Chapter 4

Restrictive Blind Issuing Protocols

In this chapter we introduce a new notion, called restrictive blinding, to enable the CA to encode attributes into certified key pairs that are unlinkable and untraceable in all other respects. We design various practical restrictive blind certificate issuing protocols, for DLREP-based certificates as well as for RSAREP-based certificates, and analyze their security. This chapter builds on Chapter 2, but may be read independently of Chapter 3. In Chapter 5 we will show how to combine the issuing and showing protocol techniques.

4.1 Restrictive blinding

Informally, a *restrictive blind* certificate scheme is a digital certificate scheme (see Section 2.6) with the following properties:

- If $\mathcal{V}$ and $\mathcal{P}$ both follow the protocol, then $\mathcal{V}$ obtains a certified key pair

  $$(s, p, \mathsf{cert}(p)).$$

  The pair (s, p) is a key pair of $\mathcal{V}$, and $\mathsf{cert}(p)$ is $\mathcal{P}$'s (secret-key or public-key) digital certificate on $\mathcal{V}$'s public key p.
- The certified public key $(p, \mathsf{cert}(p))$ obtained by $\overline{\mathcal{V}}$ by interacting with $\widetilde{\mathcal{P}}$ is statistically independent from $\widetilde{\mathcal{P}}$'s view in the protocol execution.
- If $\mathcal{P}$ follows the protocol, then $\widehat{\mathcal{V}}$ cannot forge certified key pairs.
- If $\widehat{\mathcal{V}}$ obtains a certified key pair $(s, p, \mathsf{cert}(p))$, then with overwhelming probability the secret key s contains at least one attribute encoded by $\overline{\mathcal{P}}$.

The last property is the hardest to formalize. To understand the meaning of "encoding at least one attribute," consider by way of example a scenario in which $\mathcal{V}$ is to receive $\mathcal{P}$'s certificate on a public key defined by the RSAREP function (see Section 2.3.3). If $\mathcal{P}$ is to encode $(x_1, \ldots, x_l)$ into the secret key $\mathcal{V}$ will know for the public key $\prod_{i=1}^{l} g_i^{x_i} x_{l+1}^{v}$ it will end up with, then $\mathcal{V}$ must be unable to modify these l attributes as part of its blinding operations; we say that part of $\mathcal{V}$'s secret key (the first l positions) is *blinding-invariant*. Note that $\mathcal{P}$'s ability to encode $(x_1, \ldots, x_l)$ into $\mathcal{V}$'s secret key does not contradict the requirement that $\mathcal{V}$ is able to blind its certified public key $(p, \mathsf{cert}(p))$, assuming that $\mathcal{V}$ can generate x_{l+1} at random from $\mathbb{Z}_n^*$. The difficulty resides in how to meet all four properties.

In general, $\mathcal{P}$ may encode attributes into $\mathcal{V}$'s secret key in an arbitrary fashion. All that matters is that the blinding-invariant part of $\mathcal{V}$'s secret key can be described by a polynomial-time computable (non-constant) function from the space of secret keys into the space of attributes. For example, suppose that $\widehat{\mathcal{V}}$ can obtain $\mathcal{P}$'s certificate on a public key of the form $\prod_{i=1}^{l} g_i^{\alpha x_i + \beta} x_{l+1}^{v}$, for random blinding factors $\alpha, \beta \in \mathbb{Z}_v$, but not on other forms. Then $\mathcal{P}$ can still encode $l-2$ attributes in an independent manner into $\mathcal{V}$'s secret key, since $x_i^* := (x_i - x_l)(x_{l-1} - x_l)^{-1} \bmod v$ remains unchanged for all $i \in \{1, \ldots, l-2\}$). (More generally, the invariance applies under linear transformations.) For another example, see Proposition 4.3.15. If, on the other hand, $\widehat{\mathcal{V}}$ can obtain $\mathcal{P}$'s certificate on a public key of the form $\prod_{i=1}^{l} g_i^{x_i + \alpha_i} x_{l+1}^{v}$, for random blinding factors $\alpha_1, \ldots, \alpha_l \in \mathbb{Z}_v$, then $\mathcal{P}$ cannot encode anything into $\mathcal{V}$'s secret key.

$\mathcal{P}$ need not necessarily know the attributes it encodes into $\mathcal{V}$'s secret key; knowing a one-way image may suffice, as we will see in Section 5.2.1. This generalization enables $\mathcal{P}$ to "update" previously encoded attributes without knowing their current values. For this reason, in the definition of restrictive blinding we will not be concerned with who determines, knows, or generates the attributes that are to be encoded; of importance is only the existence of a blinding-invariant part in the secret key that $\mathcal{V}$ will end up with.

We are now prepared for a formal definition.

Definition 4.1.1. *A* restrictive blind *certificate scheme is a digital certificate scheme with the following additional properties:*

- *(Blinding of the certified public key) If $\mathcal{V}$ follows the protocol and accepts, then $\mathcal{V}$ obtains a certified key pair $(s, p, \mathsf{cert}(p))$ such that the certified public key $(p, \mathsf{cert}(p))$ is statistically independent from $\widetilde{\mathcal{P}}$'s view in the protocol execution.*

- *(Blinding-invariant part) There exists a non-constant function $\{\mathrm{Inv}_i(\cdot)\}_{i \in V}$ that can be evaluated in polynomial time, such that the following two properties hold if $\mathcal{P}$ follows the protocol:*

- *Let s denote the secret key of the certified key pair obtained by $\overline{\mathcal{V}}$. Then* $\mathrm{Inv}_i(s) = \mathsf{tuple}$*, where* tuple *is the attribute tuple encoded by $\overline{\mathcal{P}}$.*
- *Let $s_1, \ldots, s_{t^*}$ denote the secret keys of any t^* certified key pairs obtained by $\widehat{\mathcal{V}}$ after engaging in $t \geq t^*$ protocol executions, and let* tuple_j *denote the attribute tuple $\overline{\mathcal{P}}$ intended to encode into $\mathcal{V}$'s certified key pair in the j-th protocol execution, for all $j \in \{1, \ldots, t\}$. For all $j^* \in \{1, \ldots, t^*\}$, there exists $j \in \{1, \ldots, t\}$ such that the following two properties hold with overwhelming probability:*
  - $\mathrm{Inv}_i(s_{j^*}) = \mathsf{tuple}_j$.
  - *The multiplicity of* $\mathrm{Inv}_i(s_{j^*})$ *is no greater than the multiplicity of* tuple_j.

We will say that a restrictive blind certificate issuing protocol execution is performed *with respect to* $(x_1, \ldots, x_l)$ if $(x_1, \ldots, x_l)$ is the attribute tuple that $\mathcal{P}$ intends to encode into $\mathcal{V}$'s certified key pair in that particular protocol execution.

The second part of the definition may seem overly complex, but is needed to capture the case where $\mathcal{V}$ consists of a plurality of receivers. Namely, operating under the assumption that an adversary can passively monitor the protocol executions of honest receivers, it must be infeasible for the adversary to benefit from this information by being able to compute a certified key pair for which the secret key encodes an attribute tuple that $\mathcal{P}$ intended to encode only in the secret key of one of the monitored honest receivers.[1] It is not hard to see that the definition captures this scenario, regardless of how protocol executions are interleaved. Note that there is no problem if $\widehat{\mathcal{V}}$ can swap attribute tuples that $\mathcal{P}$ encodes in different protocol executions with $\mathcal{V}$.

Definition 4.1.1 encompasses both public-key certificates and secret-key certificates. Note that restrictive blinding of secret-key certificates is not a special case of Chaum's blind signature paradigm [91, 92, 93, 94, 95, 96, 99, 100]: $\mathcal{P}$'s certificate is not a digital signature on $\mathcal{V}$'s public key but only on $\mathcal{V}$'s secret key, which by definition cannot be blinded.

The notion of restrictive blinding also differs from Chaum's notion of one-show blinding [90, 98]. The latter concerns a property of an issuing protocol in combination with a showing protocol, while restrictive blinding is a property of the issuing protocol only. In particular, restrictive blinding has nothing to do with restricting the number of times a certificate may be shown. One special use of restrictive blinding is to construct practical one-show blind signature schemes (see Section 5.4), but its general applicability is much broader.

Definition 4.1.1 describes the strongest possible case of blinding; not even a CA with unlimited resources can create a correlation between the certified public keys

[1] In a practical situation, session encryption can prevent monitoring of protocol executions, but the security of the session encryption method depends not only on the receiver. Moreover, as a general design principle it is undesirable to make the security of two different building blocks, that serve different goals, depend on each other.

it issues and its views in the issuing protocol. A weaker flavor would be one where linking is merely computationally infeasible, but as explained in Section 1.3.5 this is unsatisfactory.

In practical applications, it will often be desirable that $\mathcal{V}$'s secret key cannot be computed by a party that gets to learn $\mathcal{V}$'s certified public key and also knows $\widetilde{\mathcal{P}}$'s view in the originating issuing protocol. This property is not part of the definition, but holds for all the constructions in this chapter.

Two generic approaches are known to design restrictive blind certificate schemes:

- One can use any "ordinary" blind signature issuing protocol, and have the receiver use a zero-knowledge proof to prove to the issuer that it has properly encoded the attributes into its "challenge" message, before the issuer returns its final response. According to Goldreich, Micali, and Wigderson [191], zero-knowledge proofs exist for all languages in the complexity class NP.

- Techniques from the field of secure multi-party computations can be used, along the lines of Juels, Luby, and Ostrovsky [224]. (See also Damgård [126] and Pfitzmann and Waidner [303].)

Both approaches result in highly impractical protocols. A more efficient approach is to run polynomially many copies of an ordinary blind signature protocol in parallel, and have the signer complete a randomly chosen run of the protocol only when the receiver shows correct formation of the "challenge" messages it submitted in all the other protocol runs. This approach is still far from practical, though, and in fact does not qualify: the attributes cannot be encoded in polynomial time with overwhelming success probability. Note also that the improved issuing protocol of Chaum's ad hoc one-show blind signature scheme [98, 90] does not meet Definition 4.1.1.

The objective of this chapter is to design secure restrictive blind issuing protocols that are truly practical, and that enable the CA to encode polynomially many attributes without affecting the size of certified public keys.

4.2 Practical constructions

In this section we design four practical restrictive blind certificate schemes. The first two of these are based on the DLREP function, the latter two on the RSAREP function. All schemes are for issuing secret-key certificates. We will extensively analyze the schemes in the next section.

In Chapter 5 we will combine the showing protocols of the previous chapter with the issuing protocols designed here. Because the receiver in the issuing protocol will be the prover (signer) in the showing protocol, in the rest of this chapter we denote the CA by $\mathcal{P}_0$ and the receiver by $\mathcal{V}_0$, to avoid confusion with the ($\mathcal{P}$, $\mathcal{V}$) notation used in Chapter 3.

4.2.1 Restrictive blinding based on the DLREP function

DLREP-based scheme I

Let $(I_{\text{DL}}, D_{\text{DL}})$ be any invulnerable instance generator for the DL function, and let (q, g_0) denote the output of I_{DL} on input 1^k. $\mathcal{P}_0$ feeds (q, g_0) to D_{DL} to obtain x_0, and computes $h_0 := g_0^{x_0}$. $\mathcal{P}_0$ then generates $l \geq 1$ random numbers $y_1, \ldots, y_l \in \mathbb{Z}_q$, for some l of its own choice, and computes $g_i := g_0^{y_i}$, for all $i \in \{1, \ldots, l\}$.

The system parameters are (q, g_0). The public key of $\mathcal{P}_0$ is

$$h_0, (g_1, \ldots, g_l),$$

and its secret key is

$$x_0, (y_1, \ldots, y_l).$$

In addition, a correlation-intractable hash function $\mathcal{H}(\cdot) = \{\mathcal{H}_i(\cdot)\}_{i \in \{(q,g_0)\}}$, such that $\mathcal{H}_{q,g_0}(\cdot)$ maps its outputs into $\mathbb{Z}_s$ (for some s superpolynomial in k), is decided on. A concise description of $\mathcal{H}_{q,g_0}(\cdot)$ is published along with the public key. Although not made explicit in the notation, $\mathcal{H}_{q,g_0}(\cdot)$ may (and preferably does) depend also on $\mathcal{P}_0$'s public key and any other information specified before protocol executions take place. (Alternatively, in each application of the hash function all such static information is hashed along.) We will address the issue of selecting $\mathcal{H}(\cdot)$ in Section 4.3.3 when analyzing the security of the scheme.

The restrictive blind issuing protocol ($\mathcal{P}_0$, $\mathcal{V}_0$) is a proof of knowledge such that $\mathcal{V}_0$ obtains a blinded public key, $h' \in G_q$, and a blinded certificate $(c'_0, r'_0) \in \mathbb{Z}_s \times \mathbb{Z}_q$ of $\mathcal{P}_0$ on h'. The pair (c'_0, r'_0) is defined to be a certificate of $\mathcal{P}_0$ on h' if and only if the verification relation

$$c'_0 = \mathcal{H}_{q,g_0}(h', g_0^{r'_0}(h_0 h')^{-c'_0})$$

holds. The secret key of $\mathcal{V}_0$ is a DL-representation, $(x_1, \ldots, x_l, \alpha_1)$, of h' with respect to $(g_1, \ldots, g_l, g_0)$. The numbers $x_1, \ldots, x_l \in \mathbb{Z}_q$ are encoded by $\mathcal{P}_0$ into $\mathcal{V}_0$'s secret key, and in particular are known to $\mathcal{P}_0$; they form the blinding-invariant part of $\mathcal{V}_0$'s secret key. Because $\mathcal{V}_0$ generates α_1 at random, only $\mathcal{V}_0$ knows a secret key corresponding to h' (see Proposition 2.3.3). Moreover, h' is statistically uncorrelated to $(x_1, \ldots, x_l)$, regardless of the distribution of $(x_1, \ldots, x_l)$.

With h denoting $\prod_{i=1}^{l} g_i^{x_i}$, an execution of the certificate issuing protocol with respect to $(x_1, \ldots, x_l)$ is defined as follows:

Step 1. $\mathcal{P}_0$ generates a random number $w_0 \in \mathbb{Z}_q$, and sends $a_0 := g_0^{w_0}$ to $\mathcal{V}_0$.

Step 2. $\mathcal{V}_0$ generates three random numbers $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{Z}_q$. $\mathcal{V}_0$ computes $h' := hg_0^{\alpha_1}$, $c'_0 := \mathcal{H}_{q,g_0}(h', g_0^{\alpha_2}(h_0 h)^{\alpha_3} a_0)$, and sends $c_0 := c'_0 + \alpha_3 \bmod q$ to $\mathcal{P}_0$.

Step 3. $\mathcal{P}_0$ sends $r_0 := c_0(x_0 + \sum_{i=1}^{l} x_i y_i) + w_0 \bmod q$ to $\mathcal{V}_0$.

$\mathcal{V}_0$ accepts if and only if $g_0^{r_0}(h_0 h)^{-c_0} = a_0$. If this verification holds, $\mathcal{V}_0$ computes $r_0' := r_0 + \alpha_2 + c_0'\alpha_1 \bmod q$.

We restrict $\mathcal{P}_0$ in the following manner. It may perform protocol executions with respect to the same $(x_1, \ldots, x_l)$ in parallel, but must perform executions that involve different attribute tuples sequentially. (The reason for this restriction will be clarified in Section 4.3.3. In Section 4.4 we will show how to get around the restriction.) The resulting scheme is depicted in Figure 4.1.

When forming c_0' in Step 2, $\mathcal{V}_0$ may hash along additional information, such as a public key to be used for session encryption in a showing protocol or one or more initial witnesses for the showing protocol. The advantages of including the latter will become clear in Section 5.4.

DLREP-based scheme II

The following variation of DLREP-based scheme I is somewhat less efficient, but as Proposition 4.3.7 will show admits a better proof of unforgeability in the random oracle model. The required modifications are minimal, and so we only describe these:

- $\mathcal{P}_0$ generates an additional random number $f \in G_q$, which it publishes along with the other public key data. It also generates an additional random number $t \in \{0, 1\}$, serving as additional secret key information to $\mathcal{P}_0$, and forms h_0 according to $h_0 := g_0^{x_0} f^t$. No further changes are needed in the key set-up.

- A certificate of $\mathcal{P}_0$ on h' is redefined to be a triple, $(c_0', r_0', r_1') \in \mathbb{Z}_s \times \mathbb{Z}_q \times \mathbb{Z}_q$ such that

  $$c_0' = \mathcal{H}_{q,g_0}(h', g_0^{r_0'} f^{r_1'} (h_0 h')^{-c_0'})$$

  The definition of a key pair for $\mathcal{V}_0$ is not changed, nor is that of the blinding-invariant part.

- In Step 1 of the certificate issuing protocol, $\mathcal{P}_0$ generates an additional random number $w_1 \in \mathbb{Z}_q$, and forms a_0 according to $a_0 := g_0^{w_0} f^{w_1}$. In Step 2 of the protocol, $\mathcal{V}_0$ generates an additional random number $\alpha_4 \in \mathbb{Z}_q$, and multiplies f^{α_4} into the second argument to $\mathcal{H}_{q,g_0}(\cdot)$ when computing c_0'. In Step 3 of the protocol, $\mathcal{P}_0$ computes an additional response, r_1, according to $r_1 := c_0 t + w_1 \bmod q$, and sends this along to $\mathcal{V}_0$. Finally, $\mathcal{V}_0$ accepts if and only if $g_0^{r_0} f^{r_1} (h_0 h)^{-c_0} = a_0$, and in addition blinds r_1 to $r_1' := r_1 + \alpha_4 \bmod q$.

