Zero-Knowledge Proofs of Possession of Digital Signatures and Its Applications

Khanh Quoc Nguyen[1], Feng Bao[2], Yi Mu[1], and Vijay Varadharajan[1]

[1] School of Computing & Information Technology
University of Western Sydney, Nepean
PO Box 10, Kingswood, NSW 2747, Australia
{qnguyen,yimu,vijay}@cit.nepean.uws.edu.au
[2] Kent Ridge Digital Labs
21 Heng Mui Keng Terrace, Singapore 119597
baofeng@krdl.org.sg

Abstract. Demonstrating in zero-knowledge the possession of digital signatures has many cryptographic applications such as anonymous authentication, identity escrow, publicly verifiable secret sharing and group signature. This paper presents a general construction of zero-knowledge proof of possession of digital signatures. An implementation is shown for discrete logarithm settings. It includes protocols of proving exponentiation and modulo operators, which are the most interesting operators in digital signatures. The proposed construction is applicable for ElGamal signature scheme and its variations. The construction also works for the RSA signature scheme. In discrete logarithm settings, our technique is $O(l)$ times more efficient than previously known methods.

1 Introduction

1.1 The Problem

Demonstrating the knowledge of secrets, which satisfy some specific relations while revealing no useful information about the secrets, is a major cryptographic primitive. It is used to realize many practical cryptographic protocols. In Camenisch-Stadler group signature[5] a group signature on the message m is a non-interactive proof of the signer's group membership certificate which is a RSA signature signed by the group manager. In identity escrow[17], users have to prove the possession of valid identity certificates which are digital signatures on some known messages. Some other protocols to deal with publicly verifiable secret sharing [25] and fair exchange of digital signatures[1] could also be considered as this type of problems. This paper focuses on zero-knowledge proof of secrets that form a valid digital signature.

1.2 Related Works

A digital signature scheme is referred to as a specific arithmetic relation. Theoretically, protocols to demonstrate any arithmetic relation can be derived from

V. Varadharajan and Y. Mu (Eds.): ICICS'99, LNCS 1726, pp. 103–118, 1999.

zero-knowledge proof for NP-language[16] and can be converted to proof of knowledge[14]. Those protocols are very inefficient because they require to encode the relations into a Boolean circuit and apply the zero-knowledge proof for each gate. The common methodology is to commit the secrets in some commitments, form an arithmetic circuit corresponding to the relation and then prove that the committed secrets satisfy the desired relations using the circuit.

There exist general constructions for zero-knowledge protocols to prove linear relations connected by Boolean operators[3] and polynomial relations[9,15]. However, proving such relations is relatively easy as all commitments are computed in the same finite field. For digital signature relations, proving knowledge possession is not an easy task. This is because commitments of signatures are often computed in different finite fields. If there is an exponentiation in digital signatures of which both the base and exponent are the secrets to be demonstrated, the proof becomes much more difficult.

1.3 Our Contribution

This paper presents a general construction of zero-knowledge proof of possession of digital signatures. The construction yields zero-knowledge proofs for many digital signature schemes, including ElGamal, DSA. Nyberg-Rueppel and RSA signatures.

Our construction can be briefly described as follows. We first break the relation into a corresponding arithmetic circuit, that consists of gates, each is only a standard arithmetic relation. We then compute a commitment for all the secrets and the output of every gate. The protocol to prove the digital signature relation is a set of procedures proving that the committed inputs yield the committed output in all gates.

Thus our first task is to form a language $\mathcal{L}$ of all basic arithmetic relations required in digital signatures and show how to create an arithmetic circuit for any relation of $\mathcal{L}$. The remaining task is to construct building blocks of proving all those standard arithmetic relations that include addition, multiplication, exponentiation, modulo and equality. The building blocks are dependent on the underlying setting. We give a realization of these building blocks in discrete logarithm settings. Our general protocol for discrete logarithms is $O(l)$ times more efficient than current methods. Other modes of proofs and settings can be achieved with some slight modifications.

The general construction is not only useful for digital signature relations but can also be used to prove many other arithmetic relations. For instance, linear and polynomial relations between secrets committed in the same finite field can be demonstrated with addition and multiplication relation protocols. The equality relation protocol can turn the problem of showing the relations between secrets committed in different finite fields into the less difficult problem of proving the relations between secrets committed in the same finite field.

The Outline of the Paper: The construction of language $\mathcal{L}$ and the general zero-knowledge protocol to prove any arithmetic relation in $\mathcal{L}$ are given in section 2. Section 3 presents the realization of the general protocol for discrete logarithm

settings. It also discusses zero-knowledge proofs in other settings and models. The zero-knowledge protocols to prove possession of digital signatures are discussed in section 4.

2 General Construction

This section gives the description of our setting and the general zero-knowledge protocol to prove arithmetic relations. The construction is independent of any particular implementation.

