parallel version of the modified restrictive blind signature issuing protocol. For the sake of simplicity, we will restrict ourselves to two parallel executions of the issuing protocol, each with respect to a different blinding-invariant number; the argument can easily be generalized. Since v-th powers can always be multiplied in and out of the verification relation, without loss of generality we furthermore will leave s_{1i} out of our considerations. Raising the verification relations for each of the two protocol executions to a power, and multiplying the results, we obtain:

$$\left.\begin{array}{l}(r_i^v)^{l_i} = (h\, g^{s_{0i}})^{l_i c_i} a_i^{l_i} \\ (r_j^v)^{l_j} = (h\, g^{s_{0j}})^{l_j c_j} a_j^{l_j}\end{array}\right\} \Rightarrow (r_i^{l_i} r_j^{l_j})^v = h^{l_i c_i + l_j c_j}\, g^{l_i s_{0i} c_i + l_j s_{0j} c_j} (a_i^{l_i} a_j^{l_j}).$$

The goal of the attackers is to determine a number s_{0k}, not equal modulo v to each of s_{0i} and s_{0j}, and numbers l_i, l_j, c_i, c_j and c_k for which the responses r_i and r_j can be combined into a response r_k such that (c_k, r_k) is a secret-key certificate on $g^{s_{0k}}$. At the time the attackers have to provide c_i and c_j to $\mathcal{S}$ they only have two random numbers a_i and a_j of $\mathcal{S}$ at their disposal. If Conjecture 1 is true then each of c_i and c_j

must depend on both a_i and a_j if the attack is to have a significant success probability. Setting

$$c_k := \mathcal{H}(g^{s_{0k}}, a_i^{l_i} a_j^{l_j}), \quad r_k := r_i^{l_i} r_j^{l_j}, \quad a_k := a_i^{l_i} a_j^{l_j},$$

the attackers must ensure that

$$h^{l_i c_i + l_j c_j} g^{l_i s_{0i} c_i + l_j s_{0j} c_j} = (h\, g^{s_{0k}})^{c_k}.$$

This can be solved for $(l_i, l_j, s_{0k}, c_i, c_j)$ by the attackers if they can solve

$$\begin{pmatrix} l_i & l_j \\ l_i s_{0i} & l_j s_{0j} \end{pmatrix} \cdot \begin{pmatrix} c_i \\ c_j \end{pmatrix} = c_k \cdot \begin{pmatrix} 1 \\ s_{0k} \end{pmatrix} \bmod v,$$

since the remaining "div v" terms can be multiplied into r_k later on. Since $\mathbb{Z}_v$ is a field, and the matrix on the left-hand side is non-singular because $s_{0i} \neq s_{0j} \bmod v$, this leads to:

$$\begin{pmatrix} c_i \\ c_j \end{pmatrix} = \begin{pmatrix} (c_k / l_i)\,(s_{0j} - s_{0k}) \,/\, (s_{0j} - s_{0i}) \\ (c_k / l_j)\,(s_{0k} - s_{0i}) \,/\, (s_{0j} - s_{0i}) \end{pmatrix} \bmod v.$$

Knowing neither one of s_{0i}, s_{0j} because of the intractability of the Discrete Log problem, it seems that the attackers must determine s_{0k}, l_i, l_j such that neither of c_i and c_j depends on s_{0i} or s_{0j}. In other words, s_{0k}, l_i, l_j must be chosen in such a way that s_{0i} and s_{0j} drop out. Taking s_{0k} equal to s_{0i} or s_{0j} modulo v would obviously work, but this does not meet the goal of the attackers (that is, this is legitimate behavior, not an attack).

From $c_k = \mathcal{H}(g^{s_{0k}}, a_i^{l_i} a_j^{l_j})$ we see that c_k depends on each of s_{0k}, l_i and l_j. Because $\mathcal{H}(\cdot)$ is a (correlation-free) collision-intractable hash-function, it should be infeasible to determine c_k as an "algebraic" function of s_{0k}, l_i, l_j. What this means is that c_k / l_i and c_k / l_j can be chosen independently at random by the attackers, but not such that c_k is independent of s_{0k}: if s_{0k} is varied then c_k is implicitly varied along.

The gist of this argument is that it seems infeasible to misuse two parallel executions of the issuing protocols in such a way that the unknown blinding-invariant numbers s_{0i} and s_{0j} drop out of the matrix equation.