The requirement that $\mathcal{P}_0$ may not interleave protocol executions with respect to different attribute tuples still applies.

The resulting scheme is depicted in Figure 4.2.

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(q, g_0) := I_{\text{DL}}(1^k)$	
	KEY SET-UP	
$x_0 := D_{\text{DL}}(q, g_0)$		
$y_1, \ldots, y_l \in_{\mathcal{R}} \mathbb{Z}_q$		
Secret key: $x_0, (y_1, \ldots, y_l)$		
$h_0 := g_0^{x_0}$		
$g_i := g_0^{y_i}\ \forall i \in \{1, \ldots, l\}$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{q,g_0}(\cdot)$	
	PROTOCOL	
$w_0 \in_{\mathcal{R}} \mathbb{Z}_q$		
$a_0 := g_0^{w_0}$		
	$\xrightarrow{a_0}$	
		$\alpha_1, \alpha_2, \alpha_3 \in_{\mathcal{R}} \mathbb{Z}_q$
		$h' := hg_0^{\alpha_1}$
		$c_0' := \mathcal{H}_{q,g_0}(h', g_0^{\alpha_2}(h_0 h)^{\alpha_3} a_0)$
		$c_0 := c_0' + \alpha_3 \bmod q$
	$\xleftarrow{c_0}$	
$r_0 := c_0(x_0 + \sum_{i=1}^{l} x_i y_i) + w_0 \bmod q$		
	$\xrightarrow{r_0}$	
		$g_0^{r_0}(h_0 h)^{-c_0} \stackrel{?}{=} a_0$
		$r_0' := r_0 + \alpha_2 + c_0' \alpha_1 \bmod q$

Figure 4.1: DLREP-based scheme I.

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(q, g_0) := I_{\text{DL}}(1^k)$	
	KEY SET-UP	
$x_0 := D_{\text{DL}}(q, g_0)$		
$t \in_{\mathcal{R}} \{0, 1\}$		
$y_1, \ldots, y_l \in_{\mathcal{R}} \mathbb{Z}_q$		
Secret key: $(x_0, t), (y_1, \ldots, y_l)$		
$f \in_{\mathcal{R}} G_q$		
$h_0 := g_0^{x_0} f^t$		
$g_i := g_0^{y_i}\ \forall i \in \{1, \ldots, l\}$		
Public key:	$(f, h_0), (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{q,g_0}(\cdot)$	
	PROTOCOL	
$w_0, w_1 \in_{\mathcal{R}} \mathbb{Z}_q$		
$a_0 := g_0^{w_0} f^{w_1}$		
	$\xrightarrow{a_0}$	
		$\alpha_1, \alpha_2, \alpha_3, \alpha_4 \in_{\mathcal{R}} \mathbb{Z}_q$
		$h' := h g_0^{\alpha_1}$
		$c_0' := \mathcal{H}_{q,g_0}(h', g_0^{\alpha_2}(h_0 h)^{\alpha_3} f^{\alpha_4} a_0)$
		$c_0 := c_0' + \alpha_3 \bmod q$
	$\xleftarrow{c_0}$	
$r_0 := c_0(x_0 + \sum_{i=1}^{l} x_i y_i) + w_0 \bmod q$		
$r_1 := c_0 t + w_1 \bmod q$		
	$\xrightarrow{r_0, r_1}$	
		$g_0^{r_0} f^{r_1} (h_0 h)^{-c_0} \stackrel{?}{=} a_0$
		$r_0' := r_0 + \alpha_2 + c_0' \alpha_1 \bmod q$
		$r_1' := r_1 + \alpha_4 \bmod q$

Figure 4.2: DLREP-based scheme II.

4.2.2 Restrictive blinding based on the RSAREP function

RSAREP-based scheme I

Let $(I_{\text{RSA}}, D_{\text{RSA}})$ be any invulnerable instance generator for the RSA function, and let (n, v) denote the output of I_{RSA} on input 1^k. We assume that I_{RSA} outputs the prime factorization (p, q) of n as "side information" for $\mathcal{P}_0$. $\mathcal{P}_0$ feeds (n, v) to D_{RSA} to obtain x_0, and computes $h_0 := x_0^v$. $\mathcal{P}_0$ then generates $l \geq 1$ random numbers $g_1, \ldots, g_l \in \mathbb{Z}_n^*$.

The system parameters are (n, v). The public key of $\mathcal{P}_0$ is

$$h_0, (g_1, \ldots, g_l),$$

and its secret key is the prime factorization of n. In addition, a one-way hash function $\mathcal{H}(\cdot) = \{\mathcal{H}_i(\cdot)\}_{i \in \{(n,v)\}}$, such that $\mathcal{H}_{n,v}(\cdot)$ maps its outputs into $\mathbb{Z}_s$ (for some s superpolynomial in k), is decided on. A concise description of $\mathcal{H}_{n,v}(\cdot)$ is published along with the public key. Although not made explicit in the notation, its specification may depend on $\mathcal{P}_0$'s public key and any other information specified before protocol executions take place. We will address the issue of selecting $\mathcal{H}(\cdot)$ in Section 4.3.3.

Our restrictive blind issuing protocol $(\mathcal{P}_0, \mathcal{V}_0)$ is a proof of knowledge such that $\mathcal{V}_0$ obtains a blinded public key, $h' \in \mathbb{Z}_n^*$, and a blinded certificate $(c_0', r_0') \in \mathbb{Z}_s \times \mathbb{Z}_n^*$ of $\mathcal{P}_0$ on h'. The pair (c_0', r_0') is defined to be a certificate of $\mathcal{P}_0$ on h' if and only if the verification relation

$$c_0' = \mathcal{H}_{n,v}(h', (r_0')^v (h_0 h')^{-c_0'})$$

holds. The secret key of $\mathcal{V}_0$ is an RSA-representation, $(x_1, \ldots, x_l, \alpha_1)$, of h' with respect to $(g_1, \ldots, g_l, v)$. The numbers $x_1, \ldots, x_l \in \mathbb{Z}_v$ are encoded by $\mathcal{P}_0$ into $\mathcal{V}_0$'s secret key; they form the blinding-invariant part of $\mathcal{V}_0$'s secret key. Because $\mathcal{V}_0$ generates α_1 at random, h' is uncorrelated to $(x_1, \ldots, x_l)$, regardless of the distribution of $(x_1, \ldots, x_l)$.

With h denoting $\prod_{i=1}^{l} g_i^{x_i}$, an execution of the certificate issuing protocol with respect to $(x_1, \ldots, x_l)$ is defined as follows:

Step 1. $\mathcal{P}_0$ generates a random number $a_0 \in \mathbb{Z}_n^*$, and sends it to $\mathcal{V}_0$.

Step 2. $\mathcal{V}_0$ generates two random numbers $\alpha_1, \alpha_2 \in \mathbb{Z}_n^*$ and a random number $\alpha_3 \in \mathbb{Z}_v$. $\mathcal{V}_0$ computes $h' := h\alpha_1^v$, $c_0' := \mathcal{H}(h', \alpha_2^v (h_0 h)^{\alpha_3} a_0)$, and sends $c_0 := c_0' + \alpha_3 \bmod v$ to $\mathcal{P}_0$.

Step 3. $\mathcal{P}_0$ sends $r_0 := ((h_0 h)^{c_0} a_0)^{1/v}$ to $\mathcal{V}_0$. (Note that $\mathcal{P}_0$ can compute v-th roots of arbitrary numbers in $\mathbb{Z}_n^*$, because it knows the prime factorization of n.)

$\mathcal{V}_0$ accepts if and only if $r_0^v (h_0 h)^{-c_0} = a_0$. If this verification holds, $\mathcal{V}_0$ computes

$$r_0' := r_0 \alpha_2 \alpha_1^{c_0'} (h_0 h)^{(c_0' + \alpha_3) \operatorname{div} v}.$$

As with both DLREP-based schemes, $\mathcal{P}_0$ may perform protocol executions with respect to the same $(x_1, \ldots, x_l)$ in parallel, but may not interleave executions that involve different attribute tuples. (In Section 4.4 we will show how to get around this restriction.) The resulting scheme is depicted in Figure 4.3.

When forming c'_0 in Step 2, $\mathcal{V}_0$ may hash along additional information, such as a public key for session encryption or one or more initial witnesses for a subsequent showing protocol. Inclusion of the latter will be pursued further in Section 5.4.

RSAREP-based scheme II

The following variation of RSAREP-based scheme I is somewhat less efficient, but admits a better proof of unforgeability in the random oracle model. The modifications to RSAREP-based scheme I are the following:

- $\mathcal{P}_0$ generates an additional random number $f \in \mathbb{Z}_n^*$, which it publishes along with its other public key data. No further changes are needed in the key set-up.
- A certificate of $\mathcal{P}_0$ on h' is redefined to be a triple, $(c'_0, r'_0, r'_1) \in \mathbb{Z}_s \times \mathbb{Z}_n^* \times \mathbb{Z}_v$ such that
$$c'_0 = \mathcal{H}_{n,v}(h', (r'_0)^v f^{r'_1}(h_0 h')^{-c'_0})$$
The definition of a key pair for $\mathcal{V}_0$ is not changed, nor is that of the blinding-invariant part.
- In Step 2 of the protocol, $\mathcal{V}_0$ generates an additional random number $\alpha_4 \in \mathbb{Z}_v$, and multiplies f^{α_4} into the second argument to $\mathcal{H}_{n,v}(\cdot)$ when computing c'_0. In Step 3 of the protocol, $\mathcal{P}_0$ generates a random number $r_1 \in \mathbb{Z}_v$, computes r_0 according to $r_0 := ((h_0 h)^{c_0} a_0 / f^{r_1})^{1/v}$, and sends r_0 along to $\mathcal{V}_0$. Finally, $\mathcal{V}_0$ accepts if and only if $r_0^v f^{r_1}(h_0 h)^{-c_0} = a_0$, and computes
$$r'_0 := r_0 \alpha_2 \alpha_1^{c'_0} (h_0 h)^{(c'_0 + \alpha_3) \operatorname{div} v} f^{(r_1 + \alpha_4) \operatorname{div} v}$$
and $r'_1 := r_1 + \alpha_4 \bmod v$.

The requirement that $\mathcal{P}_0$ may not interleave protocol executions with respect to different attribute tuples still applies.

The resulting scheme is depicted in Figure 4.4.

4.2.3 Comparison

The constructions based on the DLREP function and on the RSAREP function follow exactly the same design principle. This may not be readily clear from the descriptions, because $\mathcal{P}_0$ in the RSAREP-based variants makes use of trapdoor information, which is not available in the DLREP-based variants. To appreciate the underlying design principle, observe that $\mathcal{P}_0$ need not make use of trapdoor information in the

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(n, v) := I_{\text{RSA}}(1^k)$	
	KEY SET-UP	
Secret key: factorization of n		
$h_0, g_1, \ldots, g_l \in_{\mathcal{R}} \mathbb{Z}_n^*$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{n,v}(\cdot)$	
	PROTOCOL	
$a_0 \in_{\mathcal{R}} \mathbb{Z}_n^*$		
	$\xrightarrow{a_0}$	
		$\alpha_1, \alpha_2 \in_{\mathcal{R}} \mathbb{Z}_n^*$
		$\alpha_3 \in_{\mathcal{R}} \mathbb{Z}_v$
		$h' := h\alpha_1^v$
		$c_0' := \mathcal{H}_{n,v}(h', \alpha_2^v (h_0 h)^{\alpha_3} a_0)$
		$c_0 := c_0' + \alpha_3 \bmod v$
	$\xleftarrow{c_0}$	
$r_0 := ((h_0 h)^{c_0} a_0)^{1/v}$		
	$\xrightarrow{r_0}$	
		$r_0^v (h_0 h)^{-c_0} \stackrel{?}{=} a_0$
		$r_0' := r_0 \alpha_2 \alpha_1^{c_0'} (h_0 h)^{(c_0' + \alpha_3) \operatorname{div} v}$

Figure 4.3: RSAREP-based scheme I.

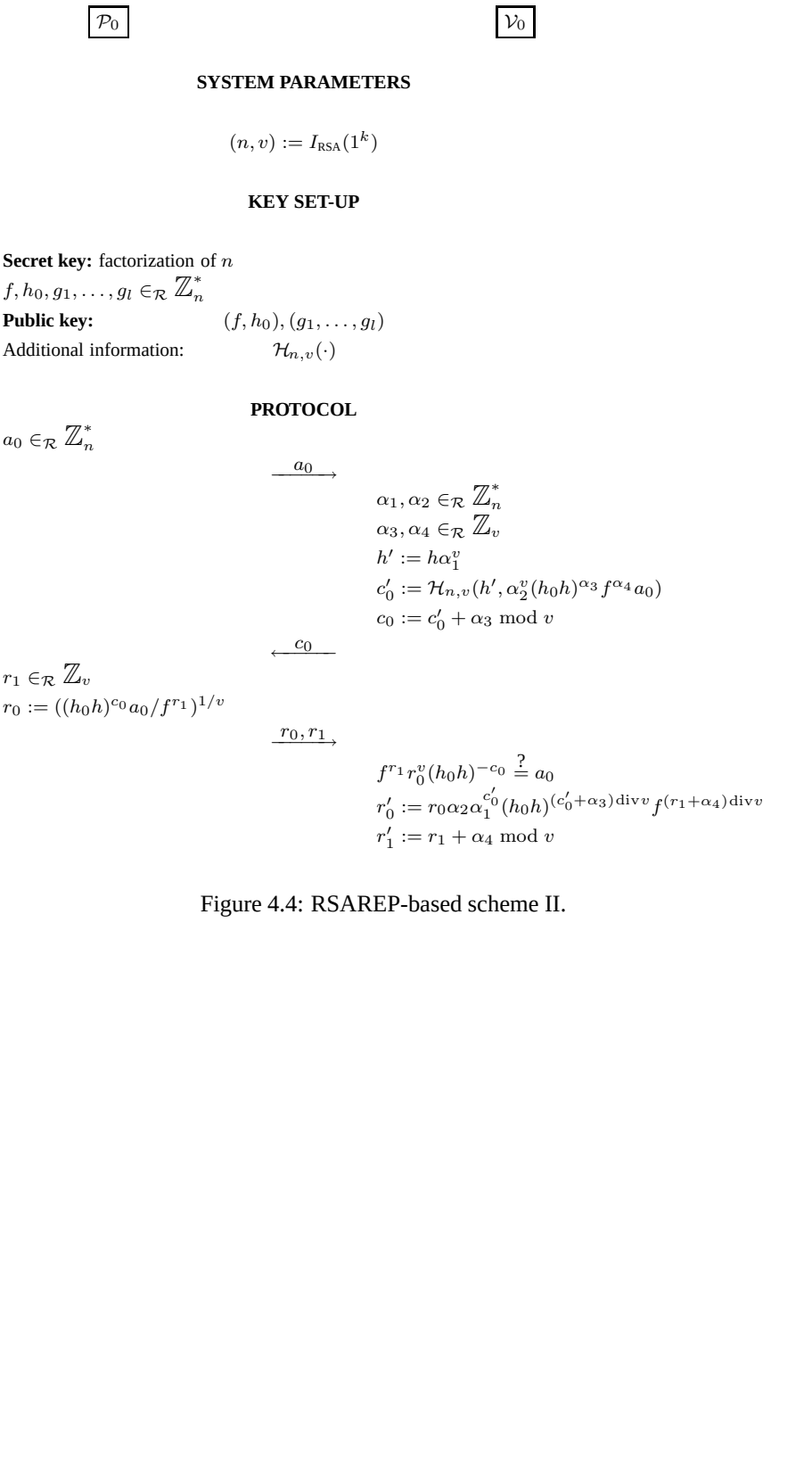

Figure 4.4: RSAREP-based scheme II.

RSAREP-based schemes. In RSAREP-based scheme I, we hereto make the following modifications:

- Instead of generating $h_0, g_1, \ldots, g_l$ at random, $\mathcal{P}_0$ generates $l+1$ random numbers $x_0, y_1, \ldots, y_l$ from $\mathbb{Z}_n^*$, and computes $h_0 := x_0^v$ and $g_i := y_i^v$, for all $i \in \{1, \ldots, l\}$. (More generally, $\mathcal{P}_0$ may set $x_0 := D_{\text{RSA}}(n, v)$.)
- In Step 1 of the issuing protocol, $\mathcal{P}_0$ generates a_0 according to $a_0 := w_0^v$, for a random $w_0 \in \mathbb{Z}_n^*$.
- In Step 3 of the issuing protocol, $\mathcal{P}_0$ computes r_0 as follows:

$$r_0 := (x_0 \prod_{i=1}^{l} y_i^{x_i})^{c_0} w_0.$$

The resulting scheme is depicted in Figure 4.5. Similar modifications can be made to RSAREP-based scheme II. With these modifications, it is easily seen that the DLREP-based and the RSAREP-based schemes are all based on the same design principle:

> $\mathcal{P}_0$ interactively issues a signed proof of knowledge of a secret key corresponding to the joint public key $h_0 h$, using one of the proofs of knowledge described in Sections 2.4.3 and 2.4.4. $\mathcal{V}_0$ blinds not only a_0 and $\mathcal{P}_0$'s response(s), but also h.

