Chapter 3

Showing Protocols with Selective Disclosure

In this chapter we present highly flexible and practical showing protocol techniques that enable the holder of an arbitrary number of attributes to selectively disclose properties about them; any other information remains unconditionally hidden. All the techniques can be based on the DLREP function as well as on the RSAREP function. The demonstrations can take several forms, including zero-knowledge proofs and signed proofs. Signed proofs are provably unforgeable in the random oracle model under the mere assumption that there exists an invulnerable instance generator for the DL function or the RSA function. We do not yet make the connection with digital certificate issuing protocols; this will be the topic of Chapters 4 and 5.

3.1 Introduction

Consider a polynomial-time prover $\mathcal{P}$ that has committed, by means of a commitment function, to one or more attributes that are (represented by) elements of a finite ring. $\mathcal{P}$ is to demonstrate to a verifier $\mathcal{V}$ that its attributes satisfy a satisfiable formula from proposition logic, where the atomic propositions are relations that are linear in the attributes. By way of example, let x_1, x_2, x_3 denote $\mathcal{P}$'s attributes, and consider the formula

$$\Big((F_1 \quad \textsf{AND} \quad F_2) \quad \textsf{OR} \quad (\textsf{NOT}\, F_3 \quad \textsf{AND} \quad F_4)\Big) \quad \textsf{AND} \quad \textsf{NOT}\, F_5 \qquad (3.1)$$

where

$$\begin{aligned} F_1 &= (x_1 + 2x_2 - 10x_3 = 13) \\ F_2 &= (x_2 - 4x_3 = 5) \end{aligned}$$

$$\begin{aligned} F_3 &= (x_1 + 3x_2 + 5x_3 = 7) \\ F_4 &= (3x_1 + 10x_2 + 18x_3 = 23) \\ F_5 &= (x_1 - 8x_2 + 11x_3 = 5) \end{aligned}$$

We require that the computations of $\overline{\mathcal{V}}$ when interacting with $\mathcal{P}$ can be performed in polynomial time, but $\mathcal{V}$ is given infinite computing power in its attempts to learn additional information about $\mathcal{P}$'s attributes. The goal is to ensure that $\mathcal{P}$ does not reveal to $\widetilde{\mathcal{V}}$ any information about its attributes beyond the validity of the formula.

Regardless of the commitment function used, a constant-round zero-knowledge argument for this task can be constructed by reducing the formula to an instance of the NP-complete language Directed Hamiltonian Cycle and applying a protocol due to Feige and Shamir [170]. Alternatively, the formula can be reduced to an instance of the NP-complete language SAT and then subjected to a slightly more efficient protocol due to Bellare, Jakobsson, and Yung [24]. The resulting protocols are highly impractical, though, because statements must be encoded into Boolean circuits and auxiliary commitments must be used for each gate.

In the remainder of this chapter we show that truly practical techniques exist when $\mathcal{P}$ commits to its attributes in a special manner. Assuming parameter sizes sufficient to guarantee long-term security, the new techniques allow $\mathcal{P}$ to non-interactively demonstrate example formula (3.1) by sending a mere 275 bytes to $\mathcal{V}$. Forming and verifying the proof both require fewer than 940 modular multiplications of numbers in G_q or $\mathbb{Z}_n^*$.

In the next section we describe how $\mathcal{P}$ should commit. We then introduce techniques for demonstrating formulae of special forms. Finally, in Section 3.6 we show how to combine these techniques to demonstrate arbitrary Boolean formulae.

3.2 How to commit

Our proof techniques require $\mathcal{P}$ to commit to its attributes by means of either the DLREP function or the RSAREP function:

- To base the security on the hardness of inverting the DL function, the attributes must all be (represented by) numbers in $\mathbb{Z}_q$, and $\mathcal{P}$ computes $h := \prod_{i=1}^{l} g_i^{x_i}$. The system parameters and $\mathcal{P}$'s secret key must be generated in accordance with Construction 2.3.2, based on any invulnerable instance generator for the DL function. Recall from Construction 2.3.2 that $(x_l, \ldots, x_{l-1})$ may have an arbitrary distribution, but $\mathcal{P}$ must generate x_l at random from $\mathbb{Z}_q$.
- To base the security on the hardness of inverting the RSA function, the attributes must all be (represented by) numbers in $\mathbb{Z}_v$, and $\mathcal{P}$ computes $h := \prod_{i=1}^{l} g_i^{x_i} x_{l+1}^v$, where $x_{l+1} \in \mathbb{Z}_n^*$. The system parameters and $\mathcal{P}$'s secret key must be generated in accordance with Construction 2.3.4, based on any

invulnerable instance generator for the RSA function. Recall from Construction 2.3.4 that $(x_1, \ldots, x_l)$ may have an arbitrary distribution, but $\mathcal{P}$ must generate x_{l+1} at random from $\mathbb{Z}_n^*$.

In either case, h is referred to as the public key of $\mathcal{P}$. Recall that l may be polynomial in the security parameter.

In case the security is based on the RSA function, there is a clean separation between the role of the attributes and that of x_{l+1}. This is not the case when the DL function is used, because one of the attributes must be chosen at random, and it will rarely make sense in practice to demonstrate a property about a random number. Nevertheless, the distinction is merely a notational one. If one insists on allowing $(x_1, \ldots, x_l)$ to have an arbitrary distribution, then $\mathcal{P}$ should form $h := \prod_{i=1}^{l+1} g_i^{x_i}$, where g_{l+1} is a generator of G_q and $\mathcal{P}$ generates x_{l+1} at random from $\mathbb{Z}_q$. We stick to the former notation, because it sometimes makes sense to demonstrate that the random x_l is unequal to zero; see Section 5.1.1.

As explained in Section 2.4, algorithms I_{DLREP} and I_{RSAREP} must be run by $\mathcal{V}$, by a party trusted by $\mathcal{V}$ and $\mathcal{P}$, or by means of a secure multi-party protocol between $\mathcal{V}$ and $\mathcal{P}$. In any case, it must be ensured that $\widehat{\mathcal{P}}$ cannot know more than one representation of h. On the basis of Propositions 2.3.3 and 2.3.5 it is easily seen how to accomplish this.

To avoid unduly repetition, throughout this chapter we detail our techniques for the case where the DLREP function is used to commit to $\mathcal{P}$'s attributes; for the RSAREP function we clarify only the differences. We also assume from now on that $l \geq 2$ when the DLREP function is used, for obvious reasons.

3.3 Formulae with zero or more "AND" connectives

We first consider the situation in which $\mathcal{P}$ is to demonstrate a satisfiable formula with zero or more "AND" connectives and no other logical connectives.

3.3.1 Technique based on the DLREP function

Without loss of generality, assume that $\mathcal{P}$ is to demonstrate that the DL-representation it knows of h with respect to $(g_1, \ldots, g_l)$ satisfies the following system of $t \geq 0$ independent linear relations:

$$\begin{pmatrix} \alpha_{11} & \ldots & \alpha_{1,l-t} & 1 & 0 & \ldots & 0 \\ \alpha_{21} & \ldots & \alpha_{2,l-t} & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_{t1} & \ldots & \alpha_{t,l-t} & 0 & 0 & \ldots & 1 \end{pmatrix} \begin{pmatrix} x_{\pi(1)} \\ x_{\pi(2)} \\ \vdots \\ x_{\pi(l)} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_t \end{pmatrix} \bmod q. \quad (3.2)$$

The coefficients α_{ij} are elements of $\mathbb{Z}_q$, and $\pi(\cdot)$ is a permutation of $\{1, \ldots, l\}$. Clearly, any satisfiable system of linear relations in $x_1, \ldots, x_l$ can be described in this form: if a system of linear relations contains dependent relations, it can be reduced to a system of independent linear relations, denoted in number by t here; then the matrix of coefficients can be brought into row canonical form, using Gaussian elimination; and, finally, by applying a suitable permutation $\pi(\cdot)$, the columns of the matrix of coefficients can be interchanged, arriving at the system displayed above. (In a practical implementation the latter step may be omitted, but here we need $\pi(\cdot)$ also to enable a generic description and analysis of the technique.)

If $t = l$ then $\mathcal{V}$ can verify the applicability of the formula without communicating with $\mathcal{P}$, by solving for the x_i's and checking that they form a preimage to h. Therefore we may assume that $t < l$.

Representing atomic propositions by linear relations over $\mathbb{Z}_q$, system (3.2) corresponds to the following Boolean formula:

$$\begin{aligned}(b_1 = \alpha_{11}x_{\pi(1)} + \cdots + \alpha_{1,l-t}x_{\pi(l-t)} + x_{\pi(l-t+1)} \bmod q) \ \mathsf{AND}\ \ldots \\ \ldots\ \mathsf{AND}\ (b_t = \alpha_{t1}x_{\pi(1)} + \cdots + \alpha_{t,l-t}x_{\pi(l-t)} + x_{\pi(l)} \bmod q).\end{aligned} \tag{3.3}$$

The special case $t = 0$ corresponds to the "empty" formula, TRUE. Although $\mathcal{P}$'s attributes obviously satisfy this formula, it may certainly make sense to demonstrate it, as will become clear in the next three chapters.

Our technique for demonstrating formula (3.3) is based on the following result.

Proposition 3.3.1. *$\mathcal{P}$ can prove knowledge of a DL-representation of*

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})^{-1}h$$

with respect to

$$\Big(g_{\pi(1)}\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}},\ \ldots,\ g_{\pi(l-t)}\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}}\Big)$$

if and only if it knows a set of attributes that satisfies the formula (3.3).

Proof. If $(x_1, \ldots, x_l)$ satisfies the formula (3.3), then

$$\begin{aligned} h &= \prod_{i=1}^{l} g_{\pi(i)}^{x_{\pi(i)}} \\ &\overset{(*)}{=} (\prod_{i=1}^{l-t} g_{\pi(i)}^{x_{\pi(i)}})(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i - \sum_{j=1}^{l-t}\alpha_{ij}x_{\pi(j)}}) \\ &\overset{(**)}{=} (\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})(g_{\pi(1)}\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}})^{x_{\pi(1)}} \cdots (g_{\pi(l-t)}\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}})^{x_{\pi(l-t)}}, \end{aligned}$$

and so the DL-representation that $\overline{\mathcal{P}}$ can prove knowledge of is $(x_{\pi(1)}, \ldots, x_{\pi(l-t)})$. (We will refer to the marked derivation steps later on.)

To prove the converse, suppose that $\widehat{\mathcal{P}}$ convinces $\overline{\mathcal{V}}$ with non-negligible success probability. According to Definition 2.4.1, there exists a polynomial-time knowledge extractor $\mathcal{K}$ that outputs with non-negligible success probability a DL-representation $(y_1, \ldots, y_{l-t})$. By expanding the relation

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})^{-1} h = \prod_{i=1}^{l-t} (g_{\pi(i)} \prod_{j=1}^{t} g_{\pi(l-t+j)}^{-\alpha_{ji}})^{y_i},$$

it is seen that

$$(y_1, \ldots, y_{l-t}, b_1 - \sum_{j=1}^{l-t} \alpha_{1j} y_j \bmod q, \ldots, b_t - \sum_{j=1}^{l-t} \alpha_{tj} y_j \bmod q)$$

is a DL-representation of h with respect to $(g_{\pi(1)}, \ldots, g_{\pi(l)})$. This must be the DL-representation $(x_{\pi(1)}, \ldots, x_{\pi(l)})$ known to $\mathcal{P}$, because $\mathcal{P}$ must know at least one such DL-representation to perform the proof of knowledge (otherwise $\mathcal{K}$ can be used directly to compute DL-representations), and if this were different then $<\widehat{\mathcal{P}}, \mathcal{K}>$ could be used to invert the DL function (see Proposition 2.3.3).

It remains to check that the DL-representation satisfies the formula (3.3). From the left-hand side of the matrix equation (3.2) it follows, for all $i \in \{1, \ldots, t\}$, that

$$\begin{aligned}\sum_{j=1}^{l-t} \alpha_{ij} x_{\pi(j)} + x_{\pi(l-t+i)} &= \sum_{j=1}^{l-t} \alpha_{ij} y_j + (b_i - \sum_{j=1}^{l-t} \alpha_{ij} y_j) \\ &= b_i \bmod q.\end{aligned}$$

This completes the proof. □

While the expressions appearing in this proposition are somewhat intimidating, the process of deriving them is simple and can easily be performed by pencil and paper using the following three steps:

1. As can be seen in the first part of the proof of Proposition 3.3.1 and in particular in the derivation step marked by $^{(*)}$, we first substitute into $\mathcal{P}$'s commitment $h := \prod_{i=1}^{l} g_i^{x_i}$ the t expressions for $x_{\pi(l-t+1)}, \ldots, x_{\pi(l)}$ that must hold if the formula (3.3) holds true.