2.1 Notations and Settings

For any number m, let Z_m denote the finite field of $0, ..., m-1$. Let $\mathcal{L} = \{F_i(x_1, ..., x_n) | i = 0, ..\}$ be the language of all functions $F_i(x_1, ..., x_n)$ computed over the finite field Z_{p_i} parsed according to the following BNF rules:

$$F_j ::= c | x_i | F_j + F_j | F_j * F_j | F_j^{F_k} | F_j \bmod p_i$$

Here c is some constant, F_j represents an expression of $\mathcal{L}$ computed in Z_{p_j}, F_k in Z_{p_k} for any j, k. Using these parsing rules, one can form an arithmetic circuit corresponding to any member $F_i()$ of $\mathcal{L}$. For convenience, let the arithmetic circuit of function $F_i()$ be *the circuit* $F_i()$. Let each node in the circuit $F_i()$ corresponding to $F_j = (F_j + F_j)$, $(F_j * F_j)$, $(F_j^{F_k})$ and $(F_j \bmod p_i)$ be addition, multiplication, exponentiation and modulo gate respectively. The node corresponds to $F_j = x_i$ is referred to as a leaf. Note that gates are computed over many different finite fields Z_{p_i} $(i = 0, ..)$ and there might be several exponentiation gates in the circuit $F_i()$ computed over the *same* group Z_{p_j} but with a different order p_k.

2.2 Protocol Description

We now give a general construction for zero-knowledge proof of possession of the secrets $s_1, ..., s_n$ for any function $F_i()$ that $F_i(s_1, ..., s_n) = y$ for a public constant y. We borrow the model from [9] and assume the existence of the following building blocks:

1. **Commitment scheme** lets the prover $\mathcal{P}$ commit a number $0 \leq a < p_i$ in a commitment $A = \texttt{commit}(a)$ in such a way that the verifier $\mathcal{V}$ cannot open the commitment to get the value a while the prover is unable to find two different values a corresponding to the same commitment A.
2. **Checking Protocol** allows $\mathcal{P}$ to prove in zero-knowledge to $\mathcal{V}$ the knowledge of the secret a for a given commitment A.
3. **Addition Gate Protocol** allows $\mathcal{P}$ to convince $\mathcal{V}$ in zero-knowledge that a secret is the sum of two other secrets given three commitments computed over Z_{p_i}.
4. **Multiplication Gate Protocol** allows $\mathcal{P}$ to convince $\mathcal{V}$ in zero-knowledge that a secret is the product of the two other secrets given three commitments computed over Z_{p_i}.

5. **Exponentiation Gate Protocol** allows $\mathcal{P}$ to convince $\mathcal{V}$ in zero-knowledge the exponentiation relation $c = a^b \bmod q_i (q_i = p_j)$ for the secrets a, b, c concealed in the commitments A, B, C respectively. A, C are computed in Z_{p_i} and B is computed in Z_{p_j}.
6. **Modulo Gate Protocol** allows $\mathcal{P}$ to convince $\mathcal{V}$ in zero-knowledge that the secrets a_i, a_j concealed in commitment A_i and A_j respectively, equal modulo q_j, i.e., $a_i - a_j = 0 \bmod q_j$. A_i is computed in Z_{p_i} while A_j is computed in Z_{p_j} ($q_i > q_j$, both are members of $\{p_i | i = 0, ..\}$).
7. **Equality Protocol** allows $\mathcal{P}$ to prove the equality of the two secrets concealed in two commitments, each computed in different finite fields and/or different multiplicative groups.

Then the protocol is executed as follows:

Protocol Description

STEP 0
First, $\mathcal{P}$ and $\mathcal{V}$ agree upon a circuit $F_i()$. This circuit is known and verified by both parties such that the circuit is correctly formed from the function $F_i()$. For convenience, let $L_1, ..., L_u$ be the leave and $G_1, ..., G_v$ be the gates of the circuit $F_i()$, G_v is the final gate and its output is y.

STEP 1
$\mathcal{P}$ makes u commitments $U_1, ..., U_u$ and v commitments $V_1, ..., V_v$ such that U_j contains the value of L_j and V_i contains the output value of G_i. All commitments are then sent to $\mathcal{V}$. $\mathcal{P}$ then uses the checking protocol for the commitments $U_1, ..., U_u$ to prove to $\mathcal{V}$ that $\mathcal{P}$ can open all of them[1].

STEP 2
For each gate, $\mathcal{P}$ proves the relation between the secrets concealed in the commitments representing the inputs and output of the gate using one of Addition, Multiplication, Exponentiation and Modulo Gate protocols.

STEP 3
If the two commitments U_i, U_j concealed the same secret $s_i = s_j$ computed in two different finite fields Z_{p_i} and Z_{p_j}, $\mathcal{P}$ uses the equality protocol to prove the equality of the secrets.

STEP 4
$\mathcal{P}$ opens the commitment V_v representing the final gate G_v to show the concealed secret y. $\mathcal{V}$ accepts the proof if and only if the circuit is correctly formed in Step 0, all the proofs in Steps 1-3 are valid and $\mathcal{P}$ opens the commitment V_v to reveal y.

[1] [9] requires $\mathcal{P}$ to prove for commitments $V_1, ..., V_v$ as well. This is unnecessary.

PROOF:
The completeness of the protocol is straightforward from the inspection of the protocol. The proof of soundness appears in Appendix A.

3 A Realization for Discrete Logarithm Setting

To give an implementation of our protocol, we show the realization of the building blocks used in the general construction. This section presents those building blocks in computational zero-knowledge mode for discrete logarithm settings. Here p_i are all primes larger than 2^l for some security parameter l ($l > 80$).

We now give the constructions of the commitment scheme, checking, addition-gate, multiplication-gate, exponentiation-gate, modulo-gate and equality protocols. Two protocols for special cases: *multiplication with a constant factor* and *exponentiation with a constant base* of multiplication-gate and exponentiation-gate, are also given for efficiency. For presentation's sake, we give the protocols that demonstrate the relation between secrets of every gate in three different blocks according to the relations demonstrated, namely modulo, polynomial and exponentiation .