In all four schemes, $h_0 h$ or h may be thought of as the auxiliary common input m^* in Definition 2.5.1. Note that for both DLREP-based scheme II and RSAREP-based scheme II the issuing protocols are provably witness-hiding. (In particular, even after $\overline{\mathcal{P}_0}$ has performed polynomially many protocol executions, arbitrarily interleaved and possibly with respect to all valid attribute tuples, its secret key provably cannot leak.)

Not using the trapdoor information in the RSAREP-based schemes has several advantages:

- Multiple provers can all operate with respect to the same (n, v), generated by a trusted party or by means of a secure multi-party protocol (see Boneh and Franklin [39] and Poupard and Stern [309]).
- The binary sizes of v and c may be much smaller than the binary sizes of the prime factors of n. This reduces $\mathcal{P}_0$'s computational burden.
- $\mathcal{P}_0$ can be split into many sub-provers that all hold a share of the public key, and that must all contribute to issue a certified key pair to $\mathcal{V}_0$. Using RSAREP-based scheme I, for instance, the i-th sub-prover could hold $h_{0i} := x_{0i}^v$, with $\prod_{i=1}^{l} h_{0i} = h_0$, and could be in charge of generating y_i. The contribution of the i-th sub-prover to the issuing protocol would be an initial witness $a_{0i} :=$

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(n, v) := I_{\text{RSA}}(1^k)$	
	KEY SET-UP	
$x_0 := D_{\text{RSA}}(n, v)$		
$y_1, \ldots, y_l \in_{\mathcal{R}} \mathbb{Z}_n^*$		
Secret key: $x_0, (y_1, \ldots, y_l)$		
$h_0 := x_0^v$		
$g_i := y_i^v \ \forall i \in \{1, \ldots, l\}$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{n,v}(\cdot)$	
	PROTOCOL	
$w_0 \in_{\mathcal{R}} \mathbb{Z}_n^*$		
$a_0 := w_0^v$		
	$\xrightarrow{a_0}$	
		$\alpha_1, \alpha_2 \in_{\mathcal{R}} \mathbb{Z}_n^*$
		$\alpha_3 \in_{\mathcal{R}} \mathbb{Z}_v$
		$h' := h\alpha_1^v$
		$c_0' := \mathcal{H}_{n,v}(h', \alpha_2^v (h_0 h)^{\alpha_3} a_0)$
		$c_0 := c_0' + \alpha_3 \bmod v$
	$\xleftarrow{c_0}$	
$r_0 := (x_0 \prod_{i=1}^{l} y_i^{x_i})^{c_0} w_0$		
	$\xrightarrow{r_0}$	
		$r_0^v (h_0 h)^{-c_0} \stackrel{?}{=} a_0$
		$r_0' := r_0 \alpha_2 \alpha_1^{c_0'} (h_0 h)^{(c_0' + \alpha_3) \operatorname{div} v}$

Figure 4.5: RSAREP-based scheme I without use of trapdoor information.

w_{0i}^v and a response $r_{0i} := (x_{0i} y_i^{x_i})^{c_0} w_{0i}$; the product of all the individual witnesses is the initial witness expected by $\mathcal{V}_0$, and likewise for the responses.

The technique of sharing $\mathcal{P}_0$'s secret key can also be applied to the DLREP-based issuing protocols (and the other protocols that will be described later in this chapter). It is even possible to extend the technique to provide for arbitrary secret sharing, requiring one of several predetermined subsets to cooperate in order to perform the role of $\mathcal{P}_0$. For relevant secret-sharing techniques, see Pedersen [299], Cerecedo, Matsumoto, and Imai [84], Gennaro, Jarecki, Krawczyk, and Rabin [184], and Takaragi, Miyazaki, and Takahashi [369]. Also, the number of entities that share $\mathcal{P}_0$'s secret key could be increased so that multiple entities are needed to approve each attribute. In a practical implementation, the sub-provers could take the form of tamper-resistant computing devices stored in independently guarded locations. This not only provides optimal protection against (insider and outsider) theft and extortion of $\mathcal{P}_0$'s secret key, but it can also ensure that different device operators must approve the same attributes that are to be encoded into a certified key pair. [2]

Using the trapdoor information in the RSAREP-based schemes also has a couple of advantages:

- It avoids the exponentiation in Step 1 of the protocol.
- $\mathcal{P}_0$ does not need to remember or reconstruct in Step 3 a secret number that it generated in Step 1, which is an advantage when implementing the protocol. ($\mathcal{P}_0$ still needs to access its secret key, of course.) The protocol can even be turned into a two-move protocol by having $\mathcal{V}_0$ form a_0 by feeding at least an identifier for $\mathcal{V}_0$ and a nonce into a sufficiently strong one-way function. ($\mathcal{V}_0$ must send the nonce along with its challenge to $\mathcal{P}_0$, so that $\mathcal{P}_0$ can check its freshness.)
- $\mathcal{P}_0$ can perform the issuing protocol without knowing $(x_1, \ldots, x_l)$; it merely needs to know h. This property enables $\mathcal{P}_0$ to recertify a previously certified public key, without knowing its blinding-invariant part. During the process, $\mathcal{P}_0$ can even update one or more of the x_i values. (Details will be provided in Section 5.2.1.)

In the RSAREP-based schemes, $\mathcal{V}_0$ cannot verify by itself that v is co-prime to $\varphi(n)$. However, if the prime v is not co-prime to $\varphi(n)$, then $\widetilde{\mathcal{P}}_0$ cannot respond to $\mathcal{V}$'s challenge c_0 with probability at least $1 - 1/v$. In other words, $\overline{\mathcal{V}}_0$ becomes convinced with overwhelming probability of the proper formation of (n, v) by engaging in a single execution of the certificate issuing protocol.

[2] Issuer fraud is a serious threat, as witnessed for instance by the 250 employees of the Department of Motor Vehicles of California who in 1998 were found to have issued over 25 000 genuine-looking but fraudulent licenses in a two-year period.

In Step 2 of all four certificate schemes, $\mathcal{V}_0$ can perform all the required exponentiations in a preprocessing stage; its real-time computational burden in each protocol amounts to one modular multiplication and one application of the hash function. This makes the schemes highly practical.

The main advantage of the DLREP-based variants over the RSAREP-based variants is that the computation of $\mathcal{P}_0$'s response(s) does not involve any exponentiations. In highly demanding applications, this enables the CA to serve more receivers using cheaper equipment, especially when using an elliptic curve implementation with short system parameters.

4.3 Analysis

In this section we analyze the certificate schemes of the previous section. We will prove that all four schemes are restrictive blind certificate schemes, under plausible cryptographic assumptions.

To avoid unnecessary duplication of security statements and proof reductions, a detailed analysis is provided only of RSAREP-based scheme I. The analysis of the other three schemes is highly similar, and so for these we merely point out the differences.

Throughout this section it is assumed that the system parameters, (q, g_0) and (n, v), respectively, are properly formed.

4.3.1 Completeness

The statements in this section hold for any choice of $\mathcal{H}(\cdot)$.

Proposition 4.3.1. *When interacting with $\overline{\mathcal{P}}_0$, $\overline{\mathcal{V}}_0$ in RSAREP-based scheme I accepts.*

Proof. This follows immediately from the manner in which $\mathcal{P}_0$ computes r_0 in Step 3 and the verification relation applied by $\mathcal{V}_0$. □

Proposition 4.3.2. *For any $\widetilde{\mathcal{P}}_0$, if $\overline{\mathcal{V}}_0$ in RSAREP-based scheme I accepts, then*

$$(x_1, \ldots, x_l, \alpha_1),\ h',\ (c'_0, r'_0)$$

is a certified key pair.

Proof. Clearly, $(x_1, \ldots, x_l, \alpha_1)$ is an RSA-representation of h' with respect to the tuple $(g_1, \ldots, g_l, v)$. To show that (c'_0, r'_0) is a certificate of $\mathcal{P}_0$ on h', note that $\overline{\mathcal{V}}_0$ in Step 2 of the issuing protocol computes $c'_0 := \mathcal{H}_{n,v}(h', \alpha_2^v (h_0 h)^{\alpha_3} a_0)$. It therefore

suffices to prove that $(r_0')^v(h_0h')^{-c_0'} = \alpha_2^v(h_0h)^{\alpha_3}a_0$ for the assignments made by $\overline{\mathcal{V}}_0$. This can be seen as follows:

$$\begin{aligned}
(r_0')^v(h_0h')^{-c_0'} &= (r_0\alpha_2\alpha_1^{c_0'}(h_0h)^{(c_0'+\alpha_3)\bmod v})^v(h_0h\alpha_1^v)^{-c_0'} \\
&= (r_0\alpha_2(h_0h)^{(c_0'+\alpha_3)\bmod v})^v(h_0h)^{-c_0'} \\
&= r_0^v\alpha_2^v(h_0h)^{v\,((c_0'+\alpha_3)\bmod v)}(h_0h)^{-c_0'} \\
&\overset{(\star)}{=} ((h_0h)^{c_0}a_0)\,\alpha_2^v(h_0h)^{v((c_0'+\alpha_3)\bmod v)}(h_0h)^{-c_0'} \\
&= (h_0h)^{(c_0'+\alpha_3 \bmod v)+v\,((c_0'+\alpha_3)\bmod v)}a_0\alpha_2^v(h_0h)^{-c_0'} \\
&= (h_0h)^{c_0'+\alpha_3}\alpha_2^v(h_0h)^{-c_0'}a_0 \\
&= \alpha_2^v(h_0h)^{\alpha_3}a_0.
\end{aligned}$$

The substitution $(\star)$ is allowed because $\overline{\mathcal{V}}_0$ accepts only if $r_0^v(h_0h)^{-c_0} = a_0$. □

In a like manner, the direct analogues of these two propositions can be proved for the other three certificate schemes in Section 4.2.

4.3.2 Privacy for the receiver

The statements in this section address the protocol ($\widetilde{\mathcal{P}}_0$, $\overline{\mathcal{V}}_0$), and hold for any choice of $\mathcal{H}(\cdot)$.

Lemma 4.3.3. *In RSAREP-based scheme I, for any properly formed system parameters, any certified public key, any $(x_1, \ldots, x_l)$, and any possible view of $\widetilde{\mathcal{P}}_0$ in an execution of the issuing protocol with respect to $(x_1, \ldots, x_l)$ in which $\overline{\mathcal{V}}_0$ accepts, there is exactly one set of random choices that $\overline{\mathcal{V}}_0$ could have made in that execution of the issuing protocol such that $\overline{\mathcal{V}}_0$ would end up with a certified key pair containing that particular certified public key.*

Proof. Consider any tuple $(x_1, \ldots, x_l)$ and any certified public key h', (c_0', r_0'). With h denoting $\prod_{i=1}^{l} g_i^{x_i}$, the response r_0 of $\widetilde{\mathcal{P}}_0$ is such that $r_0^v(h_0h)^{-c_0} = a_0$, since $\overline{\mathcal{V}}_0$ accepts. Define the following two sets:

$$\begin{aligned}
\text{Views}\,(\widetilde{\mathcal{P}}_0) &= \{(a_0, c_0, r_0) \mid a_0, r_0 \in \mathbb{Z}_n^* \text{ and } c_0 \in \mathbb{Z}_v \text{ such that} \\
&\qquad r_0^v(h_0h)^{-c_0} = a_0\} \\
\text{Choices}\,(\mathcal{V}_0) &= \{(\alpha_1, \alpha_2, \alpha_3) \mid \alpha_1, \alpha_2 \in \mathbb{Z}_n^* \text{ and } \alpha_3 \in \mathbb{Z}_v\}.
\end{aligned}$$

We will show that for all $\widetilde{\mathcal{P}}_0$-view $\in$ Views $(\widetilde{\mathcal{P}}_0)$ exactly one triple $(\alpha_1, \alpha_2, \alpha_3) \in$ Choices $(\mathcal{V}_0)$ exists such that $\widetilde{\mathcal{P}}_0$-view corresponds to an execution of the issuing protocol in which $\overline{\mathcal{V}}_0$ receives the certified public key $(h', (c_0', r_0'))$.

Suppose that $\widetilde{\mathcal{P}}_0$-view corresponds to the issuing of $h', (c'_0, r'_0)$. We determine the numbers $\alpha_1, \alpha_2, \alpha_3$ that must have been chosen by $\overline{\mathcal{V}}_0$. First, α_1 is determined from h, h' as

$$\alpha_1 := (h'\, h^{-1})^{1/v}.$$

Note that α_1 exists and is uniquely defined, since v is co-prime to $\varphi(n)$. Next, α_3 is determined from c_0, c'_0 according to

$$\alpha_3 := c_0 - c'_0 \bmod v.$$

Finally, the choices for α_1 and α_3, together with r_0, r'_0 and c'_0, uniquely determine α_2 as

$$\alpha_2 := r'_0 (r_0 \alpha_1^{c'_0} (h_0 h)^{(c'_0+\alpha_3) \operatorname{div} v})^{-1}.$$

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

$$c'_0 := \mathcal{H}_{n,v}(h',\ \alpha_2^v (h_0 h)^{\alpha_3} a_0)$$

that must have been made by $\overline{\mathcal{V}}_0$. To prove that this assignment holds as well, note that

$$c'_0 = \mathcal{H}_{n,v}(h', (r'_0)^v (h_0 h')^{-c'_0})$$

by definition of a certified public key. Therefore the proof is complete if

$$(r'_0)^v (h_0 h')^{-c'_0} = \alpha_2^v (h_0 h)^{\alpha_3} a_0$$

for the choices for α_1, α_2, and α_3 made above. This can be derived exactly as in the proof of Proposition 4.3.2, considering that the substitution $(\star)$ is allowed here because $\widetilde{\mathcal{P}}_0$-view $\in$ Views $(\widetilde{\mathcal{P}}_0)$. □

Lemma 4.3.3 does not necessarily hold in the case of improperly formed system parameters. In particular, if v is not co-prime to $\varphi(n)$ then a substantial part of the views of $\widetilde{\mathcal{P}}_0$ cannot be matched with a substantial part of the certified public keys. This is not a problem, though, as we saw in Section 4.2.3.

Proposition 4.3.4. *For any properly formed system parameters in RSAREP-based scheme I, if $\mathcal{V}_0$ follows the issuing protocol and accepts, then it obtains a certified key pair comprising a perfectly blinded certified public key, regardless of the behavior of $\widetilde{\mathcal{P}}_0$.*

Proof. This is an immediate consequence of Lemma 4.3.3 and the fact that $\overline{\mathcal{V}}_0$ generates its triples $(\alpha_1, \alpha_2, \alpha_3)$ at random from Choices $(\mathcal{V}_0)$. □

The same result can be proved for the other three certificate schemes described in Section 4.2.

In Chapter 5 we will make the connection with the showing protocols in Chapter 3 and show that that the above privacy result holds even when $\overline{\mathcal{V}}_0$ selectively discloses any property of the encoded attributes. That is, any certificate that $\overline{\mathcal{V}}_0$ shows in the showing protocol execution could have originated (with uniform probability) from any of the issuing protocol executions in which $\widetilde{\mathcal{P}}_0$ encoded attributes that satisfy the formula disclosed by $\overline{\mathcal{V}}_0$.

4.3.3 Security for the Certificate Authority

In this section we address the protocol $(\overline{\mathcal{P}}_0, \widehat{\mathcal{V}}_0)$, by analyzing the properties of unforgeability and restrictive blinding.

Unforgeability

We study the unforgeability of RSAREP-based scheme I in the strongest possible attack model. All our unforgeability results hold even if $\widehat{\mathcal{V}}_0$ can engage in polynomially many executions of the issuing protocol, can arbitrarily interleave protocol executions, and may select an arbitrary attribute tuple $(x_1, \ldots, x_l)$ at the start of each new protocol execution.

The following lemma holds for any choice of $\mathcal{H}(\cdot)$.

Lemma 4.3.5. *If the Guillou-Quisquater proof of knowledge with $s := v$ is witness-hiding, then $\widehat{\mathcal{V}}_0$ in RSAREP-based scheme I cannot output with non-negligible success probability a non-trivial RSA-representation of 1 with respect to $(g_1, \ldots, g_l, v)$.*

Proof. Suppose that $\widehat{\mathcal{V}}_0$, after engaging in t executions of the issuing protocol, outputs a non-trivial RSA-representation of 1 with respect to $(g_1, \ldots, g_l, v)$, with non-negligible probability ϵ. We construct a polynomial-time interactive algorithm $\widehat{\mathcal{V}}$ for extracting the witness of $\overline{\mathcal{P}}$ in the Guillou-Quisquater proof of knowledge, as follows.

Let (n, v) denote the system parameters in the Guillou-Quisquater proof of knowledge, and h_{GQ} the public key of $\overline{\mathcal{P}}$. $\widehat{\mathcal{V}}$ simulates $\overline{\mathcal{P}}_0$ with the help of the protocol executions of $\overline{\mathcal{P}}$, by performing the following steps:

Step A. (Simulate the key set-up for $\overline{\mathcal{P}}_0$.) Select a random index $j \in \{1, \ldots, l\}$ and $l + 1$ random numbers $x_0, y_1, \ldots, y_l \in \mathbb{Z}_n^*$. Set $h_0 := x_0^v$, $g_i := y_i^v$, for all $i \in \{1, \ldots, l\} \setminus \{j\}$, and $g_j := h_{\text{GQ}} y_j^v$. The simulated public key of $\overline{\mathcal{P}}_0$ is $h_0, (g_1, \ldots, g_l)$.

Step B. (Simulate $\overline{\mathcal{P}}_0$ in issuing protocol executions with respect to $(x_1, \ldots, x_l)$.)