2. We then group together the terms that are raised to a constant power, and collect terms for each of the variables $x_{\pi(1)}, \ldots, x_{\pi(l-t)}$; see the derivation step marked by $^{(**)}$.

3. Finally, we divide both sides by the product of all constant powers.

Example 3.3.9 will illustrate the process.

By substitution into Proposition 3.3.1 it is immediately seen that the demonstration of the formula TRUE corresponds to proving knowledge of a DL-representation of h with respect to $(g_1, \ldots, g_l)$. According to the proof of Proposition 3.3.1, this property holds in general.

Corollary 3.3.2. *Regardless of the formula demonstrated by $\mathcal{P}$, the proof of knowledge in Proposition 3.3.1 is also a proof of knowledge of a DL-representation of h with respect to $(g_1, \ldots, g_l)$.*

The importance of this simple result will become clear in Section 5.5.2, where we show how to discourage certificate lending, and in Section 6.3, where we show how to lift the demonstration technique to the smartcard setting.

Consider now a setting in which $\widetilde{\mathcal{V}}$ requests $\mathcal{P}$ to demonstrate a plurality of formulae of the form (3.3), not necessarily all the same. Clearly, with each new formula that is demonstrated for the same h, $\widetilde{\mathcal{V}}$ learns additional information about $\mathcal{P}$'s attributes.

Definition 3.3.3. *In an* adaptively chosen formula attack, *$\widetilde{\mathcal{V}}$ may select at the start of each new protocol execution which formula is to be demonstrated by $\mathcal{P}$. Protocol executions may be arbitrarily interleaved, in any way dictated by $\widetilde{\mathcal{V}}$. In case $\widetilde{\mathcal{V}}$ requests a formula that does not apply to $\mathcal{P}$'s attributes, $\mathcal{P}$ informs $\widetilde{\mathcal{V}}$ of this fact either by not responding before a time-out or by sending a predetermined fixed message; otherwise $\mathcal{P}$ demonstrates the formula to $\widetilde{\mathcal{V}}$. In either case, $\widetilde{\mathcal{V}}$ learns the* status *of the formula, namely whether it is true or false with respect to $\mathcal{P}$'s attributes.*

We are interested in determining whether $\widetilde{\mathcal{V}}$ can learn more about $\mathcal{P}$'s attributes than what can be learned from only the status of the requested formulae and $\widetilde{\mathcal{V}}$'s a priori information (the probability distribution from which $\mathcal{P}$'s attributes have been drawn). That is, do protocol executions leak additional information about $\mathcal{P}$'s attributes?

The following proposition provides a sufficient condition to guarantee that whatever $\widetilde{\mathcal{V}}$ can compute about $\overline{\mathcal{P}}$'s attributes, it can also compute using only its a priori information and the status of the requested formulae.

Proposition 3.3.4. *Let $\overline{\mathcal{P}}$ only demonstrate formulae in which x_l does not appear, using a proof of knowledge as described in Proposition 3.3.1 with the property that it is statistically witness-indistinguishable. For any distribution of $(x_1, \ldots, x_{l-1})$, whatever information $\widetilde{\mathcal{V}}$ in an adaptively chosen formula attack can compute about $(x_1, \ldots, x_{l-1})$ can also be computed using merely its a priori information and the status of the formulae requested.*

Proof. Without loss of generality we concentrate on the formulae that $\mathcal{P}$ demonstrates by means of a proof of knowledge. Consider first the demonstration of a single formula. If $x_{\pi(j)}$ does not appear in the formula, for some $j \in \{1, \ldots, l-t\}$, then the j-th column in the matrix on the left-hand side of system (3.2) contains all zeros, and it follows that

$$\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{ij}} = 1.$$

As a consequence, in Proposition 3.3.1 the number $g_{\pi(j)}$ appears separately in the tuple with respect to which $\mathcal{P}$ proves knowledge of a DL-representation. Since in our case x_l does not appear in the formulae demonstrated, it follows that, in a single formula demonstration, $\mathcal{P}$ proves knowledge of a DL-representation $(y_1, \ldots, y_j, x_l)$ of a number in G_q with respect to a tuple $(g_1^*, \ldots, g_j^*, g_l)$, for some integer $j \geq 0$ and numbers $g_1^*, \ldots, g_j^*$ in G_q that depend on the formula demonstrated. The set $\{y_1, \ldots, y_j\}$ is a subset of $\{x_1, \ldots, x_{l-1}\}$. Since g_l is a generator of G_q, for any tuple $(y_1, \ldots, y_j) \in (\mathbb{Z}_q)^j$ there is exactly one x_l such that $(y_1, \ldots, y_j, x_l)$ is the DL-representation that $\mathcal{P}$ proves knowledge of. Because the proof of knowledge is witness-indistinguishable, and x_l has been chosen at random by $\mathcal{P}$, it follows that no information is revealed about $y_1, \ldots, y_j$. Consequently, no information is leaked about $(x_1, \ldots, x_{l-1})$ beyond the status of the formula.

To complete the proof, we apply Proposition 2.4.5, according to which the (arbitrarily interleaved) demonstration of many formulae, each by means of a witness-indistinguishable proof, is also witness-indistinguishable. □

The witness-indistinguishability condition in Proposition 3.3.4 is not only sufficient, but also necessary. For example, if $l \geq 3$ and $t < l - 1$, and $\mathcal{P}$ performs the proof of knowledge by disclosing $x_{\pi(l-t)}$ and proving knowledge of a DL-representation of

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})^{-1} h (g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}})^{-x_{\pi(l-t)}}$$

with respect to

$$\Big(g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}}, \ \ldots, \ g_{\pi(l-t-1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t-1}}\Big),$$

then obviously $\mathcal{V}$ learns information that may not have been computable from its a priori information and the status of the formula. From this counterexample it is also seen that it does not suffice for the proof of knowledge in Proposition 3.3.4 to be witness-hiding.

A practical implementation of the proof of knowledge in Proposition 3.3.4 can be realized by substituting the proof of knowledge described in Section 2.4.3; according to Proposition 2.4.8 it is perfectly witness-indistinguishable. An important benefit of using this protocol is that the resulting expressions can be expanded, so that $\mathcal{P}$ and $\mathcal{V}$ can use a single precomputed table for simultaneous repeated squaring, regardless of the formulae demonstrated. Other advantages of this particular choice of protocol will become clear in Sections 3.5, 6.3, and 6.4.

The resulting (generic) protocol steps are as follows:

Step 1. $\mathcal{P}$ generates at random $l-t$ numbers, $w_1, \ldots, w_{l-t} \in \mathbb{Z}_q$, and computes

$$a := \prod_{i=1}^{l-t} g_{\pi(i)}^{w_i} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\sum_{j=1}^{l-t} \alpha_{ij} w_j}.$$

$\mathcal{P}$ then sends the initial witness a to $\mathcal{V}$.

Step 2. $\mathcal{P}$ computes a set of responses, responsive to $\mathcal{V}$'s challenge $c \in \mathbb{Z}_s$, as follows:

$$r_i := c x_{\pi(i)} + w_i \bmod q \quad \forall i \in \{1, \ldots, l-t\}.$$

$\mathcal{P}$ then sends $(r_1, \ldots, r_{l-t})$ to $\mathcal{V}$.

$\mathcal{V}$ computes

$$r_{l-t+i} := b_i c - \sum_{j=1}^{l-t} \alpha_{ij} r_j \bmod q \quad \forall i \in \{1, \ldots, t\},$$

and accepts if and only if the verification relation

$$\prod_{i=1}^{l} g_{\pi(i)}^{r_i} h^{-c} = a$$

holds.

For later reference, in Sections 5.4 and 6.3, this protocol is depicted in Figure 3.1. The exponents in the expressions in Step 1 and in the verification relation, respectively, can be rapidly computed from the matrix of coefficients in (3.2), by taking the inner products of the matrix rows and the random numbers, and of the matrix rows and the responses, respectively. Note that the communication complexity of the protocol decreases as the number of “AND” connectives increases.

As with the proof of knowledge in Section 2.4.3, the above protocol description is generic in the sense that the binary size of s and the process of generating c have not been specified. To obtain a proof of knowledge, c should be generated in a substantially random manner and become known to $\mathcal{P}$ only after it has computed its initial witness.

The following property will be of importance in Section 3.5.

Proposition 3.3.5. *The protocol obtained by implementing the proof of knowledge in Proposition 3.3.4 by means of the proof of knowledge described in Section 2.4.3 is honest-verifier zero-knowledge.*

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, \ldots, g_l) := I_{\text{DLREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, \ldots, x_{l-1} \in \mathbb{Z}_q$
$x_l \in_{\mathcal{R}} \mathbb{Z}_q$
Secret key: $(x_1, \ldots, x_l)$
Public key: $h := \prod_{i=1}^{l} g_i^{x_i}$

PROTOCOL

$w_1, \ldots, w_{l-t} \in_{\mathcal{R}} \mathbb{Z}_q$
$a := \prod_{i=1}^{l-t} g_{\pi(i)}^{w_i} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\sum_{j=1}^{l-t} \alpha_{ij} w_j}$

$$\xrightarrow{\quad a \quad}$$

$$\xleftrightarrow{\quad c \quad}$$

$\forall i \in \{1, \ldots, l-t\}$:
$r_i := c x_{\pi(i)} + w_i \bmod q$

$$\xrightarrow{r_1, \ldots, r_{l-t}}$$

$\forall i \in \{1, \ldots, t\}$:
$r_{l-t+i} := b_i c - \sum_{j=1}^{l-t} \alpha_{ij} r_j \bmod q$
$\prod_{i=1}^{l} g_{\pi(i)}^{r_i} h^{-c} \stackrel{?}{=} a$

Figure 3.1: Generic protocol for demonstrating formula (3.3).

Proof. To simulate the view of the honest verifier, the simulator picks the challenge $c \in \mathbb{Z}_s$ according to the same distribution as the honest verifier. It then selects $r_1, \ldots, r_{l-t}$ at random from $\mathbb{Z}_q$, and computes $r_{l-t+1}, \ldots, r_l$ as would $\mathcal{V}$ in the protocol. Finally, it computes a such that the verification relation holds true:

$$a := \prod_{i=1}^{l} g_{\pi(i)}^{r_i} h^{-c}.$$

It is easy to check that the resulting view, $(a, c, r_1, \ldots, r_{l-t})$, is identically distributed to the view of the honest verifier. □

Corollary 3.3.2 suggests that all the considerations in Sections 2.4.3 and 2.5.3 apply. Special care must be taken, though, because both h and the tuple with respect to which knowledge of a DL-representation is demonstrated depend on the particular formula that is demonstrated. This does not pose any problems for the zero-knowledge proof mode, which in light of the techniques that will be introduced in Section 6.3 is best realized by prepending a move in which V commits to its challenge. Caution must be exercised in case of signed proofs, though. Details follow.

To obtain a signed proof, $\mathcal{V}$'s challenge c is generated as a sufficiently strong one-way hash of at least a, in accordance with the Fiat-Shamir technique described in Section 2.5.2. To enable provability in the random oracle model, we will from now on always assume implicitly that the hash function produces outputs linear in k. The signed proof consists of $(a, (r_1, \ldots, r_{l-t}))$ or of $(c, (r_1, \ldots, r_{l-t}))$, and if the protocol is non-interactive then $\mathcal{P}$ can omit sending either a or c to $\mathcal{V}$, since $\mathcal{V}$ can recover it. If a message m is hashed along, then the signed proof also serves as a digital signature of $\mathcal{P}$ on m. Proposition 3.3.4 still applies, since the hashing does not affect the property of witness-indistinguishability; signed proofs unconditionally hide all attribute information that was not explicitly disclosed. The unforgeability of signed proofs is guaranteed computationally, as follows.

Proposition 3.3.6. *Suppose that the proof of knowledge in Proposition 3.3.4 is realized by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.3, and that $\mathcal{V}$'s challenge is formed by hashing at least a. If the DL function used to implement $\mathcal{P}$'s commitment is one-way, then non-interactively issued signed proofs are provably unforgeable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of* $(x_1, \ldots, x_{l-1})$.