In this section, let p, q denote some generic prime numbers chosen from $\{p_i | i = 0, 1, ...\}$ such that there is a multiplicative group G_q of order q over Z_p. The constants g, h are chosen randomly in G_q such that $\log_g(h)$ in Z_p is not known to the prover. The symbol $\in_R$ means a uniformly random choice.

3.1 Commitment Scheme

A commitment of a value x in Z_p is constructed as $\texttt{commit}(x, r) = g^x h^r \bmod p$ to $\mathcal{V}$, here $r \in_R Z_p$. There are two phases in this scheme:

Commit to a number x in Z_p, $\mathcal{P}$ simply sends $\texttt{commit}(x, r) = g^x h^r \bmod p$ to $\mathcal{V}$.

Open a commitment $\texttt{commit}(x, r)$ is done by sending x, r to $\mathcal{V}$. $\mathcal{V}$ can easily check the correctness of the committed values x, r with the commitment.

This scheme is unconditionally hiding. Its security is proven to be equivalent to the intractable discrete logarithm problem[22]. Unless $\mathcal{P}$ knows $\log_g(h)$ in Z_p, it is infeasible to find two different sets of input ($x, r \in Z_q$) that produce the same output $\texttt{commit}(x, r)$.

3.2 Checking Protocol

The checking protocol is zero-knowledge and convinces $\mathcal{V}$ that $\mathcal{P}$ knows the secret x for the given commitment $w = \texttt{commit}(x, r)$.

Protocol

1. $\mathcal{P}$ chooses $\alpha, \beta \in_R Z_q$ and sends $W = g^\alpha h^\beta \bmod p$ to $\mathcal{V}$.
2. $\mathcal{V}$ chooses $c \in_R Z_q$ and sends c to $\mathcal{P}$.

3. $\mathcal{P}$ sends $\mathcal{V}$ two responses $u = \alpha - cx$ and $v = \beta - cr \bmod q$.
4. $\mathcal{V}$ accepts the protocol if and only if $W = w^c g^u h^v \bmod p$.

This protocol is well-known and its security is no weaker than that of Schnorr's signature[24].

3.3 Modulo Related Protocols

Modulo related protocols demonstrate modulo and equality relations between secrets. Here commitments are computed in different group orders/finite fields. We now present the range-check protocol given in [21] and the equality proof protocol with an unknown group order. They are used in the construction of the modulo gate and equality protocols.

Range Check Protocol

`Given:` commitment $w = \texttt{commit}(x, r)$, a value $t < q$.
`Prove:` $x \in [0, t]$.

Protocol (run in parallel l times)

1. First both parties agree upon a suitable positive integer e, roughly equals $t/3 - 1$.
2. $\mathcal{P}$ chooses $\alpha_1 \in [0, e]$, and sets $\alpha_2 = \alpha_1 - e$. $\mathcal{P}$ sends to $\mathcal{V}$ the unordered pair of commitments $W_1 = \texttt{commit}(\alpha_1, \rho_1)$ and $W_2 = \texttt{commit}(\alpha_2, \rho_2)$.
3. $\mathcal{V}$ challenges $\mathcal{P}$ by $c \in_R \{0, 1\}$.
4. If $c = 0$, $\mathcal{P}$ sends to $\mathcal{V}$ the values of $\alpha_1, \alpha_2, \rho_1, \rho_2$.
   If $c = 1$, $\mathcal{P}$ sends to $\mathcal{V}$ the value of $x + \alpha_j, r + \rho_j$ such that $x + \alpha_j \in [e, t - e]$.
5. $\mathcal{V}$ verifies $W_j = \texttt{commit}(\alpha_j, \rho_j)$ $(j = 1, 2)$ in the former and $wW_j = \texttt{commit}$ $(x + \alpha_j, r + \rho_j)$, $x + \alpha_j \in [e, t - e]$ in the latter case.

This protocol allows $\mathcal{P}$ to convince $\mathcal{V}$ that the commitment $w = \texttt{commit}(x, r)$ indeed satisfies $0 \leq x \leq t$ for some value t. It works regardless whether the order q of the multiplicative group of $[g, h]$ is known. If the order q of the group is unknown, virtually the same protocol can be used[11,21][2]. We refer to this particular version of protocol as *Range Check protocol with unknown group order*. Here the corresponding commitment $w = \texttt{commit}_N(x, r)$ is also computed as $w = g^x h^r \bmod N$ but x, r are chosen in the integer field . Of course, the factoring of N is not known to $\mathcal{P}$.

[2] Another relevant protocol is given in [6]. This protocol is much more efficient. However, it is not suitable for our task as the range of valid secrets is not identical to the range that prover can prove.

Equality Protocol with an Unknown Group Order

`Given:` two commitment $w_p = \texttt{commit}_p(x_p, r_p)$ and $w_N = \texttt{commit}_N(x_N, r_N)$. Here N is a RSA modulo, much larger than p and x_N, r_N are chosen over the positive integer field.
`Prove:` $x_p = x_N = x$ $(x \in Z_q)$.