Step 1. Receive a from $\overline{\mathcal{P}}$. Generate a random number $\alpha \in \mathbb{Z}_n^*$ and pass $a_0 := a^{x_j} \alpha^v \bmod n$ on to $\widehat{\mathcal{V}}_0$.

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

Step 3. Receive r from $\overline{\mathcal{P}}$, and pass

$$r_0 := r^{x_j}(x_0 \prod_{i=1}^{l} y_i^{x_i})^{c_0}\alpha$$

on to $\widehat{\mathcal{V}}_0$.

Repeat this simulation until t executions of the issuing protocol have been performed.

Step C. Check if $\widehat{\mathcal{V}}_0$ has output a non-trivial RSA-representation, $(u_1, \ldots, u_l, u_{l+1})$, of 1. If not, then halt.

Step D. If $u_j = 0 \bmod v$, then halt.

Step E. Compute integers $e, f \in \mathbb{Z}$ such that $eu_j + fv = 1$, using the extended Euclidean algorithm. (This can always be done, because v is a prime.) Compute

$$h_{\text{GQ}}^{f}(\prod_{i=1}^{l} y_i^{u_i} u_{l+1})^{-e}$$

and output the result.

It is easy to see that the public key in Step A is generated with the same probability distribution as that by which $\overline{\mathcal{P}}_0$ generates its public key. Note that this is the case regardless of the probability distribution of h_{GQ}.

The response that is computed by $\widehat{\mathcal{V}}$ in the simulated issuing protocol is the same as the response that $\overline{\mathcal{P}}_0$ would compute:

$$\begin{aligned}
r_0^v &= (r^{x_j}(x_0 \prod_{i=1}^{l} y_i^{x_i})^{c_0}\alpha)^v \\
&= (r^v)^{x_j}(x_0^v \prod_{i=1}^{l} (y_i^v)^{x_i})^{c_0}\alpha^v \\
&\overset{(\star)}{=} (h_{\text{GQ}}^{c} a)^{x_j}\alpha^v x_0^{vc_0}(\prod_{i\in\{1,\ldots,l\}\setminus\{j\}} g_i^{x_i})^{c_0} y_j^{vcx_j} \\
&= (h_{\text{GQ}} y_j^v)^{cx_j}(a^{x_j}\alpha^v)h_0^{c_0}(\prod_{i\in\{1,\ldots,l\}\setminus\{j\}} g_i^{x_i})^{c_0} \\
&= (g_j^{x_j})^{c_0} a_0 h_0^{c_0}(\prod_{i\in\{1,\ldots,l\}\setminus\{j\}} g_i^{x_i})^{c_0} \\
&= (h_0 h)^{c_0} a_0,
\end{aligned}$$

where the substitution $(\star)$ is allowed because the response of $\overline{\mathcal{P}}$ satisfies $r^v h_{\text{GQ}}^{-c} = a$. Since α is chosen at random from $\mathbb{Z}_n^*$, a_0 is randomly distributed over $\mathbb{Z}_n^*$ regardless of x_j. From this it follows that the view of $\widehat{\mathcal{V}}_0$ in the simulated issuing protocol has the same distribution as when $\widehat{\mathcal{V}}_0$ interacts with $\overline{\mathcal{P}}_0$, regardless of the probability distributions of $(x_1, \ldots, x_l)$, its challenges, and h_{GQ}. Therefore $\widehat{\mathcal{V}}$ moves from Step C to Step D with probability ϵ.

Because j is chosen at random by $\widehat{\mathcal{V}}$, and is uncorrelated to the view of $\widehat{\mathcal{V}}_0$ in the issuing protocol, $u_j \neq 0 \bmod v$ in Step D with probability at least $1/l$. (Not all u_i can be zero, because v is co-prime to $\varphi(n)$.)

The output of $\widehat{\mathcal{V}}$ in Step E is equal to $h_{\text{GQ}}^{1/v}$:

$$\begin{aligned}
(h_{\text{GQ}}^f(\prod_{i=1}^{l} y_i^{u_i} u_{l+1})^{-e})^v &= h_{\text{GQ}}^{fv}(\prod_{i=1}^{l}(y_i^v)^{u_i} u_{l+1}^v)^{-e} \\
&= h_{\text{GQ}}^{1-eu_j} y_j^{u_j v}(\prod_{i\in\{1,\ldots,l\}\setminus\{j\}} g_i^{u_i} u_{l+1}^v)^{-e} \\
&= h_{\text{GQ}}(h_{\text{GQ}} y_j^v)^{-eu_j}(\prod_{i\in\{1,\ldots,l\}\setminus\{j\}} g_i^{u_i} u_{l+1}^v)^{-e} \\
&= h_{\text{GQ}}(g_j^{u_j})^{-e}(\prod_{i\in\{1,\ldots,l\}\setminus\{j\}} g_i^{u_i} u_{l+1}^v)^{-e} \\
&= h_{\text{GQ}}(\prod_{i=1}^{l} g_i^{u_i} u_{l+1}^v)^{-e} \\
&= h_{\text{GQ}} 1^{-e} \\
&= h_{\text{GQ}}.
\end{aligned}$$

In all, the probability that $\widehat{\mathcal{V}}$ can compute the secret key of $\overline{\mathcal{P}}$ is at least ϵ/l. Since l is polynomial in k, this probability is non-negligible if ϵ is non-negligible. This contradicts the assumption. □

Note that the reduction is tight only if l is a (small) constant; it is not clear how to achieve tightness for arbitrary l polynomial in k.

We are now prepared for the main result.

Proposition 4.3.6. *If Assumption 2.5.9 is true, then a hash function $\mathcal{H}(\cdot)$ exists such that RSAREP-based scheme I is unforgeable.*

Proof. Take $\mathcal{H}(\cdot)$ equal to the hash function $\mathcal{H}^*(\cdot)$ defined in Assumption 2.5.9. Suppose that $\widehat{\mathcal{V}}_0$ obtains $t+1$ certified key pairs with non-negligible success probability ϵ after engaging in t executions of the certificate issuing protocol, for some $t \geq 0$. We construct a polynomial-time (interactive) algorithm $\widehat{\mathcal{V}}$ that can forge signatures in the interactive Guillou-Quisquater signature scheme.

Let (n, v) denote the system parameters in the Guillou-Quisquater proof of knowledge, and h_{GQ} the public key of $\overline{\mathcal{P}}$. $\widehat{\mathcal{V}}$ simulates $\overline{\mathcal{P}}_0$ with the help of the protocol executions of $\overline{\mathcal{P}}$, by performing the following steps:

Step A. (Simulate the key set-up for $\overline{\mathcal{P}}_0$.) Generate a random number $x_0 \in \mathbb{Z}_n^*$ and set $h_0 := h_{\text{GQ}} x_0^v$. Generate l random numbers $y_1, \ldots, y_l \in \mathbb{Z}_n^*$, and compute $g_i := y_i^v$, for all $i \in \{1, \ldots, l\}$. The simulated public key of $\overline{\mathcal{P}}_0$ is $h_0, (g_1, \ldots, g_l)$.

Step B. (Simulate $\overline{\mathcal{P}}_0$ in issuing protocol executions with respect to $(x_1, \ldots, x_l)$.)

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

Step 2. Receive c_0 from $\widehat{\mathcal{V}}$, and pass $c := c_0$ on to $\overline{\mathcal{P}}$.

Step 3. Receive r from $\overline{\mathcal{P}}$, and pass $r_0 := r(x_0 \prod_{i=1}^{l} y_i^{x_i})^{c_0}$ on to $\widehat{\mathcal{V}}_0$.

Repeat this simulation until t executions of the issuing protocol have been performed.

Step C. Check if $\widehat{\mathcal{V}}$ has $t+1$ distinct certified key pairs on its tapes. If not, then halt.

Step D. For each of these $t+1$ certified key pairs, $((x_1, \ldots, x_l, \alpha_1), h', (c_0', r_0'))$, compute $c^* := c_0'$, $r^* := r_0'(x_0 \prod_{i=1}^{l} y_i^{x_i} \alpha_1)^{-c_0'}$, and $m := h'$, and output the signed message $(m, (c^*, r^*))$.

It is easy to see that the public key in Step A is generated with the same probability distribution as that by which $\overline{\mathcal{P}}_0$ generates its public key. The response that is computed by $\widehat{\mathcal{V}}$ in the simulated issuing protocol is the same as the response that $\overline{\mathcal{P}}_0$ would compute:

$$\begin{aligned} r_0^v &= (r(x_0 \prod_{i=1}^{l} y_i^{x_i})^{c_0})^v \\ &= r^v x_0^{c_0 v} ((\prod_{i=1}^{l} (y_i^v)^{x_i})^{c_0} \\ &\overset{(\star)}{=} (h_{\text{GQ}}^c a) x_0^{cv} (\prod_{i=1}^{l} g_i^{x_i})^{c_0} \\ &= (h_{\text{GQ}} x_0^v)^c a h^{c_0} \\ &= (h_0 h)^{c_0} a_0, \end{aligned}$$

where the substitution $(\star)$ is allowed because the response of $\overline{\mathcal{P}}$ satisfies $r^v h_{\text{GQ}}^{-c} = a$. It follows that the view of $\widehat{\mathcal{V}}_0$ in the simulated issuing protocol has the same distribution as when $\widehat{\mathcal{V}}_0$ interacts with $\overline{\mathcal{P}}_0$, regardless of the probability distributions

of its challenges, $(x_1, \ldots, x_l)$, and h_{GQ}. Therefore, $\widehat{\mathcal{V}}$ moves from Step C to Step D with probability ϵ.

We next show (i) that the output of $\widehat{\mathcal{V}}$ consists of $t+1$ messages with corresponding Guillou-Quisquater signatures, and (ii) that these signed messages are all distinct with overwhelming probability. Property (i) follows from

$$\begin{aligned} c^* &= c'_0 \\ &\overset{(\star)}{=} \mathcal{H}_{n,v}(h', (r'_0)^v (h_0 h')^{-c'_0}) \\ &\overset{(\star\star)}{=} \mathcal{H}_{n,v}(m, (r^*)^v h_{\text{GQ}}^{-c^*}). \end{aligned}$$

The substitution $(\star)$ is allowed by definition of a certificate, and substitution $(\star\star)$ follows from

$$\begin{aligned} (r'_0)^v (h_0 h')^{-c'_0} &= (r^* (x_0 \prod_{i=1}^{l} y_i^{x_i} \alpha_1)^{c'_0})^v (h_0 \prod_{i=1}^{l} g_i^{x_i} \alpha_1^v)^{-c'_0} \\ &= (r^*)^v x_0^{c'_0 v} (\prod_{i=1}^{l} (y_i^v)^{x_i} \alpha_1^{v c'_0} h_0^{-c'_0} (\prod_{i=1}^{l} g_i^{x_i} \alpha_1^v)^{-c'_0} \\ &= (r^*)^v (h_0 x_0^{-v})^{-c'_0} \\ &= (r^*)^v h_{\text{GQ}}^{-c^*}. \end{aligned}$$

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

$$(x_1, \ldots, x_l, \alpha_1),\ h',\ (c'_0, r'_0)$$

and

$$(x_1^*, \ldots, x_l^*, \alpha_1^*),\ h^*,\ (c_0^*, r_0^*).$$

The corresponding signed messages, as computed by $\widehat{\mathcal{V}}$ in Step D, are equal to

$$h', (c'_0,\ r'_0((h')^{1/v})^{-c'_0})$$

and

$$h^*, (c_0^*,\ r_0^*((h^*)^{1/v})^{-c_0^*}).$$

Suppose that these two signed messages are the same. From $h' = h^*$ and $c'_0 = c_0^*$ it follows that $r'_0 = r_0^*$. Furthermore, if $(x_1, \ldots, x_l, \alpha_1)$ and $(x_1^*, \ldots, x_l^*, \alpha_1^*)$ are not the same, then

$$(x_1 - x_1^* \bmod v, \ldots, x_l - x_l^* \bmod v, \prod_{i=1}^{l} g_i^{(x_i - x_i^*) \operatorname{div} v} \alpha_1 / \alpha_1^*)$$

is a non-trivial RSA-representation of 1. According to Lemma 4.3.5, this contradicts Assumption 2.5.9. Consequently, if the two signed messages are the same, then the two certified key pairs are the same, and therefore property (ii) holds as well.

To complete the proof, observe that an execution of the simulated issuing protocol constitutes exactly one execution of the protocol with $\overline{\mathcal{P}}$. In all, $\widehat{\mathcal{V}}$ can compute $t+1$ Guillou-Quisquater signed messages from t protocol executions with $\overline{\mathcal{P}}$ with probability ϵ. If ϵ is non-negligible, this contradicts Assumption 2.5.9. □

Similar reductions can be made for the other three certificate schemes in Section 4.2. Because DLREP-based scheme II and RSAREP-based scheme II are non-trivially witness-indistinguishable, we get the following results by application of Proposition 2.5.3.

Proposition 4.3.7. *Assume that $\mathcal{P}_0$ performs no more than polylogarithmically many protocol executions, and that the binary size of the outputs of $\mathcal{H}_{g,g_0}(\cdot)$ is linear in k. If $(I_{\text{DL}}, D_{\text{DL}})$ is invulnerable for the DL function, then DLREP-based scheme II is unforgeable in the random oracle model, for any distribution of $(x_1, \ldots, x_l)$.*

Proposition 4.3.8. *Assume that $\mathcal{P}_0$ performs no more than polylogarithmically many protocol executions, and that the binary size of the outputs of $\mathcal{H}_{n,v}(\cdot)$ is linear in k. If $(I_{\text{RSA}}, D_{\text{RSA}})$ is invulnerable for the RSA function, then RSAREP-based scheme II is unforgeable in the random oracle model, for any distribution of $(x_1, \ldots, x_l)$.*

These results hold even in case $\mathcal{V}_0$ may arbitrarily interleave the protocol executions and $\mathcal{P}_0$ encodes different attribute tuples of $\mathcal{V}_0$'s choice.

Blinding-invariance

To study the restrictive blinding property, we slightly weaken the attack model by assuming that $(x_1, \ldots, x_l)$ is formed independently of h_0. In most applications this requirement is naturally met, especially if $\mathcal{P}_0$ selects $(x_1, \ldots, x_l)$.

The following assumption states that the only manner to generate a pair $h, (c, r)$ for which $c = \mathcal{H}_{n,v}(h, r^v h^{-c})$ is by forming h as the v-th power of some known $x \in \mathbb{Z}_n^*$. That is, if an algorithm could output such a transcript, then with "modest" extra effort it could also compute $h^{1/v} \bmod n$.

Assumption 4.3.9. *There exists a hash function $\mathcal{H}^*(\cdot) = \{\mathcal{H}_i^*(\cdot)\}_{i \in \{n,v\}}$ and an expected polynomial-time algorithm $\mathcal{K}$, such that for any polynomial-time algorithm A, for all constants $c > 0$, and for all sufficiently large k,*

$$\Big| \mathsf{P}_k\Big(A(n,v) = (h,c,r) \text{ such that } c = \mathcal{H}_{n,v}^*(h, r^v h^{-c}) \mid (n,v) := I_{\text{RSA}}(1^k)\Big) - \mathsf{P}_k\Big(\mathcal{K}((n,v),(h,c,r);A) = \beta \in \mathbb{Z}_n^* \text{ such that } \beta^v = h\Big) \Big| < 1/k^c.$$

This assumption can be proved in the random oracle model by using the oracle replay technique of Pointcheval and Stern [307]. [3]

Proposition 4.3.10. *If Assumption 4.3.9 holds, then a hash function $\mathcal{H}(\cdot)$ exists such that the following holds for all $l \geq 1$ and all $(x_1, \ldots, x_l)$. Let $(x_1, \ldots, x_l)$ be formed independently of $h_0, (g_1, \ldots, g_l)$, and be the same in all protocol executions of RSAREP-based scheme I. If $\widehat{\mathcal{V}}_0$, after engaging in polynomially many protocol executions with respect to $(x_1, \ldots, x_l)$, outputs a certified key pair comprising a secret key $(x_1^*, \ldots, x_l^*, \alpha_1)$, then*

$$(x_1^*, \ldots, x_l^*) = (x_1, \ldots, x_l)$$

with overwhelming probability.

Proof. Suppose that $\widehat{\mathcal{V}}_0$, after t protocol executions, outputs with non-negligible success probability ϵ a certified key pair comprising a secret key $(x_1^*, \ldots, x_l^*, \alpha_1)$ for which $(x_1^*, \ldots, x_l^*)$ differs from $(x_1, \ldots, x_l)$. Using a proper choice for $\mathcal{H}(\cdot)$, we show how to use algorithm $\mathcal{K}$ in Assumption 4.3.9 to construct a polynomial-time algorithm A for inverting the RSA function, thereby obtaining a contradiction.