The proof follows by application of Proposition 2.5.2, in light of Corollary 3.3.2 and Proposition 3.3.5. We stress that Proposition 3.3.6 holds even in case of an adaptively chosen formula attack (and, if messages are hashed along, an adaptively chosen message attack), as do all the other unforgeability results in this chapter.

Assuming that $\mathcal{P}$ performs no more than polylogarithmically many protocol executions, and that the status of the formulae requested by $\mathcal{V}$ still leaves non-negligible uncertainty about $(x_1, \ldots, x_{l-1})$, it can be shown that Proposition 3.3.6 holds true even for interactively issued signed proofs. However, once $\mathcal{V}$ has requested sufficiently many formulae to be able to determine $(x_1, \ldots, x_{l-1})$ with overwhelming probability, it is unclear how to prove unforgeability in the random oracle model; we in effect end up with the question of whether interactively issued Schnorr signatures are unforgeable, which is believed true but has yet to be proved. To prove unforgeability of interactively issued signed proofs in general, we must relate the security proof to the construction of Proposition 2.5.5.

Proposition 3.3.7. *Let $l \geq 3$, let x_{l-1} be the outcome of a random coin flip by $\mathcal{P}$, and let $\mathcal{P}$ only demonstrate formulae in which both x_{l-1} and x_l do not appear. Suppose that the proof of knowledge in Proposition 3.3.1 is realized by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.3, and that $\mathcal{V}$'s challenge is formed by hashing at least a. If the DL function used to implement $\mathcal{P}$'s commitment is one-way, and $\mathcal{P}$ performs no more than polylogarithmically many formula demonstrations, then interactively issued signed proofs are provably unforgeable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-2})$.*

The proof follows from Proposition 2.5.3. It is not hard to show that the result holds even for an unbiased coin flip, if only the uncertainty about the outcome is non-negligible.[1] Note that x_{l-1} need not take on the values 0 and 1; any set of values, of any size, will do, if only x_{l-1} does not take on one particular value with overwhelming probability.

By blinding its challenge, $\mathcal{V}$ can perfectly blind the signed proof, in the manner described in Section 2.5. Since this proof mode is of no use in this book, we omit the details.

Propositions 3.3.6 and 3.3.7 tell us that the number of signed proofs computable by $\widehat{\mathcal{V}}$ cannot exceed the number of protocol executions performed by $\overline{\mathcal{P}}$, except with negligible probability. For some applications this is all we need, but in other applications it is not. An entirely different (and new) issue is the *unmodifiability* of signed proofs: it should be infeasible to construct a signed proof for a formula that does in fact not apply to $\mathcal{P}$'s representation. Unless special precaution is taken, a signed proof convinces only that $\mathcal{P}$ knows a DL-representation of h with respect to $(g_1, \ldots, g_l)$. As an example, if $h = g_1^{x_1} g_2^{x_2}$, and formulae of the form $x_2 = b \bmod q$ are to be demonstrated, for arbitrary $b \in \mathbb{Z}_q$, then $\widehat{\mathcal{P}}$ can set $a := g_1^{w_1} g_2^{w_2}$, for random w_1 and w_2, and $b := x_2 + w_2/\mathcal{H}_{q,g_2}(m, a) \bmod q$. In case the signed proof is to be issued interactively, this requires $\widehat{\mathcal{P}}$ to conspire with $\mathcal{V}$, because normally the formula will be determined prior to executing the protocol. If, on the other hand, $\mathcal{P}$

[1] In a practical implementation one would likely not bother to introduce the extra coin flip x_{l-1}, since the benefit that comes from it appears to be only of a theoretical nature; the unforgeability property of interactively issued signed proofs is believed to hold even if it is omitted.

is to non-interactively issue a signed proof for demonstrating a formula of its own choice, then it does not need the assistance of $\mathcal{V}$.

One measure to ensure that a signed proof convinces of which formula has been demonstrated is to restrict all the matrix entries (the α_{ij}'s and the b_i's) that specify the formulae requested to sets V such that $|V|/q$ is negligible in k (e.g., sets of size $\sqrt{q}$). The set V may differ for each formula coefficient. This measure, however, restricts the range of formulae that $\mathcal{P}$ can demonstrate. The following measure does not have this drawback.

Proposition 3.3.8. *Non-interactively (interactively) issued signed proofs are provably unmodifiable in the random oracle model, subject to the conditions of Proposition 3.3.6 (Proposition 3.3.7), in case a uniquely identifying description of the formula is hashed along when forming $\mathcal{V}$'s challenge.*

To form a uniquely identifying description of the formula (3.3), one can concatenate, in a predetermined order, the α_{ij}'s and the b_i's. In case t and l are not fixed as part of the system, they must be part of the formula description as well.

In other words, when presented with h, a formula description F, an (optional) message m, and a signed proof $(c, r_1, \ldots, r_{l-t})$, any verifier can convince itself that the signed proof has been computed by a prover (possibly a group of parties) that applied its knowledge of a DL-representation of h, and that the prover demonstrated that its DL-representation satisfies F. To this end, the verifier must compute $r_{l-t+1}, \ldots, r_l$ in the manner specified in Figure 3.1 and verify that

$$c = \mathcal{H}_{q,g_l}(m, F, \prod_{i=1}^{l} g_{\pi(i)}^{r_i} h^{-c}).$$

In practice, one may prefer to apply both measures in combination, that is, to hash along the formula description as well as to restrict all the formula coefficients to small sets.

It is important to note that unmodifiability does not exclude the possibility for $\mathcal{V}$ to end up with a signed proof that convinces only of a part of the formula demonstrated by $\mathcal{P}$. In particular, by hashing along the formula TRUE (if $\mathcal{P}$ allows this) and lumping together $\mathcal{P}$'s responses, $\mathcal{V}$ can hide almost entirely the formula that has been demonstrated. This property enables $\mathcal{V}$ to protect its own privacy in applications in which it routinely submits its protocol transcripts to a central authority. Details will be provided in Section 5.3.

Preferably, h is hashed along as well when forming $\mathcal{V}$'s challenge. Micali and Reyzin [269] show in a general setting that this defeats attacks in practical implementations whereby the hash function is designed by an adversary in an attempt to enable forgery in an otherwise secure signature scheme. More importantly for our purposes, hashing h along is mandatory in situations where h is not fixed but may be chosen in a fairly arbitrary manner by $\mathcal{P}$; this is the case when our showing protocol techniques are combined with certificate issuing protocols in which the receiver is in control of generating its public key. For this reason we will from now on always hash along h in our protocol descriptions. As noted in Section 2.5.2, from the point of view of security it is strongly recommended anyway to hash along any other data that the verifier of a signed proof must apply.

To illustrate the preceding techniques, we now present a practical example.

Example 3.3.9. *Suppose $\mathcal{P}$ has three attributes $x_1, x_2, x_3 \in \mathbb{Z}_q$, selected according to an arbitrary probability distribution, and is to demonstrate by means of a non-interactively issued signed proof to $\mathcal{V}$ that the following formula holds:*

$$(x_1 + 2x_2 - 10x_3 = 13) \text{ AND } (x_2 - 4x_3 = 5).$$

Rewriting this formula in the form (3.2), we get the formula

$$(x_1 = 2x_3 + 3) \text{ AND } (x_2 = 4x_3 + 5).$$

To demonstrate this formula, $\mathcal{P}$ generates a random $x_4 \in \mathbb{Z}_q$, and forms the commitment $h := \prod_{i=1}^{4} g_i^{x_i}$. If the formula holds true, then by substitution we get

$$\begin{aligned} h &= g_1^{x_1} g_2^{x_2} g_3^{x_3} g_4^{x_4} \\ &= g_1^{2x_3+3} g_2^{4x_3+5} g_3^{x_3} g_4^{x_4}, \end{aligned}$$

and by collecting the constant powers, as well as the variable powers for each of x_2, x_3, we obtain

$$h = (g_1^3 g_2^5)(g_1^2 g_2^4 g_3)^{x_3} g_4^{x_4}.$$

Finally, by dividing both sides by $g_1^3 g_2^5$, we arrive at

$$(g_1^3 g_2^5)^{-1} h = (g_1^2 g_2^4 g_3)^{x_3} g_4^{x_4}.$$

Consequently, $\overline{\mathcal{P}}$ can prove knowledge of a DL-representation of $h/(g_1^3 g_2^5)$ with respect to $(g_1^2 g_2^4 g_3, g_4)$. According to Proposition 3.3.1, this is also sufficient to convince $\mathcal{V}$. By substituting the proof of knowledge described in Section 2.4.3, and expanding the resulting expressions, we obtain the protocol depicted in Figure 3.2. Here, m denotes an arbitrary message agreed on between the parties and F denotes a description uniquely identifying the formula demonstrated. The message is optional, and typically its inclusion serves primarily to protect against replay; hereto m should contain a random number, a counter, or a sufficiently accurate estimate of the time and date. Other information, such as an identifier of $\mathcal{V}$ or a public key to be used for session encryption, may be incorporated as well.

The security of the protocol for $\overline{\mathcal{P}}$ follows from the fact that x_4 has been chosen at random and does not appear in the formula demonstrated. Specifically, assuming that the underlying DL function is one-way, it follows from Propositions 3.3.4

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, g_2, g_3, g_4) := I_{\text{DLREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, x_2, x_3 \in \mathbb{Z}_q$

$x_4 \in_{\mathcal{R}} \mathbb{Z}_q$

Secret key: $(x_1, \ldots, x_4)$

Public key: $h := \prod_{i=1}^{4} g_i^{x_i}$

Additional information: $\mathcal{H}_{q,g_4}(\cdot)$

PROTOCOL

$w_1, w_2 \in_{\mathcal{R}} \mathbb{Z}_q$

$a := g_1^{2w_1} g_2^{4w_1} g_3^{w_1} g_4^{w_2}$

$c := \mathcal{H}_{q,g_4}(h, m, F, a)$

$r_1 := cx_3 + w_1 \bmod q$

$r_2 := cx_4 + w_2 \bmod q$

$$\xrightarrow{c, r_1, r_2}$$

$$c \overset{?}{=} \mathcal{H}_{q,g_4}(h, m, F, \\ g_1^{2r_1+3c} g_2^{4r_1+5c} g_3^{r_1} g_4^{r_2} h^{-c})$$

Figure 3.2: Protocol for Example 3.3.9.

and 3.3.8 that the signed proof is unforgeable and unmodifiable in the random oracle model. Moreover, it does not leak more information about (x_1, x_2, x_3) than the validity of the formula; this holds even if $\overline{\mathcal{P}}$ demonstrates arbitrarily many other formulae involving only x_1, x_2, x_3, and $\widetilde{\mathcal{V}}$ can adaptively choose in each protocol execution which formula is to be demonstrated and which message is to be signed. As a consequence, $\overline{\mathcal{P}}$ is ensured that the data disclosed in all its showing protocol executions is no more than the aggregate of the information explicitly disclosed in each individual execution.

In accordance with the proof of Proposition 3.3.4, the powers of g_4 are cleanly "separated" from the other products of powers, because x_4 does not appear in the formula demonstrated. In particular, the computations corresponding to the powers of g_4 can all be performed by proving knowledge of the discrete logarithm of $g_4^{x_4}$ with respect to g_4, using the Schnorr proof of knowledge. This property, which can easily be seen to hold in general, will be of major importance in Sections 6.3 and 6.4, where it is shown how to lift the techniques of this chapter to the smartcard setting.

3.3.2 Technique based on the RSAREP function

To base the proof technique on the hardness of inverting the RSA function, consider $\mathcal{P}$ having to demonstrate the system of linear relations (3.2). As described in Section 3.2, in this case $\mathcal{P}$ commits to the attributes $(x_1, \ldots, x_l)$ by means of $h := g_1^{x_1} \cdots g_l^{x_l} x_{l+1}^v$, where x_{l+1} is chosen at random from $\mathbb{Z}_n^*$.

The proof of the following proposition is similar to that of Proposition 3.3.1.

Proposition 3.3.10. *$\mathcal{P}$ can prove knowledge of an RSA-representation of*

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})^{-1} h$$

with respect to

$$\Big(g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}}, \; \ldots, \; g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}}, \; v\Big)$$

if and only if it knows a set of attributes that satisfies the formula.

The RSA-representation $\overline{\mathcal{P}}$ can prove knowledge of is $(x_{\pi(1)}, \ldots, x_{\pi(l-t)}, x_{l+1})$.

Corollary 3.3.11. *Regardless of the formula demonstrated by $\mathcal{P}$, the proof of knowledge in Proposition 3.3.10 is also a proof of knowledge of an RSA-representation of h with respect to $(g_1, \ldots, g_l, v)$.*

The proof of the following result is similar to that of Proposition 3.3.4.