Protocol (run in parallel l rounds)

1. First both parties agree upon a suitable positive integer e which roughly equals $q/3 - 1$.
2. $\mathcal{P}$ chooses $\alpha_1 \in [0, e]$, and sets $\alpha_2 = \alpha_1 - e$. $\mathcal{P}$ sends to $\mathcal{V}$ the unordered set of commitments (W_1, S_1), (W_2, S_2) computed as $W_i = \texttt{commit}_p(\alpha_i, \rho_i)$ and $S_i = \texttt{commit}_N(\alpha_i, \rho_i)$ $(i = 1, 2)$.
3. $\mathcal{V}$ challenges $\mathcal{P}$ a bit $c \in_R [0, 1]$.
4. If $c = 0$, $\mathcal{P}$ sends to $\mathcal{V}$ the values of $\alpha_1, \alpha_2, \rho_1, \rho_2$.
   Otherwise, $\mathcal{P}$ sends to $\mathcal{V}$ the values of $x+\alpha_j, r+\rho_j$ such that $x+\alpha_j \in [e, q-e)$.
5. $\mathcal{V}$ verifies
   If $c = 0$, $W_j = \texttt{commit}_p(\alpha_j, \rho_j)$, $S_j = \texttt{commit}_N(\alpha_j, \rho_j)$, $\alpha_j \in [0, e]$ and $\rho_j > 0$ $(j = 1, 2)$.
   If $c = 1$, $w_p W_j = \texttt{commit}_p(x + \alpha_j, r_p + \rho_j)$, $w_N S_j = \texttt{commit}_N(x + \alpha_j, r_N + \rho_j)$ and $x + \alpha_j \in [e, q - e)$.

Security Proof

The **completeness** is straightforward by observing that if $\alpha_2 = \alpha_1 - e$, and $\alpha_1 \in [0, e]$, for at least one i :$x + \alpha_i \in [e, q)$. For the **soundness**, note that in each round $c = 0$, $\mathcal{V}$ is convinced that the same α_k is concealed in both W_i and S_i $(i = 1, 2)$. Because the choice of c is independent of $\mathcal{P}$, in each round $c = 1$, with the probability of $1/2$ the verifier is convinced that the commitments W_i, S_i conceal the same secret α_i. Also in the same round $c = 1$, $\mathcal{V}$ is convinced that $w_p W_i$, $w_N S_i$ conceal the same value $x + \alpha_i \in [e, q - e)$. Commitments $w_N S_i$ is computed in Z_N, thus $x + \alpha_i = \alpha_i + x_N$ or $x = x_N$. In the other hand, $w_p W_i$ is computed in Z_p, thus $x + \alpha_i = x_p + \alpha_i \bmod q$. Moreover $x_p < q$, $\alpha_i \leq e$, thus $x_p + \alpha_i < p + e$. Therefore, $x + \alpha_i \in [e, p - e)$ if and only if $x + \alpha_i = x_p + \alpha_i$ or $x = x_p$. This shows that $\mathcal{V}$ is convinced $x_p = x_N$ with the probability of $1/2$ in each round $c = 1$. With l rounds run in parallel, the verifier is convinced that w_p, w_N conceal the same secret with the probability of roughly $1 - 2^{l/2}$, assuming c is chosen over the toss of a coin.

In the protocol, the view of the verifier is the same as in the range-check protocol. Thus it is zero-knowledge. Formal proof can easily be constructed with a standard simulator.

Using the range check protocol and equality protocol with an unknown group order, we can construct the modulo gate and the equality protocol respectively as follows:

Equality Protocol

`Given:` $w_i = \texttt{commit}_i(x_i, r_i)$ and $w_j = \texttt{commit}_j(x_j, r_j)$. Here $\texttt{commit}_i$ and $\texttt{commit}_j$ are commitments computed in Z_{p_i}, Z_{p_j} respectively.
`Prove:` $x_i = x_j$

Protocol

1. $\mathcal{P}$ and $\mathcal{V}$ agree upon a RSA modulo N that is larger than both p_i, p_j. In computational zero-knowledge model, N is set by a trusted third party. The factoring of N should not be known to $\mathcal{P}$.
2. $\mathcal{P}$ computes $w_N = \texttt{commit}_N(x, r_N)$ $(x = x_i = x_j)$ and sends w_N to $\mathcal{V}$.
3. $\mathcal{P}$ runs two instances of equality protocol with an unknown group order to prove that the secrets concealed in w_i,w_N and w_j, w_N are respectively the same.
4. $\mathcal{V}$ accepts the proof if and only if $\mathcal{V}$ is convinced in step (3).

Modulo Gate Protocol

`Given:` $w_i = \texttt{commit}_i(x_i, r_i)$ and $w_j = \texttt{commit}_j(x_j, r_j)$ and a number t. $\texttt{commit}_i$ and $\texttt{commit}_j$ are defined as for the equality protocol.
`Prove:` $x_i = x_j \bmod t$

Protocol

(without the loss of generality, let us assume that $x_i \geq x_j$)

1. $\mathcal{P}$ and $\mathcal{V}$ agree upon a RSA modulo N as for equality protocol.
2. $\mathcal{P}$ then computes $s_k = \texttt{commit}_N(x_k, \rho_k)$ $(k = i, j)$, $s = \texttt{commit}_N(d, \rho)$ $(d = (x_i - x_j)/t)$ and sends s_i, s_j, s to $\mathcal{V}$.
3. $\mathcal{P}$ runs two instances of the equality protocol with unknown group order to prove the equality of the secrets concealed in w_k, s_k $(k = i, j)$.
4. $\mathcal{P}$ then runs an instance of the range-check protocol to prove that the secret concealed in s is in $[0, q_i/t)$.
5. $\mathcal{V}$ accepts the proof if and only if $s_i = s_j s^t \bmod N$.