Let $(I_{\text{RSA}}, D_{\text{RSA}})$ denote any invulnerable instance generator for the RSA function. On input k, this instance generator outputs a triple (n, v, x). Algorithm A, on input $(n, v, h_{\text{RSA}} := x^v)$, performs the following steps:

Step A. (Simulate the key set-up for $\overline{\mathcal{P}}_0$.) Generate l random numbers, $r_1, \ldots, r_l \in \mathbb{Z}_v$, and l random numbers, $s_1, \ldots, s_l \in \mathbb{Z}_n^*$. Set

$$g_i := h_{\text{RSA}}^{r_i} s_i^v \quad \forall i \in \{1, \ldots, l\}.$$

With h denoting $\prod_{i=1}^{l} g_i^{x_i}$, generate a random number $x_0 \in \mathbb{Z}_n^*$, and compute $h_0 := x_0^v h^{-1}$. (Since $(x_1, \ldots, x_l)$ is generated independently of h_0, we may assume that it is generated before $(h_0, g_1, \ldots, g_l)$ is generated.) The simulated public key of $\overline{\mathcal{P}}_0$ is $h_0, (g_1, \ldots, g_l)$. In addition, define $\mathcal{H}(\cdot)$ according to

$$\mathcal{H}_{n,v} : (a, b) \mapsto \mathcal{H}_{n,v}^*(h_0 a, b),$$

for all $a, b \in \mathbb{Z}_n^*$, where $\mathcal{H}^*(\cdot)$ is the hash function in Assumption 4.3.9.

Step B. (Simulate $\overline{\mathcal{P}}_0$ in issuing protocol executions with respect to $(x_1, \ldots, x_l)$.)

Step 1. Generate a random number $w_0 \in \mathbb{Z}_n^*$. Compute $a_0 := w_0^v$, and send a_0 to $\widehat{\mathcal{V}}_0$.

[3] Because A is non-interactive, it is unclear how to formalize knowledge extraction outside of the random oracle model. The intuition is that if A would keep a "history" tape that contains a copy of everything it has written on its work tape (but with previous contents never overwritten), then $\mathcal{K}$ should be able to extract knowledge from A by looking at the history tape and A's input tape and random tape.

Step 2. Receive c_0 from $\widehat{\mathcal{V}}_0$.

Step 3. Compute $r_0 := x_0^{c_0} w_0$, and send r_0 to $\widehat{\mathcal{V}}_0$.

Repeat this simulation until t executions of the issuing protocol with $\widehat{\mathcal{V}}_0$ have been performed.

Step C. Check if $\widehat{\mathcal{V}}_0$ has output a certified key pair $(x_1^*, \ldots, x_l^*, \alpha_1), h', (c_0', r_0')$ for which $(x_1^*, \ldots, x_l^*)$ does not equal $(x_1, \ldots, x_l)$. If this is not the case, then halt.

Step D. If $\sum_{i=1}^{l} r_i(x_i - x_i^*) = 0 \bmod v$, then halt.

Step E. Run algorithm $\mathcal{K}$ on input (n, v) and $(h', (c_0', r_0'))$, using $< A, \widehat{\mathcal{V}}_0 >$ as a black-box algorithm. If $\mathcal{K}$ does not output $\beta \in \mathbb{Z}_n^*$ such that $\beta^v = h' \bmod n$, then halt.

Step F. Using the extended Euclidean algorithm, compute integers $e, f \in \mathbb{Z}$ satisfying

$$e(\sum_{i=1}^{l} r_i(x_i - x_i^*)) + fv = 1.$$

(This can always be done, because v is prime.) Compute

$$h_{\text{RSA}}^f(\alpha_1 x_0 \beta^{-1} \prod_{i=1}^{l} s_i^{x_i - x_i^*})^e,$$

and output the result.

By definition of the key generation of A in Step A, the public key in Step A is simulated with the same probability distribution as that by which $\overline{\mathcal{P}}_0$ generates its public key, regardless of the distribution of h_{RSA} and $(x_1, \ldots, x_l)$. The response that is computed by A in the simulated issuing protocol is the same as the response that $\overline{\mathcal{P}}_0$ would compute:

$$\begin{aligned} r_0^v &= (x_0^{c_0} w_0)^v \\ &= (x_0^v)^{c_0} w_0^v \\ &= (h_0 h)^{c_0} a_0. \end{aligned}$$

It follows that the view of $\widehat{\mathcal{V}}_0$ in the simulated issuing protocol has the same distribution as that provided by $\overline{\mathcal{P}}_0$, regardless of the probability distribution by which $\widehat{\mathcal{V}}_0$ generates its challenges. Therefore, Step D is reached by supposition with probability ϵ.

The tuple $(r_1, \ldots, r_l)$ is unconditionally hidden from $\widehat{\mathcal{V}}_0$, due to the randomness of the s_i's and the fact that v is co-prime to $\varphi(n)$, and it is therefore independent of

$(x_1^*, \ldots, x_l^*)$. Because $(r_1, \ldots, r_l)$ is also independent of $(x_1, \ldots, x_l)$, the transition from Step D to Step E takes place with probability $1 - 1/v$.

Because of the definition of $\mathcal{H}(\cdot)$, we can infer from Assumption 4.3.9 that the output β of $\mathcal{K}$ in Step E satisfies

$$\beta^v = h_0 h' = h_0 \prod_{i=1}^{l} g_i^{x_i^*} \alpha_1^v$$

with non-negligible probability. Therefore, the transition from Step E to Step F takes place with non-negligible probability.

According to the key pair construction in Step A we also have

$$x_0^v = h_0 h = h_0 \prod_{i=1}^{l} g_i^{x_i}.$$

From these two relations we get

$$\begin{aligned}
(x_0\beta^{-1})^v &= \prod_{i=1}^{l} g_i^{x_i - x_i^*} (\alpha_1^v)^{-1} \\
&= \prod_{i=1}^{l} (h_{\text{RSA}}^{r_i} s_i^v)^{x_i - x_i^*} (\alpha_1^v)^{-1} \\
&= h_{\text{RSA}}^{\sum_{i=1}^{l} r_i (x_i - x_i^*)} (\prod_{i=1}^{l} s_i^{x_i - x_i^*} \alpha_1^{-1})^v
\end{aligned}$$

and so

$$(\alpha_1 x_0 \beta^{-1} \prod_{i=1}^{l} s_i^{x_i^* - x_i})^v = h_{\text{RSA}}^{\sum_{i=1}^{l} r_i (x_i - x_i^*)}.$$

From this it follows that the output of A in Step F is equal to $h_{\text{RSA}}^{1/v}$, with overwhelming probability:

$$\begin{aligned}
\Big(h_{\text{RSA}}^f (\alpha_1 x_0 \beta^{-1} \prod_{i=1}^{l} s_i^{x_i - x_i^*})^e\Big)^v &= h_{\text{RSA}}^{fv} \Big((\alpha_1 x_0 \beta^{-1} \prod_{i=1}^{l} s_i^{x_i - x_i^*})^v\Big)^e \\
&= h_{\text{RSA}}^{1 - e(\sum_{i=1}^{l} r_i (x_i - x_i^*))} (h_{\text{RSA}}^{\sum_{i=1}^{l} r_i (x_i - x_i^*)})^e \\
&= h_{\text{RSA}}.
\end{aligned}$$

The overall success probability of A is $(1 - 1/v)\epsilon$ times the (non-negligible) success probability of algorithm $\mathcal{K}$. If ϵ is non-negligible, then $(I_{\text{RSA}}, D_{\text{RSA}})$ is not invulnerable for the RSA function. Therefore $(x_1^*, \ldots, x_l^*) = (x_1, \ldots, x_l)$ with overwhelming probability. □

The result holds regardless of the fashion in which protocol executions with respect to the same attribute tuple are interleaved.

The hash function defined in the proof of Proposition 4.3.10 is not the same as that in the proof of Proposition 4.3.6. In practice, any sufficiently strong one-way hash function should suffice for both propositions. (Another approach is to adjust Assumption 4.3.9.)

A similar result can be proved for the other three certificate schemes described in Section 4.2. In all four schemes, $\mathcal{P}_0$ is effectively proving knowledge of a representation of the joint public key $h_0 h$, by means of a protocol that we know from Section 2.4 to be honest-verifier zero-knowledge. Since c_0 as formed by $\overline{\mathcal{V}}_0$ is randomly distributed, wiretappers cannot infer anything from the protocol executions of honest receivers. More generally, the blinding-invariance property remains valid even if $\widehat{\mathcal{V}}_0$ can wiretap the issuing as well as the showing protocol executions of honest parties, assuming that these use their certified key pairs only in zero-knowledge showing protocols.

The following negative result shows that Proposition 4.3.10 cannot easily be generalized.

Proposition 4.3.11. *If $\overline{\mathcal{P}}_0$ performs protocol executions in parallel with respect to different attribute tuples, then $\widehat{\mathcal{V}}_0$ can obtain a certified key pair for which the putative restrictive blinding-invariant part is not equal to any of these tuples.*

Proof. Suppose that $\overline{\mathcal{P}}_0$ performs its protocol executions with respect to $t > 1$ different attribute tuples, $(x_{11}, \ldots, x_{l1}), \ldots, (x_{1t}, \ldots, x_{lt})$. In the following attack, $\mathcal{V}_0$ engages in parallel in t protocol executions, each with respect to one of the tuples. Assume without loss of generality that the j-th protocol execution is with respect to the tuple $(x_{1j}, \ldots, x_{lj})$ and let $h_j := \prod_{i=1}^{l} g_i^{x_{ij}}$, for all $j \in \{1, \ldots, t\}$.[4]

Step 1. $\widehat{\mathcal{V}}_0$ obtains t numbers, $a_{01}, \ldots, a_{0t} \in \mathbb{Z}_n^*$ from $\overline{\mathcal{P}}_0$, by engaging in Step 1 of all t protocol executions.

Step 2. $\widehat{\mathcal{V}}_0$ chooses t numbers, $\alpha_1, \ldots, \alpha_t \in \mathbb{Z}_v$, subject to $\sum_{i=1}^{t} \alpha_i = 1 \bmod v$. $\mathcal{V}_0$ computes

$$h' := \prod_{i=1}^{t} h_i^{\alpha_i}$$

and

$$c_0' := \mathcal{H}_{n,v}(h', \prod_{i=1}^{t} a_{0i}).$$

For all $i \in \{1, \ldots, t\}$, $\widehat{\mathcal{V}}_0$ then computes $c_{0i} := \alpha_i c_0' \bmod v$ and sends c_{0i} to $\overline{\mathcal{P}}_0$ in Step 2 of the i-th protocol execution.

[4] The ordering of protocol executions assumed here follows for instance from the time order in which $\mathcal{V}_0$ processes the first message of each protocol execution.

Step 3. $\widehat{\mathcal{V}}_0$ obtains t numbers, $r_{01}, \ldots, r_{0t} \in \mathbb{Z}_n^*$ from $\overline{\mathcal{P}}_0$, by engaging in Step 3 of all t protocol executions.

If $\widehat{\mathcal{V}}_0$ accepts in all t protocol executions, it computes

$$r_0' := \prod_{i=1}^{t} r_{0i}(h_0 h_i)^{(\alpha_i c_0') \operatorname{div} v} h_0^{-((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)}.$$

(The additional operations needed to blind the certified key pair have been left out only for reason of clarity; they are easy to incorporate.)

If the t responses of $\mathcal{P}_0$ are all correct, then (c_0', r_0') is a certificate of $\mathcal{P}_0$ on h':

$$\begin{aligned}
(r_0')^v &= \Big(\prod_{i=1}^{t} r_{0i}(h_0 h_i)^{(\alpha_i c_0') \operatorname{div} v} h_0^{-((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)}\Big)^v \\
&= \prod_{i=1}^{t} r_{0i}^v (h_0 h_i)^{v((\alpha_i c_0') \operatorname{div} v)} h_0^{-v((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)} \\
&= \prod_{i=1}^{t} (h_0 h_i)^{c_{0i}} a_{0i} (h_0 h_i)^{v((\alpha_i c_0') \operatorname{div} v)} h_0^{-v((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)} \\
&= \prod_{i=1}^{t} (h_0 h_i)^{\alpha_i c_0' \bmod v} a_{0i} (h_0 h_i)^{v((\alpha_i c_0') \operatorname{div} v)} h_0^{-v((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)} \\
&= \prod_{i=1}^{t} (h_0 h_i)^{\alpha_i c_0'} a_{0i} h_0^{-v((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)} \\
&= (h_0^{\sum_{i=1}^{t} \alpha_i})^{c_0'} \prod_{i=1}^{t} h_i^{\alpha_i c_0'} a_{0i} h_0^{-v((\sum_{i=1}^{t} \alpha_i) \operatorname{div} v)} \\
&= (h_0^{\sum_{i=1}^{t} \alpha_i \bmod v})^{c_0'} \prod_{i=1}^{t} (h_i^{\alpha_i})^{c_0'} a_{0i} \\
&= h_0^{c_0'} (h')^{c_0'} \prod_{i=1}^{t} a_{0i} \\
&= (h_0 h')^{c_0'} \prod_{i=1}^{t} a_{0i}.
\end{aligned}$$

From

$$h' = \prod_{j=1}^{l} g_j^{\sum_{i=1}^{t} \alpha_i x_{ij}}$$

it is clear that $\widehat{\mathcal{V}}_0$ can obtain a certified key pair for which the secret key does not contain any of the t attribute tuples with respect to which the protocol executions have been performed. □

In fact, if $t > l$ then $\widehat{\mathcal{V}}_0$ can target any attribute tuple it desires, assuming a certain linear independence property; see Proposition 4.3.15 for details.

The attack in the proof of Proposition 4.3.11 requires $\widehat{\mathcal{V}}_0$ to engage in parallel executions of the issuing protocol, because each of the t challenges of $\widehat{\mathcal{V}}_0$ depends on all t initial witnesses. In case $\overline{\mathcal{P}}_0$ does not perform protocol executions with respect to different attribute tuples in parallel, it seems that $\widehat{\mathcal{V}}_0$ can only obtain certified key pairs that comprise one of the t tuples with respect to which the protocol executions have been performed. This isolation property is formalized by the following assumption.

Assumption 4.3.12. *There exists a hash function $\mathcal{H}(\cdot)$ such that the following holds for all $l, t \geq 1$. Let t attribute tuples,*

$$(x_{11}, \ldots, x_{l1}), \ldots, (x_{1t}, \ldots, x_{lt}),$$

be formed. Let $(x_1^*, \ldots, x_l^*, \alpha)$ denote the secret key of a certified key pair computed by $\widehat{\mathcal{V}}_0$ in RSAREP-based scheme I after engaging in polynomially many protocol executions with respect to tuples $(x_{1i}, \ldots, x_{li})$, $i \in \{1, \ldots, t\}$ of its own choice (possibly adaptively chosen). If $\overline{\mathcal{P}}_0$ does not perform protocol executions with respect to distinct attribute tuples in parallel, then with overwhelming probability there exists $i \in \{1, \ldots, t\}$ such that $(x_1^*, \ldots, x_l^*) = (x_{1i}, \ldots, x_{li})$. More generally, the second property in Definition 4.1.1 holds.

For DLREP-based scheme II and RSAREP-based scheme II, the analogous assumption can be proved in the random oracle model, provided that $\mathcal{P}_0$ performs no more than polylogarithmically many protocol executions. Note that the assumption does not forbid protocol executions with respect to the same attribute tuple to be performed in parallel.

4.3.4 Additional properties

$\mathcal{P}_0$ knows the numbers $(x_1, \ldots, x_l)$ that end up in the secret key of $\overline{\mathcal{V}}_0$. Once $\mathcal{P}_0$ gets to see h' and for some reason (for instance because x_1 is an identifier that $\mathcal{V}_0$ discloses in the showing protocol) is able to link it to the protocol execution in which it was certified, it can compute $h'/h = \alpha_1^v$. According to Proposition 4.3.4, α_1 is uncorrelated to the view of $\widetilde{\mathcal{P}}_0$ in $(\widetilde{\mathcal{P}}_0, \overline{\mathcal{V}}_0)$. In Section 2.2.3 we have seen that if any D_{RSA} leads to a one-way RSA function, then a random choice for α_1 certainly will. From this we get the following result.

Corollary 4.3.13. *If $(I_{\text{RSA}}, D_{\text{RSA}})$ is invulnerable for the RSA-function, and does not output the factorization of n as side information, then $\widehat{\mathcal{P}}_0$ cannot compute the secret*

key of $\overline{\mathcal{V}}_0$ from the certified public key of $\overline{\mathcal{V}}_0$ even if $\widehat{\mathcal{P}}_0$ knows the encoded attribute tuple $(x_1, \ldots, x_l)$.

Therefore, only $\mathcal{V}_0$ can feasibly perform a (signed) proof of knowledge of a secret key corresponding to its certified public key(s). Note that the interests of $\mathcal{P}_0$ and $\mathcal{V}_0$ are aligned, because $(I_{\text{RSA}}, D_{\text{RSA}})$ needs to be invulnerable to guarantee the unforgeability of certified key pairs. If the issuing protocol is combined with one of the RSAREP-based showing protocols of Chapter 3, and $\mathcal{V}_0$ does not disclose at least part of the encoded attribute tuple, then not even $\widehat{\mathcal{P}}_0$ will be able to determine $\mathcal{V}_0$'s secret key. The latter property is desirable to achieve non-repudiation, especially in the case of limited-show certificates; see Section 5.5.3 for details.

A similar result holds for the other three certificate schemes constructed in Section 4.2. The DLREP-based schemes have the advantage that a trapdoor is not known to exist, so that $\mathcal{P}_0$ may generate the system parameters by itself.

The following property clarifies the nature of the certificate scheme.