Proposition 3.3.12. *Let $\mathcal{P}$ demonstrate formulae using a proof of knowledge as described in Proposition 3.3.10 with the extra property that it is statistically witness-indistinguishable. For any distribution of $(x_1, \ldots, x_l)$, whatever information $\widetilde{\mathcal{V}}$ in an adaptively chosen formula attack can compute about $(x_1, \ldots, x_l)$ can also be computed using merely its a priori information and the status of the formulae requested.*

As in the case of the DLREP function, it is beneficial to implement the proof of knowledge in Proposition 3.3.10 using the witness-indistinguishable proof of knowledge of Section 2.4.4. The resulting expressions can be expanded, so that $\mathcal{P}$ and $\mathcal{V}$ can use a single precomputed table for simultaneous repeated squaring, regardless of the formulae demonstrated. This is straightforward, and so we refrain from a detailed stepwise description of the resulting protocol. For later reference, in Sections 5.4 and 6.3, the protocol is depicted in Figure 3.3. Note that $\mathcal{V}$ may alternatively apply the $\bmod v$ operator to each of $r_{l-t+1}, \ldots, r_l$, and multiply the corresponding "junk" factor,

$$\prod_{i=1}^{t} g_{\pi(l-t+i)}^{(b_i c - \sum_{j=1}^{l-t} \alpha_{ij} r_j) \operatorname{div} v},$$

into r_{l+1} before applying the verification relation.

It is easy to see that the protocol is honest-verifier zero-knowledge. The following three propositions are straightforward analogues of results stated for the DLREP-based setting, and can be proved in a similar manner.

Proposition 3.3.13. *Suppose that the proof of knowledge in Proposition 3.3.12 is realized by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.4, and that $\mathcal{V}$'s challenge is formed by hashing at least a. If the RSA function used to implement $\mathcal{P}$'s commitment is one-way, then non-interactively issued signed proofs are unforgeable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_l)$.*

Proposition 3.3.14. *Let $l \geq 2$, let x_l be the outcome of a random coin flip by $\mathcal{P}$, and let $\mathcal{P}$ only demonstrate formulae in which x_l does not appear. Suppose that the proof of knowledge in Proposition 3.3.10 is realized by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.4, and that $\mathcal{V}$'s challenge is formed by hashing at least a. If the RSA function used to implement $\mathcal{P}$'s commitment is one-way, and $\mathcal{P}$ does not perform more than polylogarithmically many formula demonstrations, then interactively issued signed proofs are provably unforgeable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-1})$.*

The remarks made about the coin flip in Proposition 3.3.7 apply here as well.

Proposition 3.3.15. *Non-interactively (interactively) issued signed proofs are provably unmodifiable in the random oracle model, subject to the conditions of Proposition 3.3.13 (Proposition 3.3.14), in case a uniquely identifying description of the formula is hashed along when computing $\mathcal{V}$'s challenge.*

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$(n, v, g_1, \ldots, g_l) := I_{\text{RSAREP}}(1^k)$

KEY SET-UP

Attributes: $x_1, \ldots, x_l \in \mathbb{Z}_v$
$x_{l+1} \in_{\mathcal{R}} \mathbb{Z}_n^*$
Secret key: $(x_1, \ldots, x_l, x_{l+1})$
Public key: $h := \prod_{i=1}^{l} g_i^{x_i} x_{l+1}^v$

PROTOCOL

$w_1, \ldots, w_{l-t} \in_{\mathcal{R}} \mathbb{Z}_v$
$w_{l+1} \in_{\mathcal{R}} \mathbb{Z}_n^*$
$a := \prod_{i=1}^{l-t} g_{\pi(i)}^{w_i} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\sum_{j=1}^{l-t} \alpha_{ij} w_j} w_{l+1}^v$

$\xrightarrow{a}$

$\xleftrightarrow{c}$

$\forall i \in \{1, \ldots, l-t\}:$
$\quad r_i := c x_{\pi(i)} + w_i \bmod v$
$r_{l+1} := \prod_{j=1}^{l-t} (g_{\pi(j)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{ij}})^{(c x_{\pi(j)} + w_j) \operatorname{div} v} x_{l+1}^c w_{l+1}$

$\xrightarrow{r_1, \ldots, r_{l-t}, r_{l+1}}$

$\forall i \in \{1, \ldots, t\}:$
$\quad r_{l-t+i} := b_i c - \sum_{j=1}^{l-t} \alpha_{ij} r_j$
$\prod_{i=1}^{l} g_{\pi(i)}^{r_i} r_{l+1}^v h^{-c} \stackrel{?}{=} a$

Figure 3.3: Generic protocol for demonstrating formula (3.3), with v replacing q.

As with the DLREP function, the alternative measure of restricting all the matrix entries used to specify the Boolean formulae to sets V such that $|V|/q$ is negligible in k, instead of hashing along F, restricts the range of formulae that $\mathcal{P}$ can demonstrate.

Example 3.3.16. *Suppose that $\mathcal{P}$ has three attributes $x_1, x_2, x_3 \in \mathbb{Z}_v$, and is to demonstrate by means of an interactively issued signed proof to $\mathcal{V}$ that the formula in Example 3.3.9 holds. By applying the technique of Proposition 3.3.10, substituting the proof of knowledge described in Section 2.4.4 and expanding the resulting expressions, we obtain the protocol depicted in Figure 3.4. Assuming that the underlying RSA function is one-way, and $\mathcal{P}$ performs no more than polylogarithmically many protocol executions, in the random oracle model the signed proof is unforgeable and unmodifiable. Moreover, it does not leak more information about (x_1, x_2, x_3) than the validity of the formula; this holds even if $\overline{\mathcal{P}}$ demonstrates arbitrarily many other formulae about its attributes.*

As in Example 3.3.9, m denotes an (optional) message agreed on between the parties and F denotes a description uniquely identifying the formula demonstrated. Examples of data that could be included in m are a nonce to protect against replay, an identifier of $\mathcal{V}$, a public key to be used for session encryption, and a free-form message.

When forming c, the description of any other formula that is implied by F may be hashed along instead of F itself. In addition, if the protocol is performed interactively, $\mathcal{V}$ can blind a. In Section 5.3 we will show how this enables $\mathcal{V}$ to unconditionally hide (any part of) the formula that has been demonstrated. However, $\mathcal{P}$ and $\mathcal{V}$ cannot form a signed proof for a formula that does not apply to $\mathcal{P}$'s RSA-representation, according to the property of unmodifiability.

A notable aspect of the protocol is that the computations for the numbers that are raised to the power v can all be performed by proving knowledge of the v-th root of x_5^v, using the Guillou-Quisquater proof of knowledge. More generally, the computations involving x_4 and x_5 can be performed by proving knowledge of an RSA-representation of $g_4^{x_4} x_5^v$ with respect to (g_4, v). This clean separation can easily be seen to hold in general, and will be of major importance in Sections 6.3 and 6.4.

3.4 Formulae with one "NOT" connective

We now show how to demonstrate satisfiable formulae from proposition logic with zero or more "AND" connectives, exactly one "NOT" connective, and no other connectives.

3.4.1 Technique based on the DLREP function

Any consistent system consisting of zero or more independent linear relations and one linear inequality (i.e., a linear relation where "$=$" is replaced by "$\neq$") can be

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, g_2, g_3, g_4, g_5) := I_{\text{RSAREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, x_2, x_3 \in \mathbb{Z}_q$
$x_4 \in_{\mathcal{R}} \{0, 1\}$
$x_5 \in_{\mathcal{R}} \mathbb{Z}_n^*$
Secret key: $(x_1, \dots, x_4, x_5)$
Public key: $h := \prod_{i=1}^{4} g_i^{x_i} x_5^v$
Additional information: $\mathcal{H}_{n,v}(\cdot)$

PROTOCOL

$w_1, w_2 \in_{\mathcal{R}} \mathbb{Z}_v$
$w_3 \in_{\mathcal{R}} \mathbb{Z}_n^*$
$a := g_1^{2w_1} g_2^{4w_1} g_3^{w_1} g_4^{w_2} w_3^v$

$$\xrightarrow{a}$$

$c := \mathcal{H}_{n,v}(h, m, F, a)$

$$\xleftarrow{c}$$

$r_1 := cx_3 + w_1 \bmod v$
$r_2 := cx_4 + w_2 \bmod v$
$r_3 := (g_1^2 g_2^4 g_3)^{(cx_3+w_1) \operatorname{div} v} g_4^{(cx_4+w_2) \operatorname{div} v} x_5^c w_3$

$$\xrightarrow{r_1, r_2, r_3}$$

$c \overset{?}{=} \mathcal{H}_{n,v}(h, m, F,$
$g_1^{2r_1+3c} g_2^{4r_1+5c} g_3^{r_1} g_4^{r_2} r_3^v h^{-c})$

Figure 3.4: Protocol for Example 3.3.16.

written as a system of linear relations by introducing a non-zero difference term, $\epsilon \in \mathbb{Z}_q^*$. Using Gaussian elimination, the system can be represented by the matrix equation

$$\begin{pmatrix} \alpha_{11} & \dots & \alpha_{1,l-t} & 1 & 0 & \dots & 0 \\ \alpha_{21} & \dots & \alpha_{2,l-t} & 0 & 1 & \dots & 0 \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_{t1} & \dots & \alpha_{t,l-t} & 0 & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} x_{\pi(1)} \\ x_{\pi(2)} \\ \vdots \\ x_{\pi(l)} \end{pmatrix} = \begin{pmatrix} b_1 - f_1\epsilon \\ b_2 - f_2\epsilon \\ \vdots \\ b_t - f_t\epsilon \end{pmatrix}, \qquad (3.4)$$

where $t \geq 1$. (Example 3.4.7 will clarify the process.) The coefficients α_{ij} are elements of $\mathbb{Z}_q$, and $f_1, \dots, f_t$ are numbers in $\mathbb{Z}_q$, not all equal to $0 \bmod q$. Clearly, it can always be ensured that f_1, say, is equal to 1. We may assume that $t \leq l$, because if $t = l + 1$ then $\mathcal{V}$ can verify the applicability of the formula without communicating with $\mathcal{P}$.

Our technique for demonstrating the Boolean formula that corresponds to the system (3.4) is based on the following result. Recall that $\mathcal{P}$ commits to the attributes $(x_1, \dots, x_l)$ by means of $h := \prod_{i=1}^{l} g_i^{x_i}$.

Proposition 3.4.1. *$\mathcal{P}$ can prove knowledge of a DL-representation of*

$$\prod_{i=1}^{t} g_{\pi(l-t+i)}^{f_i}$$

with respect to

$$\Big(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i} h^{-1},\ g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}},\ \dots,\ g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}}\Big)$$

if and only if it knows a set of attributes that satisfies the system (3.4).

Proof. If $(x_1, \dots, x_l)$ satisfies system (3.4), then

$$\begin{aligned} h &= \prod_{i=1}^{l} g_{\pi(i)}^{x_{\pi(i)}} \\ &= (\prod_{i=1}^{l-t} g_{\pi(i)}^{x_{\pi(i)}})(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i - f_i\epsilon - \sum_{j=1}^{l-t} \alpha_{ij} x_{\pi(j)}}) \\ &= (\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})(g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}})^{x_{\pi(1)}} \cdots (g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}})^{x_{\pi(l-t)}} \cdot \\ &\quad (\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-f_i})^{\epsilon}. \end{aligned}$$

By dividing both sides of the equation by h, as well as by $(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{-f_i})^{\epsilon}$, we see that $(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{f_i})^{\epsilon}$ equals

$$((\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i})h^{-1})(g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}})^{x_{\pi(1)}} \cdots (g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}})^{x_{\pi(l-t)}}.$$

Since $\epsilon \neq 0 \bmod q$, both sides can be raised to the power δ, where δ denotes $\epsilon^{-1} \bmod q$. From this we see that the DL-representation that $\overline{\mathcal{P}}$ can prove knowledge of is

$$(\delta, x_{\pi(1)}\delta \bmod q, \ldots, x_{\pi(l-t)}\delta \bmod q).$$

To prove the converse, suppose that $\widehat{\mathcal{P}}$ convinces $\overline{\mathcal{V}}$ with non-negligible success probability. There exists a polynomial-time knowledge extractor $\mathcal{K}$ that outputs with non-negligible success probability a DL-representation $(y_0, \ldots, y_{l-t})$. From the fact that $\prod_{i=1}^{t} g_{\pi(l-t+i)}^{f_i}$ equals

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i} h^{-1})^{y_0} (g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}})^{y_1} \cdots (g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}})^{y_{l-t}}$$

it follows that

$$h^{y_0} = \prod_{i=1}^{l-t} g_{\pi(i)}^{y_{l-t}} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i y_0 - f_i - \sum_{j=1}^{l-t} \alpha_{ij} y_j}.$$

If $y_0 = 0 \bmod q$ then

$$(y_1, \ldots, y_{l-t}, -f_1 - \sum_{j=1}^{l-t} \alpha_{1j} y_j \bmod q, \ldots, -f_t - \sum_{j=1}^{l-t} \alpha_{tj} y_j \bmod q) \qquad (3.5)$$

is a DL-representation of 1 with respect to $(g_{\pi(1)}, \ldots, g_{\pi(l)})$, and two cases can be distinguished:

- If this is the trivial DL-representation, then $y_1, \ldots, y_{l-t}, f_1, \ldots, f_t$ must all be zero, in addition to y_0. But this is a contradiction, because $t \geq 1$ and at least one of $f_1, \ldots, f_t$ is unequal to $0 \bmod q$, by virtue of the presence of one “NOT” connective.
- If this is a non-trivial DL-representation of 1, then $< \widehat{\mathcal{P}}, \overline{\mathcal{V}} >$ has computed a DL-representation of 1 other than the trivial one. (If $\mathcal{P}$ does not know a representation of h then $\mathcal{K}$ can be used directly to compute DL-representations.) According to Proposition 2.3.3 this contradicts the assumption that the DL function is one-way.