3.4 Polynomial Related Protocols

Polynomial related protocols demonstrate addition and multiplication relations. Addition and multiplication with a constant factor relations are proven by simply choosing the commitments corresponding to the input(s) and the output of the gate according to the relation. In an addition gate, let $w_1 = \texttt{commit}(x_1, r_1)$,$w_2 = \texttt{commit}(x_2, r_2)$ and $w_3 = \texttt{commit}(x_3, r_3)$ be the commitments respectively representing the two inputs and the output of the gate. As the commitment scheme is homomorphic, thus by choosing $r_3 = r_1 + r_2$ the prover has already proved the addition relation for the gate. The verifier can verify the correctness of the proof by checking $w_3 = w_1 w_2 \bmod p$. Similarly, in the case of the multiplication gate with a constant factor c, the prover chooses $r_3 = cr_1 \bmod q$. The verifier

can verify the relation with the check $w_3 = w_1^c \bmod p$. It is straightforward to see that both demonstrations are sound, complete and zero-knowledge.

The remaining is a protocol to demonstrate the relation for multiplication gate with both inputs are secret inputs. It is given as follows:

Multiplication Gate Protocol

`Given:` $w_1 = \texttt{commit}(x_1, r_1)$, $w_2 = \texttt{commit}(x_2, r_2)$ and $w_3 = \texttt{commit}(x_3, r_3)$ that respectively correspond to the two inputs and the output of the gate.
`Prove:` $x_3 = x_1 x_2 \bmod q$

Protocol:

1. $\mathcal{P}$ chooses $\alpha, \beta \in_R Z_{i+1}$, sends $W_3 = w_2^\alpha h^\beta$ and $W_2 = g^\alpha h^\beta \bmod p$ to $\mathcal{V}$.
2. Upon receiving W_3, W_2, $\mathcal{V}$ issues a challenge $c \in_R Z_q$ to $\mathcal{P}$.
3. $\mathcal{P}$ computes the responses $u = \alpha - cx_1$, $v_3 = \beta - c(r_3 - r_2 x_1)$ and $v_2 = \beta - cr_1 \bmod q$, then sends u, v_3, v_2 to $\mathcal{V}$.
4. $\mathcal{V}$ accepts the proof if and only if $W_3 = w_3^c w_2^u h^{v_3}$ and $W_2 = w_1^c g^u h^{v_2} \bmod p$.

This is an instance of Chaum-Pedersen[8] proof of equality of discrete logarithms. Here the protocol proves the equality of the discrete logarithm of w_3 to the base w_2, and of w_1 to g in respective representation of w_3 to $[w_2, h]$ and of w_1 to $[g, h]$. This shows that the discrete logarithm of w_3 to g in the representation of w_3 to $[g, h]$ is the product of the discrete logarithms of w_1, w_2 to g in the respective representations of w_1, w_2 both to $[g, h]$, i.e., $x_3 = x_1 x_2$.

3.5 Exponentiation Related Protocols

Exponentiation related protocols demonstrate the relations between commitments corresponding to exponentiation gates. In an exponentiation gate, the exponent is not computed in the same finite field as the base. However, in order to maintain the algebraic relation the value of the base has to be in the multiplicative group of order k over Z_q if the exponent is computed in Z_k ($k = p_t$ for some t in the general setting). Here the exponentiation is calculated in Z_q.

For clarity, we denote the commitments computed in Z_q and Z_p respectively as $\texttt{commit}_p$ and $\texttt{commit}_q$. Secrets committed in $\texttt{commit}_p$ and $\texttt{commit}_q$ are computed in Z_q and Z_k respectively. Similar indices are also used for g, h. Note that for the two different exponentiation gates computed with the same finite field Z_q, it is perfectly legal to have two different values of k in our model.

The base in an exponentiation gate can either be a (public) constant or a secret concealed in a commitment. There is a different protocol for each case.

Exponentiation Gate with Constant Base Protocol

`Given:` a constant b, commitments $w_k = \texttt{commit}_q(x_k, r_k)$ and $w_q = \texttt{commit}_p(x_q, r_q)$
`Prove:` $x_q = b^{x_k} \bmod q$

Protocol (run in parallel, l rounds)

1. $\mathcal{P}$ chooses $\alpha_k, \beta_k \in_R Z_k$, $\beta_q \in_R Z_q$. $\alpha_q = b^{\alpha_k} \bmod q$, sends $W_k = \texttt{commit}_q(\alpha_k, \beta_k)$ and $W_q = \texttt{commit}_p(\alpha_q, \beta_q)$ to $\mathcal{V}$.
2. $\mathcal{V}$ then challenges $\mathcal{P}$ with $c \in_R [0, 1]$
3. $\mathcal{P}$ returns to $\mathcal{V}$ the responses u_k, v_k, v_q computed as $u_k = \alpha_k - cx_k$, $v_k = \beta_k - cr_k \bmod k$ and $v_q = \beta_q - cr_q \bmod q$
4. $\mathcal{V}$ is convinced if and only if $W_k = w_k^c g_q^{u_k} h_q^{v_k}$ and $W_q = w_q^c g_p^{b^u_k} h_p^{v_q}$.

This is a variant of Stadler's protocol[25] to prove the knowledge of double discrete logarithm. Here the protocol is given for the commitment in the form of $w = g^x h^r$ instead of $w = g^x$. Both of these forms are instances of the representation problem of w to a set of bases (e.g. g, h in the former and g in the latter). The security of the representation problem is equivalent to that of the discrete logarithm problem[2].