Proposition 4.3.14. *RSAREP-based scheme I is a secret-key certificate scheme.*

Proof. We construct a polynomial-time simulation algorithm S that generates certified public keys with the same probability distribution as that according to which they are generated in the issuing protocol between $\overline{\mathcal{P}}_0$ and $\overline{\mathcal{V}}_0$. On given as input $n, v, h_0, (g_1, \ldots, g_l)$ and $\mathcal{H}_{n,v}(\cdot)$, S generates two random numbers $\alpha_2, \alpha_3 \in \mathbb{Z}_n^*$, computes $h := h_0^{-1}\alpha_2^v$, $c_0 := \mathcal{H}_{n,v}(h, \alpha_3^v)$ and $r_0 := \alpha_2^{c_0}\alpha_3$, and outputs the pair h, (c_0, r_0). The output of S is a certified public key:

$$\begin{aligned} c_0 &= \mathcal{H}_{n,v}(h, \alpha_3^v) \\ &= \mathcal{H}_{n,v}(h, (r_0\alpha_2^{-c_0})^v) \\ &= \mathcal{H}_{n,v}(h, r_0^v(\alpha_2^v)^{-c_0}) \\ &= \mathcal{H}_{n,v}(h, r_0^v(h_0 h)^{-c_0}). \end{aligned}$$

Since v is co-prime to $\varphi(n)$, and α_2 and α_3 in Step 1 are chosen at random from $\mathbb{Z}_n^*$, the output distribution of A is identical to that of certified public keys issued to $\overline{\mathcal{V}}_0$ by $\overline{\mathcal{P}}_0$. □

The other three schemes described in Section 4.2 are secret-key certificate schemes as well. For the advantages of this property, see Section 2.6 and Section 5.2.2.

For any of the certificate schemes in Section 4.2, a limited degree of parallelization can be achieved without any modifications. Observe that the crux of the proof of Proposition 4.3.10 is that ($\mathcal{P}_0$, $\mathcal{V}_0$) is a proof of knowledge of the v-th root of $h_0 h$, but not of the v-th root of h_0. Elaborating on this observation, we can obtain the following result.

Proposition 4.3.15. *There exists a hash function $\mathcal{H}(\cdot)$ such that the following holds. Let $(x_1^*, \ldots, x_l^*, \alpha)$ denote the secret key of a certified key pair computed by $\widehat{\mathcal{V}}_0$ in*

RSAREP-based scheme I after engaging in polynomially many protocol executions (that may be arbitrarily interleaved) with respect to attribute tuples $(x_{1i}, \ldots, x_{li})$*, for* $i \in \{1, \ldots, t\}$*, of its own choice, subject to the restriction that the tuples are formed independently of* h_0*. If Assumption 4.3.9 holds, then for all* $l, t \geq 1$*, with overwhelming probability* $(1, x_1^*, \ldots, x_l^*)$ *is contained in the linear span of the* t *vectors* $(1, x_{1i}, \ldots, x_{li})$*, for* $i \in \{1, \ldots, t\}$*. In particular, if* $t \leq l$ *the second property in Definition 4.1.1 holds.*

In other words, if $\mathcal{P}_0$ performs protocol executions with respect to up to $t \leq l$ independent tuples in parallel, it can still encode $l - t + 1$ attributes into the secret key of each certified key pair that $\widehat{\mathcal{V}}_0$ ends up with. The same holds for the DLREP-based schemes. This immunization technique is not very practical, though, because the degree of parallelization depends on l and the number of attributes to be encoded. In the next section we show how to guarantee security in the presence of arbitrary parallelization.

4.4 Parallelization of protocol executions

Whether or not the measure of not running protocol executions with respect to different attribute tuples in parallel poses a performance bottleneck depends on the application at hand. Sequential protocol executions need not be inefficient, because $\mathcal{P}_0$ can send out a_0 for a new protocol execution as soon as it has received the challenge c_0 for the current protocol execution. To prevent queuing, $\mathcal{P}_0$ should abort an execution of the issuing protocol if a predetermined time lag between the transmittal of a_0 and the reception of c_0 is exceeded; the receiver must then try again in a later protocol execution. Assuming that requests for protocol executions arrive in accordance with a Poisson process, this strategy is the M/D/1 model with feedback known from queueing theory. The feedback may be purposely limited by $\mathcal{P}_0$, to shut out parties that frequently exceed the permitted time lag. Furthermore, executions of the certificate issuing protocol can be scheduled to take place at a convenient time and can be repeated if necessary. Also, remember that protocol executions with respect to the same attribute tuple may always be performed in parallel.

The ability to arbitrarily interleave protocol executions offers two benefits in highly demanding applications:

- The role of $\mathcal{P}_0$ can be performed by distributed processors that need not communicate or synchronize; they merely need access to the same secret key.

- Receivers can go off-line between Step 1 and Step 2, in principle for as long as they please. $\mathcal{P}_0$ in Step 1 could even send to $\mathcal{V}_0$ an authenticated encryption of the random bits it used to form its initial witness, and have $\mathcal{V}_0$ return it in Step 2. (Obviously, $\mathcal{P}_0$ must prevent replay.) In the RSAREP-based protocols

$\mathcal{V}_0$ may even form a_0 on its own as the output of a sufficiently strong one-way function, as pointed out in Section 4.2.3.

In the following two sections we describe two techniques to "immunize" the certificate schemes of Section 4.2 against parallel mode attacks. Both immunizations admit arbitrary parallelization, and do not affect the definition of the system parameters and $\mathcal{P}_0$'s public key; only the definition of a certificate changes slightly.

4.4.1 Masking the initial witness

Our first immunization technique aims to destroy the multiplicative relation in the initial witnesses that is exploited by $\mathcal{V}_0$ in Step 2 of the parallel mode attack of Proposition 4.3.11. It applies to both DLREP-based schemes and to both RSAREP-based schemes.

Concretely, to enable full parallelization of protocol executions in RSAREP-based scheme I, we have $\mathcal{P}_0$ send $f_{n,v}(a_0)$ instead of a_0 in Step 1 of the issuing protocol. The function $\{f_i(\cdot)\}_{i\in\{(n,v)\}}$ must satisfy the following two requirements:

1. For random $a_0, b_0 \in \mathbb{Z}_n^*$, it is easy to compute $f_{n,v}(a_0 b_0)$ from $f_{n,v}(a_0)$ and b_0.

2. For random $a_0, b_0 \in \mathbb{Z}_n^*$, it is infeasible to compute a triple

$$\alpha \neq 0 \bmod v, \beta \neq 0 \bmod v, f_{n,v}(a_0^\alpha b_0^\beta)$$

from $f_{n,v}(a_0)$ and $f_{n,v}(b_0)$.

The first requirement ensures that $\mathcal{V}_0$ can retrieve certified public keys in exactly the same manner as in the original issuing protocol, while the second requirement prevents parallel mode attacks based on the exploitation of multiplicative properties. The second requirement may be weakened by having $\mathcal{P}_0$ time the delay between sending out a_0 and receiving c_0, aborting when a predetermined time bound is exceeded; it then suffices that triples $\alpha \neq 0 \bmod v, \beta \neq 0 \bmod v, f_{n,v}(a_0^\alpha b_0^\beta)$ cannot be computed within the imposed time bound. Note that we do not require that the computation of $f_{n,v}(a_0^\alpha)$ from $f_{n,v}(a_0)$ be infeasible.

Correspondingly, the following modifications must be made to RSAREP-based scheme I:

- The pair (c_0', r_0') is redefined to be a certificate of $\mathcal{P}_0$ on h' if and only if

$$c_0' = \mathcal{H}_{n,v}(h', f_{n,v}((r_0')^v (h_0 h')^{-c_0'})).$$

- In Step 1 of the issuing protocol, $\mathcal{P}_0$ sends $f_{n,v}(a_0)$ instead of a_0.

- In Step 2 of the issuing protocol, $\mathcal{V}_0$ computes c_0' according to

$$c_0' := \mathcal{H}_{n,v}(h', f_{n,v}(\alpha_2^v (h_0 h)^{\alpha_3} a_0)).$$

  $\mathcal{V}_0$ can compute c_0' by virtue of the first requirement for $f(\cdot)$.

- $\mathcal{V}_0$ accepts if and only if $f_{n,v}(r_0^v (h_0 h)^{-c_0})$ is equal to the number provided by $\mathcal{P}_0$ in Step 1.

Note that the definition of a key pair for $\mathcal{V}_0$ is not affected; only the definition of a certificate is changed. The resulting scheme is depicted in Figure 4.6.

Proposition 4.4.1. *If $f(\cdot)$ is one-to-one, then the immunized RSAREP-based scheme I is at least as secure as the original scheme.*

The proof is trivial: the security of the immunized scheme is easily seen to reduce to that of the original scheme in case it is feasible to invert $f(\cdot)$.

Assumption 4.4.2. *There exists a function $f(\cdot)$ and a hash function $\mathcal{H}(\cdot)$ such that the issuing protocol of the immunized RSAREP-based scheme I is restrictive blind with blinding-invariant part $(x_1, \ldots, x_l)$, even when protocol executions with respect to different attribute tuples are arbitrarily interleaved.*

A concrete suggestion for a one-to-one function $f(\cdot)$ satisfying our two requirements is the following. Let M be a random prime such that n divides $M-1$, and let F be a random element of order n in $\mathbb{Z}_M^*$. Define

$$f_{n,v} : a_0 \rightarrow F^{a_0} \bmod M \quad \forall a_0 \in \mathbb{Z}_n^*.$$

It is easy to see that the first requirement for $f(\cdot)$ is met. Whether the second requirement is met depends on the hardness of the *Diffie-Hellman problem* [136]; this is the problem of computing g^{ab}, on input (g, g^a, g^b) for random a, b and a random group element g of large order. It is widely believed that there exist groups in which the ability to solve the Diffie-Hellman problem is polynomial-time equivalent to the ability to compute discrete logarithms; see Maurer and Wolf [258] for partial evidence.

Proposition 4.4.3. *Suppose there exist positive integers (α, β) and a polynomial-time algorithm that, on given as input a randomly chosen tuple (n, M, F) of the specified format and a pair $(F^{a_0} \bmod M, F^{b_0} \bmod M)$ for randomly chosen a_0, b_0 in $\mathbb{Z}_n^*$, outputs $F^{a_0^\alpha b_0^\beta} \bmod M$ with non-negligible success probability. Then the Diffie-Hellman problem in groups $\mathbb{Z}_M^*$, with M of the specified form, is tractable.*

The proof of this proposition makes use of standard techniques, and is therefore omitted.

Proposition 4.4.3 does not suffice to prove the second requirement, because it pertains only to algorithms that compute $F^{a_0^\alpha b_0^\beta} \bmod M$ for fixed (α, β). Nevertheless, it provides evidence in favor of $f(\cdot)$ meeting the second requirement.

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(n, v) := I_{\text{RSA}}(1^k)$	
	KEY SET-UP	
Secret key: factorization of n		
$h_0, g_1, \ldots, g_l \in_{\mathcal{R}} \mathbb{Z}_n^*$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{n,v}(\cdot), f_{n,v}(\cdot)$	
	PROTOCOL	
$a_0 \in_{\mathcal{R}} \mathbb{Z}_n^*$		
$a_0^* := f_{n,v}(a_0)$		
	$\xrightarrow{a_0^*}$	
		$\alpha_1, \alpha_2 \in_{\mathcal{R}} \mathbb{Z}_n^*$
		$\alpha_3 \in_{\mathcal{R}} \mathbb{Z}_v$
		$h' := h\alpha_1^v$
		$c_0' := \mathcal{H}_{n,v}(h', f_{n,v}(\alpha_2^v (h_0 h)^{\alpha_3} a_0))$
		$c_0 := c_0' + \alpha_3 \bmod v$
	$\xleftarrow{c_0}$	
$r_0 := ((h_0 h)^{c_0} a_0)^{1/v}$		
	$\xrightarrow{r_0}$	
		$f_{n,v}(r_0^v (h_0 h)^{-c_0}) \stackrel{?}{=} a_0^*$
		$r_0' := r_0 \alpha_2 \alpha_1^{c_0'} (h_0 h)^{(c_0' + \alpha_3) \operatorname{div} v}$

Figure 4.6: Immunization I of RSAREP-based scheme I.

This immunization technique also applies to the other three certificate schemes described in Section 4.2. Its drawback is decreased performance: $\mathcal{V}_0$ cannot precompute the application of $f_{n,v}(\cdot)$ in Step 2, and certificates are larger and more costly to verify. Furthermore, an elliptic curve implementation of the immunized DLREP-based schemes seems out of the question.

4.4.2 Swapping exponents in the verification relation

The second immunization technique applies to both DLREP-based schemes as well as to RSAREP-based scheme II, and fully preserves their efficiency. On the downside, it does not apply to RSAREP-based scheme I, and it is unclear how to prove unforgeability in the random oracle model.

The required modifications are the result of swapping the position of the challenge with that of (one of) the response(s) in the verification relation. In the case of DLREP-based scheme I, the certificate verification relation

$$c_0' = \mathcal{H}_{q,g_0}(h', g_0^{r_0'}(h_0 h')^{-c_0'}),$$

becomes

$$c_0' = \mathcal{H}_{q,g_0}(h', g_0^{c_0'}(h')^{r_0'}).$$

The secret key of $\mathcal{V}_0$ is redefined to be a DL-representation of h' with respect to $(g_1, \ldots, g_l, h_0)$, instead of with respect to $(g_1, \ldots, g_l, g_0)$. No changes are needed to the process of generating the system parameters. The issuing protocol is modified correspondingly, as follows:

Step 1. $\mathcal{P}_0$ generates a random number $w_0 \in \mathbb{Z}_q$, and sends $a_0 := g_0^{w_0}$ to $\mathcal{V}_0$.

Step 2. $\mathcal{V}_0$ generates three random numbers $\alpha_1 \in \mathbb{Z}_q^*$ and $\alpha_2, \alpha_3 \in \mathbb{Z}_q$. $\mathcal{V}_0$ computes $h' := (h_0 h)^{\alpha_1}$, $c_0' := \mathcal{H}_{q,g_0}(h', g_0^{\alpha_2}(h_0 h)^{\alpha_3} a_0)$, and sends $c_0 := c_0' - \alpha_2 \bmod q$ to $\mathcal{P}_0$.

Step 3. $\mathcal{P}_0$ sends $r_0 := (w_0 - c_0)/(x_0 + \sum_{i=1}^{l} x_i y_i) \bmod q$ to $\mathcal{V}_0$.[5]

$\mathcal{V}_0$ accepts if and only if $g_0^{c_0}(h_0 h)^{r_0} = a_0$. If this verification holds, $\mathcal{V}_0$ computes $r_0' := (r_0 + \alpha_3)/\alpha_1 \bmod q$. The resulting scheme is depicted in Figure 4.7.

It is easy to verify that the protocol is complete, and that (c_0', r_0') is a secret-key certificate of $\mathcal{P}_0$ on h'. Excluding public keys h' that are equal to 1, the following result can be proved in a manner similar to the proof of Proposition 4.3.4.

[5] To guarantee that $\mathcal{P}_0$ can always perform Step 3, the attribute tuple that is encoded must satisfy $(x_0 + \sum_{i=1}^{l} x_i y_i) \neq 0 \bmod q$. Since finding a tuple for which equality hold should be infeasible for $\mathcal{V}_0$ there is no need to check for this.

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(q, g_0) := I_{\text{DL}}(1^k)$	
	KEY SET-UP	
$x_0 := D_{\text{DL}}(q, g_0)$		
$y_1, \ldots, y_l \in_{\mathcal{R}} \mathbb{Z}_q$		
Secret key: $x_0, (y_1, \ldots, y_l)$		
$h_0 := g_0^{x_0}$		
$g_i := g_0^{y_i}\ \forall i \in \{1, \ldots, l\}$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{q,g_0}(\cdot)$	
	PROTOCOL	
$w_0 \in_{\mathcal{R}} \mathbb{Z}_q$		
$a_0 := g_0^{w_0}$		
	$\xrightarrow{a_0}$	
		$\alpha_1 \in_{\mathcal{R}} \mathbb{Z}_q^*$
		$\alpha_2, \alpha_3 \in_{\mathcal{R}} \mathbb{Z}_q$
		$h' := (h_0 h)^{\alpha_1}$
		$c_0' := \mathcal{H}_{q,g_0}(h', g_0^{\alpha_2}(h_0 h)^{\alpha_3} a_0)$
		$c_0 := c_0' - \alpha_2 \bmod q$
	$\xleftarrow{c_0}$	
$r_0 := (w_0 - c_0)/(x_0 + \sum_{i=1}^{l} x_i y_i) \bmod q$		
	$\xrightarrow{r_0}$	
		$g_0^{c_0}(h_0 h)^{r_0} \stackrel{?}{=} a_0$
		$r_0' := (r_0 + \alpha_3)/\alpha_1 \bmod q$

Figure 4.7: Immunization II of DLREP-based scheme I.

Proposition 4.4.4. *For any properly formed system parameters in the immunized DLREP-based scheme I, if $\mathcal{V}_0$ follows the issuing protocol and accepts, then it obtains a certified key pair comprising a perfectly blinded certified public key, regardless of the behavior of $\widetilde{\mathcal{P}}_0$.*

Obtaining a certificate on $h' = 1$ seems infeasible; it implies the ability to compute a number $c_0 \in \mathbb{Z}_q$ such that $\mathcal{H}_{q,g_0}(1, g_0^{c_0}) = c_0$. However, there is no need to make an assumption to this effect, since this case can be recognized and declared invalid.

While it would seem that the unforgeability of the modified certificate scheme can be proved in a manner similar to the proof of Proposition 4.3.6, this is not the case. Nevertheless, unforgeability is believed to hold for the modified scheme as well.