Therefore, $y_0 \neq 0 \bmod q$, and it follows that $(y_1 y_0^{-1}, \ldots, y_{l-t} y_0^{-1}, b_1 - f_1 y_0^{-1} - y_0^{-1} \sum_{j=1}^{l-t} \alpha_{1j} y_j, \ldots, b_t - f_t y_0^{-1} - y_0^{-1} \sum_{j=1}^{l-t} \alpha_{tj} y_j)$ is a DL-representation of h with respect to $(g_{\pi(1)}, \ldots, g_{\pi(l)})$. This must be the DL-representation $(x_{\pi(1)}, \ldots, x_{\pi(l)})$, for the same reason as in the proof of Proposition 3.3.1.

It remains to check that the DL-representation satisfies the formula (3.4). From the left-hand side of the matrix equation (3.4) it follows, for all $i \in \{1, \ldots, t\}$, that

$$\begin{aligned}
\sum_{j=1}^{l-t} \alpha_{ij} x_{\pi(j)} + x_{\pi(l-t+i)} &= \sum_{j=1}^{l-t} \alpha_{ij} (y_j y_0^{-1}) + b_i - f_i y_0^{-1} - y_0^{-1} \sum_{j=1}^{l-t} \alpha_{ij} y_j \\
&= y_0^{-1} \sum_{j=1}^{l-t} \alpha_{ij} y_j + b_i - f_i y_0^{-1} - y_0^{-1} \sum_{j=1}^{l-t} \alpha_{ij} y_j \\
&= b_i - f_i y_0^{-1} \bmod q.
\end{aligned}$$

Since $y_0 \neq 0 \bmod q$, this is equal to the i-th entry in the vector on the righthand side; note that we have $y_0^{-1} = \epsilon \bmod q$. This completes the proof. □

As in the case of Proposition 3.3.1, the expressions appearing in this proposition are fairly intimidating, but the process of deriving them is simple. In addition to the two steps of substitution and regrouping by collecting terms, we now also collect the terms that are raised to the power ϵ and move the resulting expression to the left-hand side. Example 3.4.7 will illustrate the process.

The following property is implicit in the proof of Proposition 3.4.1, and will be of major importance in Sections 5.5.2 and 6.3.

Corollary 3.4.2. *Regardless of the formula demonstrated by $\mathcal{P}$, the proof of knowledge in Proposition 3.4.1 is also a proof of knowledge of a DL-representation of h with respect to $(g_1, \ldots, g_l)$.*

We now extend the category of adaptively chosen formula attacks defined in Definition 3.3.3. This time $\widetilde{\mathcal{V}}$ may request the demonstration of formulae of the form (3.3) as well as of the form (3.4).

Proposition 3.4.3. *Let $\overline{\mathcal{P}}$ only demonstrate formulae in which x_l does not appear, using a proof of knowledge as described in Proposition 3.4.1 with the property that it is statistically witness-indistinguishable. For any distribution of $(x_1, \ldots, x_{l-1})$, whatever information $\widetilde{\mathcal{V}}$ in an adaptively chosen formula attack can compute about $(x_1, \ldots, x_{l-1})$ can also be computed using merely its a priori information and the status of the formulae requested.*

Proof. Consider first the demonstration of a single formula. If x_l does not appear in the formula (3.4), then in Proposition 3.4.1 the generator g_l appears separately in the tuple with respect to which $\mathcal{P}$ proves knowledge of a DL-representation. In particular, $\mathcal{P}$ proves knowledge of a DL-representation $(\epsilon^{-1} \bmod q, y_1, \ldots, y_j, x_l \epsilon^{-1} \bmod$

q) of a number in G_q with respect to a tuple $(g_0^*, \ldots, g_j^*, g_l)$, for some $j \in \{0, \ldots, l-1\}$ and numbers $g_0^*, \ldots, g_j^*$ in G_q that depend on the formula demonstrated. The set $\{y_1, \ldots, y_j\}$ is a subset of

$$\{x_1\epsilon^{-1} \bmod q, \ldots, x_{l-1}\epsilon^{-1} \bmod q\}.$$

Clearly, $x_l\epsilon^{-1} \bmod q$ is uniformly distributed over $\mathbb{Z}_q$, because x_l has been chosen at random by $\mathcal{P}$ and $\epsilon^{-1} \neq 0 \bmod q$. Since g_l is a generator of G_q, for any tuple $(y_1, \ldots, y_j) \in (\mathbb{Z}_q)^j$ and for any $\epsilon \in \mathbb{Z}_q^*$ there is exactly one x_l such that $(\epsilon^{-1} \bmod q, y_1, \ldots, y_j, x_l\epsilon^{-1} \bmod q)$ is the DL-representation that $\mathcal{P}$ proves knowledge of. Because the proof of knowledge is witness-indistinguishable, and x_l has been chosen at random by $\mathcal{P}$, it follows that no information is revealed about $(x_1, \ldots, x_{l-1})$ beyond the status of the formula. In particular, no information about ϵ leaks.

The rest of proof is as in Proposition 3.3.4. $\square$

A practical implementation of the proof of knowledge in Proposition 3.4.3 can be realized by substituting the perfect witness-indistinguishable proof of knowledge described in Section 2.4.3, and expanding the resulting expressions. The resulting (generic) protocol steps are as follows:

Step 1. $\mathcal{P}$ generates at random $l-t+1$ numbers, $w_0, \ldots, w_{l-t} \in \mathbb{Z}_q$, and computes

$$a := h^{-w_0} \prod_{i=1}^{l-t} g_{\pi(i)}^{w_i} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i w_0 - \sum_{j=1}^{l-t} \alpha_{ij} w_j}.$$

$\mathcal{P}$ then sends the initial witness a to $\mathcal{V}$.

Step 2. Let $\delta := \epsilon^{-1} \bmod q$. $\mathcal{P}$ computes a set of responses, responsive to $\mathcal{V}$’s challenge $c \in \mathbb{Z}_s$, as follows:

$$\begin{array}{rcl} r_0 & := & c\delta + w_0 \bmod q, \\ r_i & := & c x_{\pi(i)}\delta + w_i \bmod q \quad \forall i \in \{1, \ldots, l-t\}. \end{array}$$

$\mathcal{P}$ then sends $(r_0, \ldots, r_{l-t})$ to $\mathcal{V}$.

$\mathcal{V}$ computes

$$r_{l-t+i} := b_i r_0 - f_i c - \sum_{j=1}^{l-t} \alpha_{ij} r_j \bmod q \quad \forall i \in \{1, \ldots, t\},$$

and accepts if and only if the verification relation

$$\prod_{i=1}^{l} g_{\pi(i)}^{r_i} h^{-r_0} = a$$

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, \ldots, g_l) := I_{\text{DLREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, \ldots, x_{l-1} \in \mathbb{Z}_q$
$x_l \in_{\mathcal{R}} \mathbb{Z}_q$
Secret key: $(x_1, \ldots, x_l)$
Public key: $h := \prod_{i=1}^{l} g_i^{x_i}$

PROTOCOL

$w_0, \ldots, w_{l-t} \in_{\mathcal{R}} \mathbb{Z}_q$
$a := h^{-w_0} \prod_{i=1}^{l-t} g_{\pi(i)}^{w_i} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i w_0 - \sum_{j=1}^{l-t} \alpha_{ij} w_j}$

$$\xrightarrow{\quad a \quad}$$

$$\xleftrightarrow{\quad c \quad}$$

$\delta := \epsilon^{-1} \bmod q$
$r_0 := c\delta + w_0 \bmod q$
$\forall i \in \{1, \ldots, l-t\}:$
  $r_i := c x_{\pi(i)} \delta + w_i \bmod q$

$$\xrightarrow{r_0, \ldots, r_{l-t}}$$

$\forall i \in \{1, \ldots, t\}:$
  $r_{l-t+i} := b_i r_0 - f_i c - \sum_{j=1}^{l-t} \alpha_{ij} r_j \bmod q$
$\prod_{i=1}^{l} g_{\pi(i)}^{r_i} h^{-r_0} \overset{?}{=} a$

Figure 3.5: Generic protocol for demonstrating formula (3.4).

holds.

For later reference, in Sections 5.4 and 6.3, the protocol is depicted in Figure 3.5. As with the proof of knowledge in Section 2.4.3, the protocol description is generic in the sense that the binary size of s and the process of generating c have not yet been specified.

The following proposition will be of importance in Section 3.5, and can be proved in a manner similar to the proof of Proposition 3.3.5.

Proposition 3.4.4. *The protocol obtained by implementing the proof of knowledge in Proposition 3.4.3 by means of the proof of knowledge described in Section 2.4.3 is honest-verifier zero-knowledge.*

As suggested by Corollary 3.4.2, the propositions of Section 2.4.3 apply, with the obvious modifications. Special care must be taken with respect to signed proofs, though. By way of example, consider $h := g_1^{x_1} g_2^{x_2}$, and assume that a formula of the form $x_1 \neq \alpha x_2 + \beta \bmod q$ must be demonstrated. The verification relation is

$$g_1^{\alpha r_1 + \beta r_2 - c} g_2^{r_1} = h^{r_2} a.$$

Now, suppose that c is formed by hashing a and possibly h but not a description of the formula. In sharp contrast to the situation in Section 3.3.1, where signed proofs for which the formula description is not hashed along still serve as proofs of knowledge of a DL-representation of h, the triple (c, r_1, r_2) does not serve as a signed proof of knowledge of a DL-representation of h with respect to (g_1, g_2). To come up with (h, a) and (c, r_1, r_2) that meet the verification relation, one can pick any $h \in G_q$, set $a := h^{-r_2} g_1^{w_1} g_2^{w_2}$ for any $(r_2, w_1, w_2) \in \mathbb{Z}_q \times \mathbb{Z}_q \times \mathbb{Z}_q^*$, set $c := \mathcal{H}_{q,g_2}(h, a)$, set $r_1 := w_2$, and compute $\alpha := (w_1 + c - \beta r_2) r_1^{-1} \bmod q$ for any $\beta \in \mathbb{Z}_q$. To get around this, a description of the demonstrated formula must be hashed along. (The alternative is to restrict the matrix entries used to specify the formula to sets V such that $|V|/q$ is negligible in k.) In the following two propositions, F denotes a unique description of the formula demonstrated.

Proposition 3.4.5. *Suppose that the proof of knowledge in Proposition 3.4.3 is realized by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.3, and that $\mathcal{V}$'s challenge is formed by hashing at least (a, F). If the DL function that is used to implement $\mathcal{P}$'s commitment is one-way, then non-interactively issued signed proofs are provably unforgeable and unmodifiable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-1})$.*

The proof follows by application of Proposition 2.5.2, in light of Corollary 3.4.2 and Proposition 3.4.4.