Exponentiation Gate Protocol

`Given:` $w_1 = \texttt{commit}_p(x_1, r_1)$, $w_2 = \texttt{commit}_q(x_2, r_2)$ and $w_3 = \texttt{commit}_p(x_3, r_3)$ that correspond to the base, exponent and the output of the gate.
`Prove:` $x_3 = x_1^{x_2} \bmod q$

Protocol (run in parallel, l rounds)

1. $\mathcal{P}$ chooses $\alpha, \beta \in_R Z_k$, forms $u_1 = g_p^{h_q^\alpha}$ and $u_2 = g_p^{h_q^\beta}$ and sends u_1, u_2 to $\mathcal{V}$.
2. $\mathcal{V}$ challenges $\mathcal{P}$ a bit $c \in_R [0, 1]$.
3. If $c = 0$, $\mathcal{P}$ sends to $\mathcal{V}$: α, β. Otherwise $\mathcal{P}$ sends to $\mathcal{V}$: $v_1 = x_1 h_q^\alpha$ and $v_2 = x_1^{x_2} h_q^\beta \bmod q$.
4. $\mathcal{V}$ (with the help of $\mathcal{P}$) processes as follows:
   a) If $c = 0$, $\mathcal{V}$ checks the correctness of u_1, u_2 with α and β
   b) If $c = 1$, $\mathcal{P}$ runs two instances of multiplication protocol to prove v_1 (resp. v_2) is the product of secrets concealed in u_1 and w_1 (u_2 and w_3). $\mathcal{P}$ also runs the **sub-protocol** to prove that the same secret x_2 is concealed in both w_2 and $v_2 = \texttt{commit}(x_2, \beta - x_2\alpha)$. Here $\texttt{commit}(x, r) = v_1^x h_q^r \bmod q$ is a commitment scheme computed in Z_q. This scheme is legitimate since the discrete logarithm of v_1 to h is unknown and they are both in the multiplicative group G_k of order k in Z_q.
5. $\mathcal{V}$ accepts the proof if $\mathcal{V}$ is convinced at every step.

The Sub-protocol

`Given:` $w_2 = g_q^{x_2} h_q^{r_2}$ and $v_2 = v_1^{x_2} h_q^r \bmod q$
`Prove:` w_2, v_2 are commitments of the same value x_2

Process:

1. $\mathcal{P}$ chooses $\alpha, \rho_w, \rho_v \in Z_k$, sends $w = g_q^\alpha h_q^{\rho_w}$ and $v = v_1^\alpha h_q^{\rho_v} \bmod q$ to $\mathcal{V}$.
2. $\mathcal{V}$ challenges $\mathcal{P}$ a number $c \in_R Z_k$.

3. $\mathcal{P}$ responses with $u = \alpha - cx_2$, $r_w = \rho_w - cr_2$ and $r_v = \rho_v - cr \bmod q$.
4. $\mathcal{V}$ accepts the proof if and only if $w = w_2^c g_q^u h_q^{r_w}$ and $v = v_2^c v_1^u h_q^{r_v}$ hold.

This sub-protocol is an instance of Chaum-Pedersen's proof of equality of discrete logarithms. It proves that the discrete logarithm part of w to g_q in the representation of w to the base $[g_q, h_q]$ equals the discrete logarithm part of v to v_1 in the representation of v to $[v_1, h_q]$. Note h_q is chosen such that $\log_{h_q}(v_1)$ is not known.

Security Proof

The completeness comes straight from the inspection of the protocol. For the soundness, note that in each round $c = 0$, the verifier is convinced with the probability of 1 that $\mathcal{P}$ knows the double discrete logarithm of both u_1, u_2. Because c is chosen by $\mathcal{V}$ over the toss of a coin, in each round $c = 1$ $\mathcal{V}$ is convinced with probability of $1/2$ that $\mathcal{P}$ knows the discrete logarithm α, β of both $u_1 = g_p^{h^\alpha}$ and $u_2 = g_p^{h^\beta}$.

The two instances of the multiplication protocol convince $\mathcal{V}$ that v_1 (resp. v_2) is the product of the secrets committed in w_1 and u_1 (w_3 and u_2), i.e., $v_1 = x_1 \log_{g_p} u_1 = x_1 h^\alpha$ and $v_2 = x_3 \log_{g_p} u_2 = x_3 h^\beta$. Since the discrete logarithm of x_1 to h is not known, v_2 is a commitment of $\log_{x_1} x_3$.

The *sub-protocol* then proves that the secrets concealed in both v_2 and w_2 are the same. This means $\mathcal{V}$ is convinced that x_2 equals the discrete logarithm of x_3 to x_1 or $x_3 = x_1^{x_2} \bmod q$ with the probability of $1/2(1 - 2^l)$ for each round ($c = 1$). After l rounds, $\mathcal{V}$ is convinced with the probability of roughly $1 - 2^{l/2}$ that $x_3 = x_1^{x_2}$, assuming c is chosen uniformly as either 1 or 0.

In each round for $c = 0$, the verifier learns nothing about the secrets because all information shown by the prover, is independent of w_1, w_2, w_3. In each round for $c = 1$, $\mathcal{P}$ executes an instance of equality protocol and two instances of multiplication protocol. The multiplication protocol and the sub-protocol are zero-knowledge and reveals no useful information to the verifier. All instances are independent of others. The rest of verifier's view is v_2, v_1. As α, β are chosen at random, $v_1 = x_1 h^\alpha$ and $v_2 = x_1^{x_2} h^\beta$ are witness-indistinguishable. Thus at no stage, the verifier learns any useful information about the secrets.