We now arrive at the crucial difference with DLREP-based scheme I. The parallel mode attack described in the proof of Proposition 4.3.11 does not apply, because $\mathcal{V}_0$ in Step 2 has to solve linear relations in terms of the responses of $\mathcal{P}_0$, which it cannot anticipate at that time.

Assumption 4.4.5. *There exists a hash function $\mathcal{H}(\cdot)$ such that in the immunized DLREP-based scheme I the following holds for all $l, t \geq 1$. Let t attribute tuples,*

$$(x_{11}, \ldots, x_{l1}), \ldots, (x_{1t}, \ldots, x_{lt}),$$

be formed. Suppose that $\widehat{\mathcal{V}}_0$, after engaging in polynomially many protocol executions (arbitrarily interleaved) with respect to tuples $(x_{1i}, \ldots, x_{li})$, $i \in \{1, \ldots, t\}$ of its own choice (possibly adaptively chosen), outputs a certified key pair comprising a secret key $(x_1^*, \ldots, x_l^*, \alpha_1)$. With overwhelming probability, there exists $i \in \{1, \ldots, t\}$ such that $(x_1^*, \ldots, x_l^*) = (\alpha_1 x_{1i} \bmod q, \ldots, \alpha_1 x_{li} \bmod q)$. More generally, the second property in Definition 4.1.1 holds.

Assuming that public keys equal to 1 are declared invalid, it follows that $\alpha_1 \neq 0$, and so

$$(x_1^*/\alpha_1 \bmod q, \ldots, x_l^*/\alpha_1 \bmod q) = (x_{1i}, \ldots, x_{li}).$$

The following argument gives some insight as to why the assumption should hold. If we restrict ourselves in Assumption 4.4.5 to protocol executions that involve the same $(x_1, \ldots, x_l)$, which is formed independently of h_0, then the proof of Proposition 4.3.10 applies in virtually the same manner, assuming the DL-based analogue to Assumption 4.3.9. Therefore, attacks must exploit the parallel nature of the issuing protocol with respect to different attribute tuples, if they are to have a non-negligible success probability. In the following, we consider only "algebraic" attacks on the parallel version of the issuing protocol. We restrict ourselves to the case $l = 1$; it is easy to prove that if Assumption 4.4.5 holds for $l = 1$ then it also holds for general l. Furthermore, we consider only two parallel executions of the issuing protocol, each

with respect to a different blinding-invariant number; the argument can easily be generalized. Finally, we assume that $\widehat{\mathcal{V}}_0$ cannot compute with non-negligible probability of success a non-trivial representation of 1 with respect to (g_0, g_1, h_0). (It is easy to prove that $\log_{g_0} h_0$ and $\log_{g_0} g_1$ do not leak.)

Denote by x_{10} and $x_{11} \neq x_{10} \bmod q$, respectively, the putative blinding-invariant parts corresponding to each of the parallel two executions of the certificate issuing protocol. The goal of $\widehat{\mathcal{V}}_0$ is to obtain a certified public key $h' \neq 1, (c_0, r_0)$ and a secret key (β_0, β_1) for h' such that

$$\beta_0 \neq x_{10}\beta_1 \bmod q \quad \text{and} \quad \beta_0 \neq x_{11}\beta_1 \bmod q.$$

Knowing (c_0, r_0) such that

$$c_0 = \mathcal{H}_{q,g_0}(h', g_0^{c_0}(h')^{r_0})$$

is equivalent to knowing (a_0, r_0) such that

$$g_0^{\mathcal{H}_{q,g_0}(h',a_0)}(h')^{r_0} = a_0.$$

Therefore, the attack target is a triple (β_0, β_1), $h' = g_1^{\beta_0} h_0^{\beta_1}$, (a_0, r_0) such that $g_0^{c_0}(g_1^{\beta_0} h_0^{\beta_1})^{r_0} = a_0$, where c_0 denotes $\mathcal{H}_{q,g_0}(g_1^{\beta_0} h_0^{\beta_1}, a_0)$. Raising the verification relations for each of the two protocol executions to the powers γ_0 and γ_1, respectively, and multiplying the results, we obtain

$$g_0^{\gamma_0 c_{00} + \gamma_1 c_{01}} h_0^{\gamma_0 r_{00} + \gamma_1 r_{01}} g_1^{\gamma_0 x_{10} r_{00} + \gamma_1 x_{11} r_{01}} = a_{00}^{\gamma_0} a_{01}^{\gamma_1}.$$

$\widehat{\mathcal{V}}_0$ must determine a pair β_0, β_1, and numbers γ_0, γ_1, c_{00}, c_{01} for which the information provided by $\mathcal{P}_0$ can be combined into a pair (a_0, r_0) such that

$$g_0^{c_0} g_1^{\beta_0 r_0} h_0^{\beta_1 r_0} = a_0,$$

where $c_0 = \mathcal{H}_{q,g_0}(g_1^{\beta_0} h_0^{\beta_1}, a_0)$.

Assume first that $\widehat{\mathcal{V}}_0$ computes $a_0 := a_{00}^{\gamma_0} a_{01}^{\gamma_1}$, for $\gamma_0, \gamma_1 \neq 0 \bmod q$ that need not be explicitly known at the time c_{00} and c_{01} have to be provided. $\widehat{\mathcal{V}}_0$ must ensure that

$$g_0^{\gamma_0 c_{00} + \gamma_1 c_{01}} h_0^{\gamma_0 r_{00} + \gamma_1 r_{01}} g_1^{\gamma_0 x_{10} r_{00} + \gamma_1 x_{11} r_{01}} = g_0^{c_0} g_1^{\beta_0 r_0} h_0^{\beta_1 r_0}.$$

Assume furthermore that γ_0 and γ_1 are computable by $\widehat{\mathcal{V}}_0$ once the attack has been completed successfully (a plausible assumption given the algebraic nature of the attack). It follows from the assumption that $\widehat{\mathcal{V}}_0$ cannot compute a non-trivial representation of 1 with respect to (g_0, g_1, h_0) that $\widehat{\mathcal{V}}_0$ has to (implicitly) solve the following three relations,

$$\begin{aligned}
&\gamma_0 x_{10} r_{00} + \gamma_1 x_{11} r_{01} = \beta_0 r_0 \bmod q,\\
&\gamma_0 r_{00} + \gamma_1 r_{01} = \beta_1 r_0 \bmod q,\\
&\gamma_0 c_{00} + \gamma_1 c_{01} = c_0 \bmod q,
\end{aligned}$$

for $(\gamma_0, \gamma_1, \beta_0, \beta_1, c_{00}, c_{01})$ and r_0. It seems that $(\gamma_0, \gamma_1, \beta_0, \beta_1, c_{00}, c_{01})$ must be committed to before r_{00} and r_{01} are provided; only r_0 can be computed afterwards. Since r_0 can be computed by $\widehat{\mathcal{V}}_0$ after r_{00} and r_{01} have been received, it may seem that there are many workable choices for $\gamma_0, \gamma_1, \beta_0, \beta_1$. This is not true, however, since $\widehat{\mathcal{V}}_0$ has to solve, in terms of $\gamma_0, \gamma_1, \beta_0, \beta_1$, a single relation that does not involve r_0 but does involve r_{00} and r_{01}. Multiplying both sides of $\gamma_0 r_{00} + \gamma_1 r_{01} = \beta_1 r_0 \bmod q$ by $\beta_0/\beta_1 \bmod q$, and subtracting the result from $\gamma_0 x_{10} r_{00} + \gamma_1 x_{11} r_{01} = \beta_0 r_0 \bmod q$, we get

$$(\gamma_0 (x_{10} - \beta_0/\beta_1)) r_{00} + (\gamma_1 (x_{11} - \beta_0/\beta_1)) r_{01} = 0 \bmod q.$$

Because r_{00} and r_{01} cannot be anticipated, and because $\beta_0/\beta_1 \bmod q$ cannot be equal to both x_{10} and x_{11}, the only workable non-zero choices for $\gamma_0, \gamma_1, \beta_0, \beta_1$ seem to be to take $\gamma_0 = A_{00} r_{00}^{-1} \bmod q$ and $\gamma_1 = A_{01} r_{01}^{-1} \bmod q$, or $\gamma_0 = A_{00} r_{01} \bmod q$ and $\gamma_1 = A_{01} r_{00} \bmod q$, for some suitable constants A_{00} and A_{01} that may depend on β_0 and β_1. To argue that $\widehat{\mathcal{V}}_0$ cannot compute $a := a_{00}^{\gamma_0} a_{01}^{\gamma_1}$ for such a choice for γ_0, γ_1, we focus on the third relation, $\gamma_0 c_{00} + \gamma_1 c_{01} = c_0 \bmod q$. (After all, it is not completely inconceivable that a can be computed in this way before r_{00} and r_{01} become known, since r_{00} and r_{01} are known to satisfy the two verification relations.) Even if a could be computed, the fact that c_0 is the outcome of a sufficiently strong one-way hash function applied to a_0 implies that its value cannot be expressed in terms of r_{00} and r_{01}. (Note that $c_0 = \mathcal{H}_{q,g_0}(g_1^{\beta_0} h_0^{\beta_1}, a_{00}^{\gamma_0} a_{01}^{\gamma_1})$ should imply, by virtue of the strength of the hash-function, that c_0 cannot be chosen as an algebraic function of $\beta_0, \beta_1, \gamma_0, \gamma_1$; this trivially holds in the random oracle model.) Consequently, $\gamma_0 c_{00} + \gamma_1 c_{01} = c_0 \bmod q$ can only be solved for values c_{00} and c_{01} that are expressed in terms of r_{00}, r_{01}. Because c_{00} and c_{01} have to be provided by $\widehat{\mathcal{V}}_0$ before r_{00} and r_{01} become known, workable choices for γ_0 and γ_1 should be infeasible.

We assumed in this argument that $\widehat{\mathcal{V}}_0$ computes $a_0 := a_{00}^{\gamma_0} a_{01}^{\gamma_1}$. The information contained in

$$g_0^{\gamma_0 c_{00} + \gamma_1 c_{01}} h_0^{\gamma_0 r_{00} + \gamma_1 r_{01}} g_1^{\gamma_0 x_{10} r_{00} + \gamma_1 x_{11} r_{01}}$$

can be combined into $g_0^{c_0} (g_1^{\beta_0} h_0^{\beta_1})^{r_0}$ in a more general way. The most general form seems to be $a_0 := a_{00}^{\gamma_0} a_{01}^{\gamma_1} g_1^{\delta_0} g_1^{\delta_1} h_0^{\delta_2}$ for smart choices for $\delta_0, \delta_1, \delta_2$. Assuming again that it is infeasible to compute with non-negligible probability of success a non-trivial representation of 1 with respect to (g_0, g_1, h_0), we can derive three relations similar to those previously displayed. From the first two of these we can again derive one relation that involves r_{00} and r_{01} but not r_0, and that relation must be solved (implicitly) for $\gamma_0, \gamma_1, \beta_0, \beta_1$. The only way to arrive at a relation in which γ_0 and γ_1 are not expressions in terms of r_{00}, r_{01} (for which the preceding argument applies) seems to be to choose δ_0, δ_1 such that $\gamma_0 r_{00} + \gamma_1 r_{01} + \delta_1 = \beta_0 r_0 \bmod q$ and $\gamma_0 x_{10} r_{00} + \gamma_1 x_{11} r_{01} + \delta_0 = \beta_1 r_0 \bmod q$ are linearly dependent in r_0; in that case r_0 cannot be made to drop out of the equations. Such choices for δ_0, δ_1 seem to require expressions in terms of r_{00} and r_{01} that cannot be anticipated.

This completes our argument as to why Assumption 4.4.5 should hold. Unfortunately, it is unclear how to prove Assumption 4.4.5, even in the random oracle model.

A similar immunization applies to DLREP-based scheme II; we simply swap the position of the challenge in the verification relation with that of one of the two responses. The immunization technique does not apply to RSAREP-based scheme I, since it does not have a response that appears as an exponent in the verification relation. It can be applied to RSAREP-based scheme II, though, but not without a twist. Redefine a certificate of $\mathcal{P}_0$ on $h' \in \mathbb{Z}_n^*$ to be a triple, $(c_0', r_0', r_1') \in \mathbb{Z}_s \times \mathbb{Z}_n^* \times \mathbb{Z}_v$ such that

$$c_0' = \mathcal{H}_{n,v}(h', (r_0')^v f^{c_0'} (h')^{-r_1'}).$$

$\mathcal{V}_0$'s secret key now is an RSA-representation of h' with respect to $(g_1, \ldots, g_l, h_0, v)$. In the modified issuing protocol, $\mathcal{V}_0$ can blind $h = \prod_{i=1}^{l} g_i^{x_i}$ to $h' = (h_0 h)^\beta \alpha_1^v$, for arbitrary $\beta \in \mathbb{Z}_v$ and $\alpha_1 \in \mathbb{Z}_n^*$. While in an application this general blinding form must be taken into account, for unlinkability it suffices for $\mathcal{V}_0$ to simply fix $\beta = 1$, say, and use a random α_1. The resulting issuing protocol is depicted in Figure 4.8. (Alternatively, $\mathcal{P}_0$ can perform this protocol without using the factorization of n, similar as described in Section 4.2.3. The protocol can be converted into a two-move protocol in the manner pointed out in Section 4.2.3.) Now, from h' one cannot infer that $\beta \neq 0 \bmod v$, yet this choice must be prevented. We can get around this by having $\mathcal{V}_0$ in the showing protocol demonstrate that $\beta \neq 0 \bmod v$, as part of the formula it is demonstrating: see Section 5.1.1 for details.

4.5 Other certificate schemes

The certificate schemes in Sections 4.2 and 4.4 are all based on the digital signature schemes discussed in Sections 2.5.3 and 2.5.4. As in many areas of cryptography, it is of interest to have alternatives based on different underlying assumptions, instead of placing all bets on one horse. In this section we describe two such alternatives. Both alternatives are believed to be secure even when protocol executions with respect to different attribute tuples are arbitrarily interleaved.

4.5.1 DSA-like certificates

The system parameter generation and the key set-up for this scheme are the same as for DLREP-based scheme I. It is preferable that $\mathcal{H}(\cdot)$ do not map arguments to zero, but since this event should have negligible probability anyway there is no need to make an assumption to this effect.

For $a \in G_q$, let $\overline{a}$ denote $a \bmod q$. In the DSA [277], a signature standard originally proposed in 1994 by the U.S. National Institute of Standards and Technology, a signature on a message m with respect to a public key $h_0 = g_0^{x_0}$ is a pair

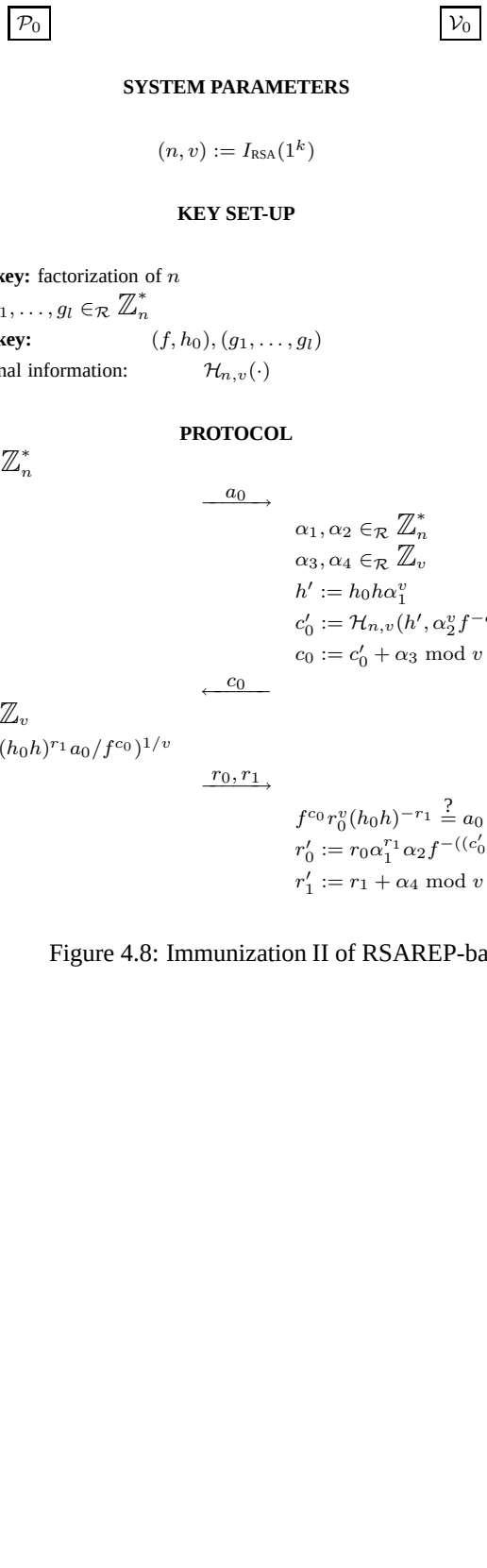

Figure 4.8: Immunization II of RSAREP-based scheme II.

$(\overline{a_0}, r_0) \in \mathbb{Z}_q \times \mathbb{Z}_q$ such that

$$(g_0^{\mathcal{H}_{q,g_0}(m)/r_0} h_0^{\overline{a_0}/r_0}) \bmod q = \overline{a_0}.$$

The DSA makes the following specific choices: G_q is constructed using the subgroup construction, q is a 160-bit prime, and $\mathcal{H}_{q,g_0}(\cdot)$ is set equal to SHA-I [276].