Proposition 3.4.6. *Let $l \geq 3$, let x_{l-1} be the outcome of a random coin flip by $\mathcal{P}$, and let $\mathcal{P}$ only demonstrate formulae in which both x_{l-1} and x_l do not appear.*

Suppose that the proof of knowledge in Proposition 3.4.1 is realized by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.3, and that $\mathcal{V}$'s challenge is formed by hashing at least (a, F). If the DL function used to implement $\mathcal{P}$'s commitment is one-way, and $\mathcal{P}$ performs no more than polylogarithmically many formula demonstrations, then interactively issued signed proofs are provably unforgeable and unmodifiable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-2})$.

As before, if h is not fixed but may be chosen in a substantially arbitrary manner by $\mathcal{P}$, then h should be hashed along when forming $\mathcal{V}$'s challenge.

According to the unmodifiability property, $\mathcal{P}$ and $\mathcal{V}$ cannot construct a signed proof for a formula that does in fact not apply to $\mathcal{P}$'s DL-representation. On the other hand, as we will show in Section 5.3, it is possible for $\mathcal{V}$ to obtain a signed proof for any formula F' implied by F, while being convinced itself of F.

We end with a practical example.

Example 3.4.7. *Suppose $\mathcal{P}$ has three attributes $x_1, x_2, x_3 \in \mathbb{Z}_q$, and is to demonstrate by means of a non-interactively issued signed proof to $\mathcal{V}$ that the following formula holds:*

$$\mathsf{NOT}(x_1 + 3x_2 + 5x_3 = 7) \;\mathsf{AND}\; (3x_1 + 10x_2 + 18x_3 = 23).$$

With ϵ denoting $7 - (x_1 + 3x_2 + 5x_3) \bmod q$, this formula is equivalent to the statement that there exists an $\epsilon \neq 0$ such that

$$(x_1 = 1 + 4x_3 - 10\epsilon) \;\mathsf{AND}\; (x_2 = 2 - 3x_3 + 3\epsilon).$$

To demonstrate this formula, $\mathcal{P}$ generates a random $x_4 \in \mathbb{Z}_q$, and forms the commitment $h := \prod_{i=1}^4 g_i^{x_i}$. If the formula holds true, then by substitution we get

$$\begin{aligned} h &= g_1^{x_1} g_2^{x_2} g_3^{x_3} g_4^{x_4} \\ &= g_1^{1+4x_3-10\epsilon} g_2^{2-3x_3+3\epsilon} g_3^{x_3} g_4^{x_4}, \end{aligned}$$

and by regrouping in three manners (according to constants, variables, and ϵ), we obtain

$$h = (g_1^1 g_2^2)(g_1^4 g_2^{-3} g_3)^{x_3} g_4^{x_4} (g_1^{-10} g_2^3)^{\epsilon}.$$

Finally, we divide both sides by h, as well as by $(g_1^{-10} g_2^3)^{\epsilon}$, and raise both sides to the power $\epsilon^{-1} \bmod q$, arriving at

$$g_1^{10} g_2^{-3} = (g_1^1 g_2^2 h^{-1})^{1/\epsilon} (g_1^4 g_2^{-3} g_3)^{x_3/\epsilon} g_4^{x_4/\epsilon}.$$

Therefore, $\mathcal{P}$ can prove knowledge of a DL-representation of $g_1^{10} g_2^{-3}$ with respect to the triple $(g_1^1 g_2^2 h^{-1}, g_1^4 g_2^{-3} g_3, g_4)$. According to Proposition 3.4.1, this is also

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, g_2, g_3, g_4) := I_{\text{DLREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, x_2, x_3 \in \mathbb{Z}_q$
$x_4 \in_{\mathcal{R}} \mathbb{Z}_q$
Secret key: $(x_1, \ldots, x_4)$
Public key: $h := \prod_{i=1}^{4} g_i^{x_i}$
Additional information: $\mathcal{H}_{q,g_4}(\cdot)$

PROTOCOL

$w_1, w_2, w_3 \in_{\mathcal{R}} \mathbb{Z}_q$
$a := g_1^{w_1+4w_2} g_2^{2w_1-3w_2} g_3^{w_2} g_4^{w_3} h^{-w_1}$
$c := \mathcal{H}_{q,g_4}(h, m, F, a)$
$\epsilon := 7 - (x_1 + 3x_2 + 5x_3) \bmod q$
$\delta := \epsilon^{-1} \bmod q$
$r_1 := c\delta + w_1 \bmod q$
$r_2 := cx_3\delta + w_2 \bmod q$
$r_3 := cx_4\delta + w_3 \bmod q$

$$\xrightarrow{c, r_1, r_2, r_3}$$

$$c \overset{?}{=} \mathcal{H}_{q,g_4}(h, m, F,$$
$$g_1^{r_1+4r_2-10c} g_2^{2r_1-3r_2+3c} g_3^{r_2} g_4^{r_3} h^{-r_1})$$

Figure 3.6: Protocol for Example 3.4.7.

sufficient to convince $\mathcal{V}$. By substituting the proof of knowledge described in Section 2.4.3, and expanding the resulting expressions, we obtain the protocol depicted in Figure 3.6.

The security of the protocol for $\overline{\mathcal{P}}$ follows from the fact that x_4 has been chosen at random and does not appear in the formula demonstrated. Specifically, assuming that the underlying DL function is one-way, in the random oracle model the signed proof is unforgeable and unmodifiable. Moreover, it does not leak more information about (x_1, x_2, x_3) than the validity of the formula; this holds even if $\overline{\mathcal{P}}$ demonstrates arbitrarily many other formulae (with x_4 not appearing in any of these).

A noteworthy aspect of the protocol is that the computations corresponding to the powers of g_4 can all be performed by proving knowledge of the discrete logarithm of $(g_4^{x_4})^{1/\epsilon}$ with respect to g_4, using the Schnorr proof of knowledge, because x_4 does not appear in the formula demonstrated. This property can easily be seen to hold in general, and will be of great importance in Sections 6.3 and 6.4.

3.4.2 Technique based on the RSAREP function

To base the technique on the hardness of inverting the RSA function, consider $\mathcal{P}$ having to demonstrate formula (3.4), with "$\bmod v$" replacing "$\bmod q$," and $\epsilon \in \mathbb{Z}_v^*$. Recall that $\mathcal{P}$ commits to $(x_1, \ldots, x_l)$ by means of $h := g_1^{x_1} \cdots g_l^{x_l} x_{l+1}^v$.

Proposition 3.4.8. *$\mathcal{P}$ can prove knowledge of an RSA-representation of*

$$\prod_{i=1}^{t} g_{\pi(l-t+i)}^{f_i}$$

with respect to

$$\Big(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i} h^{-1},\ g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}},\ \ldots,\ g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}}, v\Big)$$

if and only if it knows a set of attributes that satisfies the system (3.4).

Proof. If $(x_1, \ldots, x_l)$ satisfies the system (3.4), then by regrouping expressions it is seen that

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{f_i})^{\epsilon}$$

equals

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i} h^{-1})\, (g_{\pi(1)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i1}})^{x_{\pi(1)}} \cdots (g_{\pi(l-t)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{i,l-t}})^{x_{\pi(l-t)}} x_{l+1}^{v}.$$

Because inequality holds, $\epsilon \neq 0 \bmod v$ and so $\mathcal{P}$ can compute integers $e, f \in \mathbb{Z}$ such that $e\,\epsilon + fv = 1$, by using the extended Euclidean algorithm. (This can be always be done, because v is prime.) It follows that

$$\Big(e \bmod v, ex_{\pi(1)} \bmod v, \ldots, ex_{\pi(l-t)} \bmod v, z\Big)$$

is the RSA-representation sought for, where z denotes the expression

$$(\prod_{i=1}^{t} g_{\pi(l-t+i)}^{b_i} h^{-1})^{e \operatorname{div} v} \prod_{j=1}^{l-t} \Big(g_{\pi(j)} \prod_{i=1}^{t} g_{\pi(l-t+i)}^{-\alpha_{ij}}\Big)^{(ex_{\pi(j)}) \operatorname{div} v} x_{l+1}^{e} (\prod_{i=1}^{t} g_{\pi(l-t+i)}^{f_i})^{f}.$$

The proof of the converse is similar to the second part of the proof of Proposition 3.4.1, with the additional application of normalizations where needed. □

The protocol can be efficiently implemented by using the proof of knowledge described in Section 2.4.4 and expanding the resulting expressions. The direct analogue of Proposition 3.4.3 can be proved, as well as the unforgeability and unmodifiability of signed proofs (assuming at least a and F are hashed along when forming c). Since the necessary adaptations are straightforward, a further description is omitted.

3.5 Atomic formulae connected by “OR” connectives

We now show how $\mathcal{P}$ can demonstrate Boolean formulae in which subformulae, of either one of the two forms discussed in the previous sections, are connected by zero or more “OR” connectives.

3.5.1 Technique based on the DLREP function

The following definition defines the basic building block for the technique in this section.

Definition 3.5.1. *An* atomic *formula is one in which linear relations over $\mathbb{Z}_q$ are connected by zero or more “AND” connectives and at most one “NOT” connective.*

Any atomic formula can be described either by system (3.2) or by system (3.4). Consequently, any atomic formula applying to $\mathcal{P}$’s attributes can be demonstrated using either the method described in Section 3.3 or that described in Section 3.4.

Consider now the situation in which $\mathcal{P}$ is to demonstrate a satisfiable Boolean formula F of the form

$$F = F_1 \textsf{ OR } \ldots \textsf{ OR } F_j, \tag{3.6}$$

for some $j \geq 1$ that may be polynomial in the security parameter k. Each of $F_1, \ldots, F_j$ is an atomic formula. If F holds true for $\mathcal{P}$’s attributes, then at least one of the j atomic subformulae holds true, but $\widetilde{\mathcal{V}}$ should not be able to learn which one(s). Suppose that (at least) F_t holds true, for some $t \in \{1, \ldots, j\}$.

In the following protocol for demonstrating F, it is assumed that atomic formulae are demonstrated by substituting the witness-indistinguishable proof of knowledge described in Section 2.4.3 into the proof of knowledge in either Proposition 3.3.1 or Proposition 3.4.1 (whichever is appropriate):

Step 1. Using the honest-verifier zero-knowledge simulator that exists according to Proposition 3.3.5 or 3.4.4, $\mathcal{P}$ generates $j-1$ transcripts of subformulae demonstrations for $F_1, \ldots, F_{t-1}, F_{t+1}, \ldots, F_j$. For each $i \in \{1, \ldots, j\} \setminus \{t\}$, the simulated proof for formula F_i involves a random *self-chosen* challenge that we denote by c_i, an initial witness, and one or more self-chosen responses. For subformula F_t, $\mathcal{P}$ generates an initial witness in the manner specified by the standard protocol for demonstrating an atomic formula (to prepare for a genuine proof). $\mathcal{P}$ then sends all j initial witnesses, referred to as its *initial witness set*, to $\mathcal{V}$.

Step 2. $\mathcal{V}$ generates a random challenge $c \in \mathbb{Z}_s$, and sends it to $\mathcal{P}$.

Step 3. $\mathcal{P}$ forms j response sets, as follows. $\mathcal{P}$ computes $c_t := c - \sum_{i \neq t} c_i \bmod s$. Responsive to challenge c_t, $\mathcal{P}$ computes its responses corresponding to the demonstration of F_t, in the manner described in Step 2 of the protocol in either Section 3.3 or Section 3.4 (whichever is appropriate). For each of the remaining $j-1$ subformulae, $\mathcal{P}$ uses the self-chosen responses from the simulated formulae demonstrations prepared in Step 1. $\mathcal{P}$ then sends all j response sets and $(c_1, \ldots, c_j)$ to $\mathcal{V}$.

$\mathcal{V}$ verifies that $c = \sum_{i=1}^{j} c_i \bmod s$. If this verification holds, then for each of the j atomic subformulae it applies the verification relation for that subformula. Specifically, $\mathcal{V}$ verifies the demonstration of F_i by applying the verification relation that applies in the standard protocol for demonstrating F_i, using the i-th initial witness and response set provided by $\mathcal{P}$ and the challenge c_i. $\mathcal{V}$ accepts if and only if the j challenges sum up correctly and all j verification relations hold; together, these $j+1$ verification relations make up the verification process for the demonstration of F.

Note that the protocol encompasses the protocols of Sections 3.3 and 3.4 as special cases.

Proposition 3.5.2. *In the protocol for demonstrating F, assume that $\mathcal{P}$ only demonstrates formulae F in which x_l does not appear. The following properties hold:*

(a) The protocol is complete and sound.

(b) The protocol is a proof of knowledge of a DL-representation of h with respect to the tuple $(g_1, \ldots, g_l)$.