3.6 Discussion

This construction is unconditionally hiding and computational interactive honest-verifier zero-knowledge. Non-interactive zero-knowledge is achieved by forming the challenge c as the hash value of information sent to $\mathcal{V}$ by $\mathcal{P}$. Unconditionally binding, in which the computational power of the prover is not limited, is achieved by applying the technique of [9]. Honest-verifier can also be loosen by standard techniques[12].

With some slight modification, similar building blocks can be built in RSA setting[3] for any mode of proofs. The technique of [15] can be used to construct our building blocks for multiplication, addition, modulo gate for the commitments computed in multiplicative groups of finite fields with unknown group order(s). Moreover, if the secret is a single value, i.e., there is no descendant gate(s) of the corresponding gate, exponentiation gates with constant base can

also be demonstrated for such commitments. It is possible to add boolean relations to our general circuit using the construction of [3]. Especially *AND* relation can be shown at no extra cost. Other arithmetic relations, such that `div`, `mod` with secret modulus, can be achieved with similar techniques.

4 Zero-Knowledge Proof of Possession of Digital Signatures

Zero-knowledge proof of possession of digital signatures can be constructed if the underlying signature scheme is in $\mathcal{L}$. It is done by first forming an arithmetic circuit corresponding to the underlying signature scheme and then using the general technique to realize zero-knowledge proofs.

Signature schemes that do not require hash functions can easily be shown to be members of $\mathcal{L}$. They include RSA, ElGamal[13], DSS[18] and Nyberg-Rueppel[20] signature schemes. Let $f()$ be the signature function, m be the message and S be the signature. The function $f()$ takes S as its input and outputs m or some isomorphic value of m under some one-way function. In RSA, $\mathcal{P}$ proves the knowledge of $S = s$: $f(s) = m$ for $f(x) = x^d$ and d is the public key. In ElGamal signature, $\mathcal{P}$ proves the knowledge of $S = (r, s)$: $f(r, s) = g^m$ for $f(\alpha, \beta) = y^\alpha \alpha^\beta \bmod p$ and y is the public key. In Nyberg-Rueppel signature, $\mathcal{P}$ proves the knowledge of $S = (r, s)$: $f(r, s) = m$ for $f(\alpha, \beta) = g^{\alpha \bmod q} y^\beta \beta$. See the appendix for detailed circuit constructions of these signature schemes

4.1 Comparison to Previous Works

RSA-based signatures can be proven in zero-knowledge without modulo related nor exponentiation related protocols. Our construction achieves virtually the same efficiency as those presented in [5,15].

Kilian and Petrank gave the only zero-knowledge proof of possession of discrete-logarithm based digital signatures in [17]. Their protocol is specifically designed for ElGamal signature scheme, while ours are applicable for many signature schemes. Nevertheless, with the security parameter l, Kilian and Petrank protocol requires $O(l^2)$ rounds. No protocol in our overall construction requires more than $O(l)$ rounds. All (sub) protocols are independent and thus can be done in parallel. Therefore the overall complexity of our protocol to prove the possession of digital signatures is $O(l)$. This is an improvement of the order $O(l)$ over Kilian-Petrank protocol. Recently Camenisch and Michels[4] proposed a new general construction that can prove any arithmetic relation. But in proving possession of digital signatures, their proposal requires a minimum of $O(l)$ rounds and is no better efficient than Kilian-Petrank protocol. It is still highly desirable to construct protocols with $O(1)$ rounds. This is achieved if cut-and-choose is eliminated in modulo and exponentiation related protocols. Such solutions are unfortunately not found yet.

4.2 Applications

Our zero-knowledge proofs of possession of digital signatures have many applications. An obvious application is identity-escrow. Here identity certificates can be of any mentioned signature schemes. The identity escrow then consists of two parts. One is the zero-knowledge proof of possession of a valid identity certificate, i.e., a signature. The other is the ciphertext of some information that uniquely identify the certificate under some trusted authority public key. Similarly, our proof can realize fix-size group signature schemes. So far, fix-size groups signatures exist only for RSA-based group certificates. Those group signatures are non-interactive proofs of possession of RSA signatures. Our construction allows group signatures to be constructed with group certificates being any mentioned signature.

Another significant application is fair-exchange of digital signatures. The standard method requires the encryption of the digital signatures under some trusted authority public keys. The sender then proves the validity of the ciphertext to the receiver. This is proposed by Asokan, Shoup and Waidner[1]. In their protocols, only "half" of the signature is encrypted and proven in zero-knowledge, the other half is shown directly to the receiver. It could be undesirable if the sender is required to be anonymous/unlinkable when the exchange fails. Our technique facilitates complete hiding. It is done by encrypting every part of the signature under the trusted authority public key. Each ciphertext is then considered as a unconditionally binding commitment. Then the sender proves in zero-knowledge that the corresponding plaintext is a valid signature of a specific message. The whole signature is not revealed unless the exchange succeeds.