We modify the DSA scheme by applying a cyclic left shift to the role of the exponents, $(\mathcal{H}_{q,g_0}(m), r_0, \overline{a_0})$, in the DSA verification relation.[6] A certificate of $\mathcal{P}_0$ on a public key $h' \neq 1$ is defined to be a pair $(\overline{a'_0}, r'_0) \in \mathbb{Z}_q \times \mathbb{Z}_q$ such that

$$(g_0^{\overline{a'_0}/c'_0} (h')^{r'_0/c'_0}) \bmod q = \overline{a'_0},$$

where $c'_0 = \mathcal{H}_{g,g_0}(h_0, \overline{a'_0})$. The presence of $\overline{a'_0}$ in $\mathcal{H}_{q,g_0}(h_0, \overline{a'_0})$ is not mandatory, but is believed preferable. The secret key of $\mathcal{V}_0$ is a DL-representation of h' with respect to $(g_1, \ldots, g_l, h_0)$.

Let h denote $\prod_{i=1}^{l} g_i^{x_i}$. The issuing protocol is as follows:

Step 1. $\mathcal{P}_0$ generates a random number $w_0 \in \mathbb{Z}_q$, and sends $a_0 := g_0^{w_0}$ to $\mathcal{V}_0$.

Step 2. $\mathcal{V}_0$ generates a random number $\alpha_1 \in \mathbb{Z}_q^*$ and two random numbers $\alpha_2, \alpha_3 \in \mathbb{Z}_q$. It computes $h' := (h_0 h)^{\alpha_1}$, $a'_0 := a_0^{\alpha_2} (h_0 h)^{\alpha_3}$, and $c'_0 := \mathcal{H}_{q,g_0}(h', \overline{a'_0})$. Finally, $\mathcal{V}_0$ sends $c_0 := c'_0 \alpha_2 \overline{a_0} \overline{a'_0}^{-1} \bmod q$ to $\mathcal{P}_0$.

Step 3. $\mathcal{P}_0$ sends $r_0 := (x_0 + \sum_{i=1}^{l} x_i y_i)^{-1} (c_0 w_0 - \overline{a_0}) \bmod q$ to $\mathcal{V}_0$. (To guarantee that $\mathcal{P}_0$ can always perform Step 3, $(x_1, \ldots, x_l)$ must satisfy $(x_0 + \sum_{i=1}^{l} x_i y_i) \neq 0 \bmod q$.)

$\mathcal{V}_0$ accepts if and only if $g_0^{\overline{a_0}/c_0} (h_0 h)^{r_0/c_0} = a_0$. If this verification holds, $\mathcal{V}_0$ computes $r'_0 := \alpha_1^{-1} (r_0 \overline{a_0}^{-1} \overline{a'_0} + c'_0 \alpha_3) \bmod q$. The resulting scheme is depicted in Figure 4.9.

It is easy to verify that the protocol is a proof of knowledge (the probability that the inverses of $\overline{a_0}$ and $\overline{a'_0}$ are defined is overwhelming) and that $(\overline{a'_0}, r'_0)$ is a secret-key certificate of $\mathcal{P}_0$ on h'. As with the schemes in Section 4.4.2, it is unclear how to reduce the unforgeability of the underlying signature scheme to that of the new scheme, but unforgeability is believed to hold nevertheless. Furthermore, if $\mathcal{V}_0$ follows the issuing protocol and accepts then it obtains a certified key pair comprising a perfectly blinded certified public key.

Assumption 4.4.5 should apply here as well. Following the argument in Section 4.4.2, we arrive at three relations that differ only in that $\gamma_0 c_{00} + \gamma_1 c_{01} = c_0 \bmod q$ is replaced by $\gamma_0 \overline{a_{00}} + \gamma_1 \overline{a_{01}} = a_0 \bmod q$, where $a_0 = a_{00}^{\gamma_0 c_{00}} a_{01}^{\gamma_1 c_{01}}$.

[6] Camenisch, Piveteau, and Stadler [72] applied another shift of the exponents, in order to construct an ordinary DSA-like blind signature scheme in Chaum's sense. Their shift does not give rise to a restrictive blind certificate scheme that is secure in parallel mode.

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(q, g_0) := I_{\mathrm{DL}}(1^k)$	
	KEY SET-UP	
$x_0 := D_{\mathrm{DL}}(q, g_0)$		
$y_1, \ldots, y_l \in_{\mathcal{R}} \mathbb{Z}_q$		
Secret key: $x_0, (y_1, \ldots, y_l)$		
$h_0 := g_0^{x_0}$		
$g_i := g_0^{y_i}\ \forall i \in \{1, \ldots, l\}$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{q,g_0}(\cdot)$	
	PROTOCOL	
$w_0 \in_{\mathcal{R}} \mathbb{Z}_q$		
$a_0 := g_0^{w_0}$		
	$\xrightarrow{a_0}$	
		$\alpha_1 \in_{\mathcal{R}} \mathbb{Z}_q^*$
		$\alpha_2, \alpha_3 \in_{\mathcal{R}} \mathbb{Z}_q$
		$h' := (h_0 h)^{\alpha_1}$
		$a_0' := a_0^{\alpha_2} (h_0 h)^{\alpha_3}$
		$c_0' := \mathcal{H}_{q,g_0}(h', \overline{a_0'})$
		$c_0 := c_0' \alpha_2 \overline{a_0} \overline{a_0'}^{-1} \bmod q$
	$\xleftarrow{c_0}$	
$r_0 := (x_0 + \sum_{i=1}^{l} x_i y_i)^{-1} (c_0 w_0 - \overline{a_0}) \bmod q$		
	$\xrightarrow{r_0}$	
		$g_0^{\overline{a_0}/c_0} (h_0 h)^{r_0/c_0} \stackrel{?}{=} a_0$
		$r_0' := \alpha_1^{-1} (r_0 \overline{a_0}^{-1} \overline{a_0'} + c_0' \alpha_3) \bmod q$

Figure 4.9: DSA-like scheme.

The latter relation seems even harder to handle, although it is unclear how to prove this intuition.

A drawback of this DSA-like certificate scheme over the immunized DLREP-based scheme I in Section 4.4.2 is that $\mathcal{V}_0$ cannot precompute the exponentiation $a_0^{\alpha_2}$ in Step 2.

Although the DSA makes the explicit choice $G_q \subset \mathbb{Z}_p^*$, an elliptic curve implementation may be used instead. Indeed, the elliptic curve analog of the DSA, called the ECDSA, has been adopted as a standard by ISO, by ANSI, by IEEE, and by NIST; see Johnson and Menezes [223] for an overview. Since numbers in G_q are represented by coordinate pairs, and base numbers occur also as exponents, a secure mapping is needed from base numbers to numbers in $\mathbb{Z}_q$. One solution is to use the first coordinate; this is the approach taken in the ECDSA.

4.5.2 Certificates based on Chaum-Pedersen signatures

The following scheme differs from all the other certificate schemes in this chapter in that public-key certificates are issued, not secret-key certificates. The system parameter generation and the key set-up for this scheme are the same as for DLREP-based scheme I.

Chaum and Pedersen [109] presented a digital signature scheme that can be viewed as an entangled application of two Schnorr protocol executions; it is an optimization of a protocol due to Chaum, Evertse, and van de Graaf [108, Protocol 4]. A signature on a message m with respect to a public key $h_0 = g_0^{x_0}$ is a triple $(z, c_0, r_0) \in G_q \times \mathbb{Z}_s \times \mathbb{Z}_q$ such that

$$c_0 = \mathcal{H}_{q,g_0}(m, z, g^{r_0} h_0^{-c_0}, m^{r_0} z^{-c_0}).$$

Chaum and Pedersen defined G_q using the subgroup construction, but an elliptic curve implementation is permitted as well. They also described how to construct a blind issuing protocol for their signatures. The interactive Chaum-Pedersen protocol is a Fiat-Shamir type proof of knowledge of both $\log_{g_0} h_0$ and $\log_m z$ that also demonstrates that these two discrete logarithms are equal. For the purpose of ordinary blind signatures, though, the Chaum-Pedersen scheme does not have advantages over the Schnorr signature scheme.

The situation is different for the restrictive blind certificate scheme that we will now construct from their signature scheme by applying our techniques. Define a certificate of $\mathcal{P}_0$ on a public key $h' \neq 1$ to be a triple, $(z', c_0', r_0') \in G_q \times \mathbb{Z}_s \times \mathbb{Z}_q$ such that

$$c_0' = \mathcal{H}_{q,g_0}(h', z', g_0^{r_0'} h_0^{-c_0'}, (h')^{r_0'} (z')^{-c_0'}).$$

The secret key of $\mathcal{V}_0$ is a DL-representation $(x_1, \ldots, x_l, \alpha_1)$ of h' with respect to $(g_1, \ldots, g_l, h_0)$, with $\mathcal{P}_0$ encoding $x_1, \ldots, x_l \in \mathbb{Z}_q$ into $\mathcal{V}_0$'s secret key. As before, let h denote $\prod_{i=1}^{l} g_i^{x_i}$. $\mathcal{V}_0$ this time needs to know $z := (h_0 h)^{x_0}$. Hereto $\mathcal{P}_0$ must

either compute z for $\mathcal{V}_0$ or publish $(g_1^{x_0}, \ldots, g_l^{x_0}, h_0^{x_0})$ along with its public key; in the latter case, $\mathcal{V}_0$ can compute z by itself.

The issuing protocol is as follows:

Step 1. $\mathcal{P}_0$ generates a random number $w_0 \in \mathbb{Z}_q$, and sends $a_0 := g_0^{w_0}$ and $b_0 := (h_0 h)^{w_0}$ to $\mathcal{V}_0$.

Step 2. $\mathcal{V}_0$ generates a random number $\alpha_1 \in \mathbb{Z}_q^*$ and two random numbers $\alpha_2, \alpha_3 \in \mathbb{Z}_q$. $\mathcal{V}_0$ computes $h' := (h_0 h)^{\alpha_1}$, $z' := z^{\alpha_1}$, $a_0' := h_0^{\alpha_2} g_0^{\alpha_3} a_0$, $b_0' := (z')^{\alpha_2} (h')^{\alpha_3} b_0^{\alpha_1}$, $c_0' := \mathcal{H}_{q,g_0}(h', z', a_0', b_0')$, and sends $c_0 := c_0' + \alpha_2 \bmod q$ to $\mathcal{P}_0$.

Step 3. $\mathcal{P}_0$ sends $r_0 := c_0 x_0 + w_0 \bmod q$ to $\mathcal{V}_0$.

$\mathcal{V}_0$ accepts if and only if $g_0^{r_0} h_0^{-c_0} = a_0$ and $(h_0 h)^{r_0} z^{-c_0} = b_0$. If this verification holds, then $\mathcal{V}_0$ computes $r_0' := r_0 + \alpha_3 \bmod q$. (Alternatively, $\mathcal{V}_0$ first computes r_0' and then checks whether $(g_0 h')^{r_0'} (h_0 z')^{-c_0'} = a_0' b_0'$.) The resulting scheme is depicted in Figure 4.10.

It is easy to verify that the protocol is a Fiat-Shamir type proof of knowledge, and that (z', r_0', c_0') is a certificate of $\mathcal{P}_0$ on h' if $\overline{\mathcal{V}}_0$ accepts. The witness-hiding property can be argued in the same manner as with the Schnorr proof of knowledge, but no proof is known. The unforgeability of certificates follows directly from the unforgeability of blind Chaum-Pedersen signatures and the fact that $\widehat{\mathcal{V}}_0$ cannot know two different DL-representations of the same public key. Proving the latter fact is trivial: $\mathcal{P}_0$ can perform the protocol without knowing a non-trivial DL-representation of 1 with respect to $(g_1, \ldots, g_l, h_0)$, and in particular it may generate $g_1, \ldots, g_l$ at random.

Excluding public keys h' equal to 1, it can be shown that $\overline{\mathcal{V}}_0$ obtains a certified key pair comprising a perfectly blinded certified public key. Finally, Assumption 4.4.5 is believed to apply.

This certificate scheme has several drawbacks in comparison to DLREP-based schemes I and II and their immunization described in Section 4.4.2:

- Certificates are larger and more costly to verify.
- $\mathcal{V}_0$'s computation of $b_0^{\alpha_1}$ in Step 2 of the issuing protocol cannot be preprocessed.
- It is not clear how to prove the property of restrictive blinding even when only sequential protocol executions involving the same attribute tuple are considered.
- The scheme is a public-key certificate scheme; certified public keys cannot be simulated.

$\boxed{\mathcal{P}_0}$		$\boxed{\mathcal{V}_0}$
	SYSTEM PARAMETERS	
	$(q, g_0) := I_{\text{DL}}(1^k)$	
	KEY SET-UP	
$x_0 := D_{\text{DL}}(q, g_0)$		
$y_1, \ldots, y_l \in_{\mathcal{R}} \mathbb{Z}_q$		
Secret key: $x_0, (y_1, \ldots, y_l)$		
$h_0 := g_0^{x_0}$		
$g_1, \ldots, g_l \in_{\mathcal{R}} G_q$		
Public key:	$h_0, (g_1, \ldots, g_l)$	
Additional information:	$\mathcal{H}_{q,g_0}(\cdot), z := (h_0 h)^{x_0}$	
	PROTOCOL	
$w_0 \in_{\mathcal{R}} \mathbb{Z}_q$		
$a_0 := g_0^{w_0}$		
$b_0 := (h_0 h)^{w_0}$		
	$\xrightarrow{a_0, b_0}$	
		$\alpha_1 \in \mathbb{Z}_q^*$
		$\alpha_2, \alpha_3 \in \mathbb{Z}_q$
		$h' := (h_0 h)^{\alpha_1}$
		$z' := z^{\alpha_1}$
		$a'_0 := h_0^{\alpha_2} g_0^{\alpha_3} a_0$
		$b'_0 := (z')^{\alpha_2} (h')^{\alpha_3} b_0^{\alpha_1}$
		$c'_0 := \mathcal{H}_{q,g_0}(h', z', a'_0, b'_0)$
		$c_0 := c'_0 + \alpha_2 \bmod q$
	$\xleftarrow{c_0}$	
$r_0 := c_0 x_0 + w_0 \bmod q$		
	$\xrightarrow{r_0}$	
		$g_0^{r_0} h_0^{-c_0} \stackrel{?}{=} a_0$
		$(h_0 h)^{r_0} z^{-c_0} \stackrel{?}{=} b_0$
		$r'_0 := r_0 + \alpha_3 \bmod q$

Figure 4.10: Scheme based on Chaum-Pedersen signatures.

On the upside, $\mathcal{P}_0$ can perform the issuing protocol without knowing $(x_1, \ldots, x_l)$. This has several advantages:

- $\mathcal{P}_0$ can certify attributes without needing to know them. Hereto $\mathcal{V}_0$ forms h as its commitment to attributes $x_1, \ldots, x_{l-1}$ (the number x_l must be generated at random to unconditionally hide the $l-1$ attributes) and at the start of the issuing protocol presents this to $\mathcal{P}_0$ together with a proof of knowledge of a DL-representation with respect to $(g_1, \ldots, g_l)$.

- More generally, $\mathcal{P}_0$ can certify attributes of which it knows no more than a property demonstrated by $\mathcal{V}_0$. $\mathcal{V}_0$ can demonstrate this property when presenting h by using the showing protocol techniques, for instance by sending along a signed proof.

- $\mathcal{P}_0$ can recertify previously certified attributes without knowing their values; see Section 5.2.1 for details. (It cannot update their values, though; for this an RSAREP-based issuing protocol is needed.)

- It is possible to protect against framing attempts by parties with unlimited computing resources; see Section 5.5.3 for details.

Whether these advantages are desirable depends on the application at hand.

Another advantage is that the delegation strategy (see Section 2.6) is excluded altogether, because public-key certificates are used. This is not the case for the certificate issuing schemes based on secret-key certificates, as Section 5.1.2 will show.

4.6 Bibliographic notes

The notion of restrictive blinding in Section 4.1 originates from Brands [46, 48]. Definition 4.1.1 appears here for the first time.

The four certificate schemes in Section 4.2 originate from Brands [54]. The case $l = 2$ of DLREP-based scheme I was introduced by Brands [49] for the purpose of withdrawing electronic coins with embedded identifiers in an off-line electronic cash system. The case $l = 1$ of RSAREP-based scheme I was analyzed by Brands [51]. The security proofs presented in Section 4.3.3 are stronger than those in [49, 51], in that the reductions are based on any invulnerable instance generator instead of one specific distribution.

The immunization technique described in Section 4.4.1 is due to Brands [50]. The immunized DLREP-based schemes in Section 4.4.2 are a generalization of a withdrawal protocol devised by Schoenmakers [341] in the context of electronic cash.

The immunization described in Section 4.4.2 of RSAREP-based scheme II, which is based on a new twist, appears here for the first time; previously, the immunization technique of Schoenmakers was believed not to apply to RSAREP-based schemes.

The DSA-based certificate scheme described in Section 4.5.1 has not previously appeared in the academic literature.

Brands [46, 48] introduced the special case $l = 2$ of the scheme in Section 4.5.2, for the purpose of designing an off-line electronic cash scheme. Cramer and Pedersen [122] subsequently used this scheme to modify a protocol by Chaum and Pedersen [109]; the slightly more general form used by them was already considered by Brands [46, page 27 & 28]. Radu, Govaerts, and Vandewalle [316] proposed a variation to make the scheme provably witness-hiding, but their variation does not improve the overall security of the scheme.