(c) For any distribution of $(x_1, \ldots, x_{l-1})$, whatever information $\widetilde{\mathcal{V}}$ in an adaptively chosen formula attack[2] *can compute about $(x_1, \ldots, x_{l-1})$ can also be computed using merely its a priori information and the status of the formulae requested.*

Proof. We only sketch the proof here; the details are easy to fill in.

(a) Completeness is verified straightforwardly. Soundness follows from the condition that the c_i's have to sum up to $\mathcal{V}$'s challenge c, so that $\mathcal{P}$ cannot simulate the demonstration for all j subformulae. For at least one subformula $\mathcal{P}$ has to use a challenge that it cannot anticipate: it must perform a genuine proof of knowledge for that subformula. Because the demonstration of an atomic subformula is a sound proof of knowledge, $\mathcal{P}$ can do this only if the subformula indeed holds true for its attributes; in other words, if F holds true.

(b) The soundness of the protocol for demonstrating F implies that at least one of the atomic subformulae holds true for $\mathcal{P}$'s attributes. According to Corollaries 3.3.2 and 3.4.2, the proof of knowledge for this atomic subformula is also a proof of knowledge of a DL-representation of h with respect to $(g_1, \ldots, g_l)$.

(c) The third claim follows from the perfect simulatability of the $j - 1$ subformulae demonstrations and the fact that the relation $\sum_{i=1}^{j} c_i = c \bmod s$ reveals no information on which $j - 1$ challenges have been self-chosen by $\mathcal{P}$, regardless of the manner in which c is formed. □

Care must be taken in a practical implementation that the subformula for which a genuine proof is performed cannot be deduced by timing the delay between sending $\mathcal{V}$'s challenge and receiving the j response sets.

To obtain a signed proof, $\mathcal{V}$'s challenge c should be generated as a sufficiently strong one-way hash of at least the j initial witnesses, a description of F, and an (optional) message m. If h is not fixed, then it should be hashed along as well. The signed proof consists of $(c_1, \ldots, c_j)$ and the j response sets, or of the j initial witnesses and the j response sets. If the protocol is performed non-interactively then $\mathcal{P}$ need not send either its initial witness set or its j challenges to $\mathcal{V}$, since they can be recovered by $\mathcal{V}$.

Proposition 3.5.3. *In Proposition 3.5.2, if the DL function used to implement $\mathcal{P}$'s commitment is one-way, and $\mathcal{V}$'s challenge is formed by hashing at least all j initial witnesses and F, then non-interactively issued signed proofs are provably unforgeable and unmodifiable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-1})$.*

Proposition 3.5.4. *Let $l \geq 3$, let x_{l-1} be the outcome of a random coin flip by $\mathcal{P}$, and let $\mathcal{P}$ only demonstrate formulae in which both x_{l-1} and x_l do not appear. If the DL function used to implement $\mathcal{P}$'s commitment is one-way, $\mathcal{V}$'s challenge is formed by*

[2]This time, $\mathcal{V}$ may request the demonstration of any Boolean formula of the form (3.6).

hashing at least all j initial witnesses and F, and $\mathcal{P}$ performs no more than polylogarithmically many formula demonstrations, then interactively issued signed proofs are provably unforgeable and unmodifiable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-2})$.

Again, the remarks made about the coin flip in Proposition 3.3.7 apply here as well.

If $j \geq 2$ in formula (3.6) the protocol is non-trivially witness-indistinguishable, and in Proposition 3.5.4 we can omit the requirement that x_{l-1} be random and do not appear in the formulae demonstrated by $\mathcal{P}$.

The protocol for demonstrating F admits several variations. For example:

- The relation $\sum_{i=1}^{j} c_i = c \bmod s$ can be replaced by any other relation with the property that, for any c, the selection of any $j-1$ challenges uniquely determines the remaining challenge. For example, for s a prime one can use $\prod_{i=1}^{j} c_i = c \bmod s$ or, if s is a power of 2, $\oplus_{i=1}^{j} c_i = c$, where $\oplus$ denotes the bitwise exclusive-or operator.

- Instead of simulating all but one of the subformula demonstrations, $\mathcal{P}$ could perform a genuine proof of knowledge for all subformulae that hold true and simulate the demonstration only for those that do not hold true.

Both variations bring no noteworthy performance gain, and we will see in Section 6.3 that they are disadvantageous when lifting the formulae demonstration techniques to the smartcard setting. Moreover, in Sections 5.3 and 5.4.2 we will show that the relation $\sum_{i=1}^{j} c_i = c \bmod q$ can be exploited to improve efficiency and privacy in applications where $\mathcal{V}$ must relay signed proofs to the CA. For these reasons we will not consider the two variations any further.

Example 3.5.5. *Suppose $\mathcal{P}$ has three attributes $x_1, x_2, x_3 \in \mathbb{Z}_q$, and is to demonstrate by means of a non-interactively issued signed proof to $\mathcal{V}$ that the following formula holds:*

$$\Big((x_1 + 2x_2 - 10x_3 = 13) \text{ AND } (x_2 - 4x_3 = 5)\Big) \text{ OR}$$
$$\Big(\text{NOT}(x_1 + 3x_2 + 5x_3 = 7) \text{ AND } (3x_1 + 10x_2 + 18x_3 = 23)\Big)$$

Assume for concreteness that

$$(x_1 + 2x_2 - 10x_3 = 13) \text{ AND } (x_2 - 4x_3 = 5)$$

holds, and that

$$\text{NOT}(x_1 + 3x_2 + 5x_3 = 7) \text{ AND } (3x_1 + 10x_2 + 18x_3 = 23)$$

does not necessarily hold. To demonstrate the formula, $\mathcal{P}$ generates a random $x_4 \in \mathbb{Z}_q$, and forms the commitment $h := \prod_{i=1}^4 g_i^{x_i}$. As we have seen in Example 3.3.9, to demonstrate the first subformula $\mathcal{P}$ proves knowledge of a DL-representation of $h/(g_1^3 g_2^5)$ with respect to $(g_1^2 g_2^4 g_3, g_4)$. According to Example 3.4.7, the second subformula requires a proof of knowledge of a DL-representation of $g_1^{10} g_2^{-3}$ with respect to $(g_1 g_2^2 h^{-1}, g_1^4 g_2^{-3} g_3, g_4)$; $\mathcal{P}$ simulates this demonstration, using a self-chosen challenge and self-chosen responses. The resulting protocol is depicted in Figure 3.7. Here, m denotes an (optional) message and F denotes a description uniquely identifying the formula demonstrated. The signed proof consists of $(c_1, c_2, r_1, r_2, r_3, r_4, r_5)$. Assuming that the underlying DL function is one-way, in the random oracle model the signed proof is unforgeable and unmodifiable. Moreover, it does not leak more information about (x_1, x_2, x_3) than the validity of the formula; this holds even if $\mathcal{P}$ demonstrates arbitrarily many other formulae.

3.5.2 Technique based on the RSAREP function

Adaptation to the difficulty of inverting the RSA function poses no particular difficulties, and is therefore omitted.

3.6 Demonstrating arbitrary Boolean formulae

We are now prepared for the final step: demonstrating arbitrary satisfiable Boolean formulae.

3.6.1 Technique based on the DLREP function

Suppose $\mathcal{P}$ is to demonstrate an arbitrary Boolean formula that applies to its attributes. Without loss of generality, we assume that F is of the conjunctive normal form,

$$F = F_1 \text{ AND } \ldots \text{ AND } F_j, \tag{3.7}$$

for some $j \geq 1$ that may be polynomial in k, where each of $F_1, \ldots, F_j$ are atomic subformulae of the form (3.6). The indices j here and in formula (3.6) do not bear any relation to one another. Likewise, each subformula may be composed of an arbitrary and distinct number of subformulae. The issue of writing an arbitrary Boolean formula in the form (3.7) is outside of the scope of this book.

Our technique for demonstrating F is as follows. F holds true for $\mathcal{P}$'s attributes if and only if $F_1, \ldots, F_j$ all hold true. To demonstrate F, $\mathcal{P}$ demonstrates each of $F_1, \ldots, F_j$ by means of the proof of knowledge described in the previous section, subject to the following two constraints:

- All j protocol executions are performed in parallel; and

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, g_2, g_3, g_4) := I_{\text{DLREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, x_2, x_3 \in \mathbb{Z}_q$
$x_4 \in_{\mathcal{R}} \mathbb{Z}_q$
Secret key: $(x_1, \ldots, x_4)$
Public key: $h := \prod_{i=1}^{4} g_i^{x_i}$
Additional information: $\mathcal{H}_{q,g_4}(\cdot)$

PROTOCOL

$w_1, w_2, c_2, r_3, r_4, r_5 \in_{\mathcal{R}} \mathbb{Z}_q$
$a_1 := g_1^{2w_1} g_2^{4w_1} g_3^{w_1} g_4^{w_2}$
$a_2 := g_1^{r_3+4r_4-10c_2} g_2^{2r_3-3r_4+3c_2} g_3^{r_4} g_4^{r_5} h^{-r_3}$
$c := \mathcal{H}_{q,g_4}(h, m, F, a_1, a_2)$
$c_1 := c - c_2 \bmod s$
$r_1 := c_1 x_3 + w_1 \bmod q$
$r_2 := c_1 x_4 + w_2 \bmod q$

$$\xrightarrow{c_1, c_2, (r_1, r_2, r_3, r_4, r_5)}$$

$c := c_1 + c_2 \bmod s$
$c \overset{?}{=} \mathcal{H}_{q,g_4}(h, m, F, g_1^{2r_1+3c_1} g_2^{4r_1+5c_1} g_3^{r_1} g_4^{r_2} h^{-c_1},$
$g_1^{r_3+4r_4-10c_2} g_2^{2r_3-3r_4+3c_2} g_3^{r_4} g_4^{r_5} h^{-r_3})$

Figure 3.7: Protocol for Example 3.5.5.

- $\mathcal{V}$'s challenge is the same in all j protocol executions.

$\mathcal{V}$ accepts if and only if it accepts $\mathcal{P}$'s demonstration of each of the j subformulae.

Although the two constraints are not strictly necessary, they are preferable in light of efficiency and, more importantly, for the purpose of lifting the protocol to the smartcard setting, as we will see in Section 6.3. Also, the constraints are desirable for the purpose of forming signed proofs.

The following proposition follows straightforwardly from Proposition 3.5.2 and the fact that using the same challenge for all subformulae does not increase $\widehat{\mathcal{P}}$'s cheating probability.

Proposition 3.6.1. *In the protocol for demonstrating F, assume that $\mathcal{P}$ only demonstrates formulae F in which x_l does not appear. The following properties hold:*

(a) The protocol is complete and sound.

(b) The protocol is a proof of knowledge of a DL-representation of h with respect to the tuple $(g_1, \ldots, g_l)$.

(c) For any distribution of $(x_1, \ldots, x_{l-1})$, whatever information $\widetilde{\mathcal{V}}$ in an adaptively chosen formula attack[3] *can compute about $(x_1, \ldots, x_{l-1})$ can also be computed using merely its a priori information and the status of the formulae requested.*

The proof is straightforward. (A description of what is essentially the knowledge extractor required to prove soundness is contained in the proof of Proposition 5.4.1.)

To obtain a signed proof, $\mathcal{V}$'s challenge c should be generated as a sufficiently strong one-way hash of the j initial witness sets, a description of F, and an (optional) message m. In case h is not fixed a priori, it should be hashed along as well. The signed proof consists all $\mathcal{P}$'s challenge and response sets, or equivalently of all $\mathcal{P}$'s initial witness sets and response sets. (As before, instead of hashing along a description of F one may alternatively restrict all the matrix entries to sets V such that $|V|/q$ is negligible in k.)

Proposition 3.6.2. *In Proposition 3.6.1, if the DL function used to implement $\mathcal{P}$'s commitment is one-way, and $\mathcal{V}$'s challenge is formed by hashing at least all initial witnesses and F, then non-interactively issued signed proofs are provably unforgeable and unmodifiable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-1})$.*

Proposition 3.6.3. *Let $l \geq 3$, let x_{l-1} be the outcome of a random coin flip by $\mathcal{P}$, and let $\mathcal{P}$ only demonstrate formulae F in which both x_{l-1} and x_l do not appear. If the DL function used to implement $\mathcal{P}$'s commitment is one-way, $\mathcal{V}$'s challenge is formed by hashing at least all initial witnesses and F, and $\mathcal{P}$ performs no more than*

[3]This time, $\mathcal{V}$ may request the demonstration of any Boolean formula of the form (3.7).

polylogarithmically many formula demonstrations, then interactively issued signed proofs are provably unforgeable and unmodifiable in the random oracle model, regardless of the formula(e) demonstrated and the distribution of $(x_1, \ldots, x_{l-2})$.

The remarks about the coin flip in Proposition 3.3.7 apply here as well.

We now posses all the machinery for $\mathcal{P}$ to demonstrate the example formula (3.1) in Section 3.1.

Example 3.6.4. *Suppose* $\mathcal{P}$ *has three attributes* $x_1, x_2, x_3 \in \mathbb{Z}_q$*, selected according to an arbitrary probability distribution, and is to demonstrate by means of a non-interactively issued signed proof to* $\mathcal{V}$ *that the example formula (3.1) holds true. Assume for concreteness that*

$$(x_1 + 2x_2 - 10x_3 = 13) \text{ AND } (x_2 - 4x_3 = 5)$$

holds. In Example 3.5.5 we have seen how to demonstrate the subformula appearing before the third "AND" connective. In accordance with Section 3.4, $\mathcal{P}$ *demonstrates the remaining part by proving knowledge of a DL-representation of* g_1 *with respect to* $(g_1^5 h^{-1}, g_1^8 g_2, g_1^{-11} g_3, g_4)$*. By merging the two protocols in accordance with the technique described in this section, we obtain the protocol depicted in Figure 3.8. As before,* m *denotes a message and* F *denotes a description uniquely identifying formula (3.1). Examples of data that could be included in* m *are a nonce to protect against replay, an identifier of* $\mathcal{V}$*, a public key to be used for session encryption, and a free-form message. The signed proof consists of* $(c_1, c_2, r_1, \ldots, r_9)$*. Assuming that the underlying DL function is one-way, in the random oracle model the signed proof is unforgeable and unmodifiable. Moreover, it does not leak more information about* (x_1, x_2, x_3) *than the validity of the formula; this holds even if* $\mathcal{P}$ *demonstrates arbitrarily many other formulae about its attributes.*

The performance estimates at the end of Section 3.1 are readily obtained. Namely, according to Section 2.2.2, 200*-bit challenges and responses suffice for long-term security. Both* $\mathcal{P}$ *and* $\mathcal{V}$*, to perform their computations in* G_q*, can use a precomputed table that contains the* 31 *products of the numbers in the non-empty subsets of* $\{g_1, \ldots, g_4, h\}$*. Performance can be pushed to the limit by using an elliptic curve implementation with* 20*-byte base numbers, but beware of the reservations expressed in Section 2.2.2. Also, the computational burden for* $\mathcal{V}$ *can be reduced by a factor of almost* 3 *by having* $\mathcal{P}$ *send along* (a_1, a_2, a_3) *(one of* c_1, c_2 *may be left out) and applying batch-verification, in the manner described in Section 2.5.3.*

3.6.2 Technique based on the RSAREP function

Again, adaptation to the difficulty of inverting the RSA function poses no particular difficulties.

$\boxed{\mathcal{P}}$ $\boxed{\mathcal{V}}$

SYSTEM PARAMETERS

$$(q, g_1, g_2, g_3, g_4) := I_{\text{DLREP}}(1^k)$$

KEY SET-UP

Attributes: $x_1, x_2, x_3 \in \mathbb{Z}_q$
$x_4 \in_{\mathcal{R}} \mathbb{Z}_q$
Secret key: $(x_1, \ldots, x_4)$
Public key: $h := \prod_{i=1}^{4} g_i^{x_i}$
Additional information: $\mathcal{H}_{q,g_4}(\cdot)$

PROTOCOL

$w_1, \ldots, w_6, c_2, r_3, r_4, r_5 \in_{\mathcal{R}} \mathbb{Z}_q$
$a_1 := g_1^{2w_1} g_2^{4w_1} g_3^{w_1} g_4^{w_2}$
$a_2 := g_1^{r_3+4r_4-10c_2} g_2^{2r_3-3r_4+3c_2} g_3^{r_4} g_4^{r_5} h^{-r_3}$
$a_3 := g_1^{5w_3+8w_4-11w_5} g_2^{w_4} g_3^{w_5} g_4^{w_6} h^{-w_3}$
$c := \mathcal{H}_{q,g_4}(h, m, F, a_1, a_2, a_3)$
$c_1 := c - c_2 \bmod s$
$r_1 := c_1 x_3 + w_1 \bmod q$
$r_2 := c_1 x_4 + w_2 \bmod q$
$\delta := \epsilon^{-1} \bmod q$
$r_6 := c\delta + w_3 \bmod q$
$r_7 := c x_2 \delta + w_4 \bmod q$
$r_8 := c x_3 \delta + w_5 \bmod q$
$r_9 := c x_4 + w_6 \bmod q$

$$\xrightarrow{c_1, c_2, (r_1, \ldots, r_9)}$$

$c := c_1 + c_2 \bmod s$
$c \overset{?}{=} \mathcal{H}_{q,g_4}(h, m, F,$
$g_1^{2r_1+3c_1} g_2^{4r_1+5c_1} g_3^{r_1} g_4^{r_2} h^{-c_1},$
$g_1^{r_3+4r_4-10c_2} g_2^{2r_3-3r_4+3c_2} g_3^{r_4} g_4^{r_5} h^{-r_3},$
$g_1^{5r_6+8r_7-11r_8-c} g_2^{r_7} g_3^{r_8} g_4^{r_9} h^{-r_6})$

Figure 3.8: Protocol for Example 3.6.4.

3.7 Optimizations and extensions

Modulo a small constant factor, the communication and computation complexity of our proof techniques for atomic formulae are the same as for proofs with full disclosure. In the latter case, $\mathcal{P}$ would simply transmit $(x_1, \ldots, x_l)$ to $\mathcal{V}$, and digitally sign $\mathcal{V}$'s challenge message using the Schnorr or Guillou-Quisquater signature scheme, say. Thus, our selective disclosure techniques for atomic formulae achieve privacy essentially for free.

This extreme efficiency does not hold when our techniques are used to demonstrate formulae of the form (3.6) or (3.7), although the increase in communication and computation complexity is only linear in the number of logical connectives. We now describe a slight optimization of our techniques for these two cases. To demonstrate an atomic formula, our techniques have $\mathcal{P}$ demonstrate knowledge of a secret key corresponding to a public key, where the definition of the public key and of what constitutes a secret key both depend on the formula demonstrated; see Propositions 3.3.1 and 3.4.1. In this interpretation, "complex" propositions of the form

"I know a secret key corresponding to public key h and it satisfies formula F"

(with h fixed a priori and F atomic) are mapped to simple propositions of the form

"I know a secret key corresponding to public key h_i,"

where the "distorted" public key h_i is not fixed a priori but derived from h in a formula-dependent manner, and the definition of what constitutes a secret key corresponding to h_i also depends on F. Cramer, Damgård, and Schoenmakers [123] show how to demonstrate monotone Boolean formulae over such simple propositions.[4] Since the witness-indistinguishable proof of knowledge in Section 2.4.3 satisfies their requirements [123, Corollary 14], their technique can be applied here. The idea is to dictate the restrictions, according to which $\mathcal{P}$ generates its self-chosen challenges from the challenge message, in accordance with a secret-sharing construction due to Benaloh and Leichter [25] for the access structure defined by the "dual" of the formula. Since this technique does not require F to be expressed in the conjunctive normal form (3.7), the resulting proof of knowledge may be more compact. On the downside, lifting the optimized demonstration protocols to the smartcard setting (see Chapter 6) is not always possible without significantly increasing the complexity of the protocol. Therefore, we will not consider this optimization any further.

Another optimization is for $\mathcal{V}$ to batch-process all the verification relations, one for each atomic (sub)subformula, in the manner described in Section 2.5.3. Batch-verification may also be applied to the verification of multiple protocol executions.

It is possible to use our showing protocol techniques in such a manner that $\mathcal{P}$ can rapidly demonstrate possession of t out of u "qualitative" attributes, for any

[4]De Santis, Di Crescenzo, Persiano, and Yung [334] independently devised a similar technique.

$t \leq u \leq l$. Namely, if each of $x_1, \ldots, x_u$ is guaranteed to be either 1 or 0, then $\mathcal{P}$ can simply demonstrate that $\sum_{i=1}^{u} x_i = t \bmod q$. In Chapter 5 we will see that the guarantee that each x_i is either 0 or 1 can come from a CA that encodes the attribute values into $\mathcal{P}$'s key pair.

The atomic propositions for which Boolean formulae can be demonstrated can be extended beyond linear relations:

- A technique of Damgård [127] can be adapted to our scenario in order to demonstrate polynomial relations. Demonstration of

  $$x_1^{\gamma_1} + \beta_2 x_2^{\gamma_2} + \cdots + \beta_k x_l^{\gamma_l} = \beta_1 \bmod q$$

  requires $\mathcal{P}$ to spawn in the order of $\sum_{i=1}^{l} \gamma_i$ auxiliary commitments, and to perform two basic proofs of knowledge for each of these. (Either a proof of knowledge of a representation or a proof of equality of two secrets; the latter can be handled using a protocol of Chaum and Pedersen [109] that will also be used in Section 4.5.2.) This is practical only for low-degree polynomials. [5]

- Brickell, Chaum, Dåmgard, and van de Graaf [63] show how to demonstrate that a secret is contained in an interval. Their technique, which can be adapted to our scenario, is not very practical because it requires polynomially many repetitions of a three-move protocol with binary challenges. Moreover, the interval for which $\mathcal{P}$ must perform the proof must be three times larger than the interval one is interested in, to avoid leakage of information (so that $\mathcal{P}$ is actually demonstrating a different statement).

  An improvement is due to Schoenmakers [342]. With $h = g_1^{x_1} g_2^{\alpha}$, say, to demonstrate that $x_1 \in \{0, \ldots, 2^t - 1\}$ the prover discloses t auxiliary commitments

  $$h_0 := g_1^{b_1} g_2^{\alpha_1}, \ldots, h_{t-1} := g_1^{b_t} g_2^{\alpha_t}.$$

  The b_i's satisfy $\sum_{i=0}^{t-1} b_i 2^i = x_1$ and the α_i's are chosen at random subject to the condition

  $$\alpha = \sum_{i=0}^{t-1} \alpha_i 2^i \bmod q.$$

  The prover proves that each $b_i \in \{0, 1\}$, using our technique in Section 3.5, and the verifier in addition checks that

  $$\prod_{i=0}^{t-1} h_i^{2^i} = h.$$

  Generalization to arbitrary intervals is accomplished by proving that x_1 is in the intersection of two appropriately shifted intervals, each of length a power

[5] Fujisaka and Okamoto [179] proposed another technique that is equally impractical and less elegant.

of 2. This technique requires the prover to spawn a number of auxiliary commitments that is linear in the size of the interval, and to perform essentially two Schnorr proofs of knowledge for each of these.

Demonstrating an atomic proposition in both cases involves a serious amount of overhead, and the practical relevance of demonstrating polynomial relations is unclear. Therefore we will not consider these techniques any further.

3.8 Bibliographic notes

The technique in Section 3.3 for demonstrating a single linear relation is due to Brands [54]. The general techniques for demonstrating arbitrary Boolean propositions, for atomic propositions that are linear relations, originate from Brands [45], with a summary in Brands [55]. None of the formal security statements and their proofs have appeared elsewhere previously, nor have the examples.

The suggestion in Section 3.7 to adapt a technique of Damgård [127] to demonstrate polynomial relations is due to Brands [55], and was later on used by Camenisch and Stadler [70, 73]. They [74] also rediscovered the simple technique in Section 5.2.2 for demonstrating properties of the attributes encoded into different key pairs, due to Brands [46, page 22] (see also Brands [54]); the resulting proof system is significantly less practical and flexible, though.

The techniques in this chapter have numerous applications that do not necessarily involve digital certificates. For example, the techniques in Sections 3.3 and 3.4 can be used to improve the undeniable signature scheme of Chaum, van Heijst, and Pfitzmann [111, 112]. The technique in Section 3.5, which was inspired by a technique of Schoenmakers [340] to prove knowledge of a secret key corresponding to at least one of two public keys, has been applied by Cramer, Franklin, Schoenmakers, and Yung [124] and Cramer, Gennaro, and Schoenmakers [125] to design “key escrow” electronic voting schemes (see the Epilogue). Other applications that need not involve digital certificates but could benefit from our showing protocol techniques include fair exchange of digital signatures, incremental signing, digital watermarking, private information retrieval, and distributed database querying. In the remaining chapters we will confine ourselves to applications involving digital certification.