References

1. N. Asokan, V. Shoup and M. Waidner, *Optimistic fair exchange of digital signatures*, Proceedings of EUROCRYPT'98, LNCS 1403, pp.591-606.
2. S. Brands, *Untraceable off-line cash based on the representation problem*, Technical Report CS-R9323, Centrum voor Wiskunde en Informatica, April 1993.
3. S. Brands, *Rapid Demonstration of Linear Relations Connected by Boolean Operators*, Proceedings of EUROCRYPT'97, LNCS 1223, pp.318-333.
4. J. Camenisch and M. Michels, *Proving in Zero-Knowledge that a Number is the Product of Two Safe Primes*. BRICS Technical Report RS-98-29, To appear in Eurocrypt'99.
5. J. Camenisch and M. Stadler, *Efficient Group Signatures for Large Groups*, Proceedings of Crypto'97, LNCS 1294, pp. 465-479.
6. A. Chan, Y. Frankel and T. Tsiounis, *Easy come-easy go divisible cash*, Proceedings of EUROCRYPT'98, LNCS, Finland, pp. 561-575.
7. D. Chaum, E. van Heijst and B. Pfitzmann, *Cryptographically Strong Undeniable Signature, Unconditionally Secure for the Signer*, Proceedings of Crypto'91, LNCS 576, pp.204-212.
8. D. Chaum and T.P. Pedersen, *Wallet databases with observers*, Proceedings of Crypto'92, LNCS 740, pp. 89-105.
9. R. Cramer and I. Damgard, *Zero-Knowledge Proofs for Finite Field Arithmetic or: Can Zero-Knowledge be for Free?*, Proceedings of Crypto'98, to appear.

10. R. Cramer and V. Shoup, *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack*, Proceedings of Crypto'98, to appear.
11. I. Damgard, *Practical and Provably Secure Release of a Secret and Exchanges of Signatures*, Journal of Cryptology, vol. 8, n. 4, 1995, pp.201-222.
12. G. Di Crescenzo, T. Okamoto and M. Yung *Keeping the SZK-Verifier Honest Un conditionally*, Proceedings of Crypto'97, LNCS 1294, pp.31-45.
13. T. ElGamal, *A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*, IEEE Transactions on Information Theory, v. IT-31, n. 4, 1985, pp. 469-472.
14. U. Feige, A. Fiat and A. Shamir, *Zero-knowledge proofs of Identity*, Journal of Cryptology 1988, vol. 1, pp.77-94.
15. E. Fujisaki and T.Okamoto, *Statistical Zero-Knowledge Protocols to Prove Modular Polynomial Relations*, Proceedings of Crypto'97, LNCS 1294, pp. 16-30.
16. O. Goldreich, S. Micali and A. Wigderson, *Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design*, Proceedings of Foundation of Computer Science 1986, pp. 174-187.
17. J. Kilian and E. Petrank, *Identity Escrow*, Proceedings of Crypto'98, to appear.
18. National Institute of Standards and Technology, NIST FIPS PUB 186, *Digital Signature Standard*, US Department of Commerce, May 1994.
19. A. Menezes, P. van Oorschot and S. Vanstone *Handbook of Applied Cryptography*, CRC Press, 1997.
20. K. Nyberg and R.A Rueppel, *Message Recovery for Signature Schemes Based on the Discrete Logarithm Problem*, Proceedings of EUROCRYPT'94, LNCS 950, pp.182-193.
21. T. Okamoto, *An efficient divisible electronic cash scheme*, Proceedings of Crypto'95, LNCS 963, pp.438-451.
22. T. Pedersen, *Non-Interactive and Information Theoretic Secure Verifiable Secret Sharing*, Proceedings of Crypto'91, LNCS 576, pp. 129-140.
23. B. Schneider, *Applied Cryptography*, 2nd edition, John Wiley & Sons, Inc, 1996.
24. C. Schnorr, *Efficient Signature Generation for Smart Cards*, Proceedings of Crypto 89, LNCS 435, pp. 239-252.
25. M. Stadler, *Publicly Verifiable Secret Sharing*, Proceedings of EUROCRYPT'96, LNCS 1070, pp.190-199.

A Soundness Proof for Protocol in Section 2.2

To prove the soundness of the protocol proposed in section 2.2, we make three assumptions:

1. The circuit is constructed correctly from the function $F_i()$ with the cheating probability of 0. This is plausible as such construction is public and $F_i()$ is public.
2. The circuit is of polynomial size, i.e., both number of gates and leaves are of polynomial order.
3. All the building blocks are secure and have a negligible cheating probability.

We now prove the cheating probability of the protocol ϵ is negligible.

Lemma 1. *For each building block, let the combined cheating probability of the input(s) be ϵ_I, the cheating probability of the output be ϵ_O and of the protocol be ϵ_G, The following inequality holds:*

$$\epsilon_O \leq \epsilon_I + \epsilon_G$$

This is because for the inputs are correct for with the probability of $(1 - \epsilon_I)$. Accordingly the cheating probability of the output when the inputs are correct, is $\epsilon_O = (1 - \epsilon_I)\epsilon_G$. Combined this with the cheating probability ϵ_I of inputs, we have:

$$\epsilon_O = \epsilon_I + (1 - \epsilon_I)\epsilon_G \leq \epsilon_I + \epsilon_G$$

Here for any probability ϵ, it satisfies $0 \leq \epsilon \leq 1$. Similarly, we can also obtain the following lemma:

Lemma 2. *Let the cheating probability of the inputs A, B be ϵ_A, ϵ_B, the combined cheating probability of the inputs be ϵ_I, the inequality holds:*

$$\epsilon_I \leq \epsilon_A + \epsilon_B$$

As in each of our building block there are at most two inputs, the cheating probability of the output is always no greater than the sum of all the cheating probabilities of the inputs and the protocol itself for any stance of the building blocks. This comes straightforward from two lemmas above.

In our circuit $F_i()$, the inputs of each gate are the outputs of other gates unless they are the leaves. The final output of all the gates is G_v and all the non-reducible inputs are $U_1, ..., U_u$. Each gate G_i is associated with a protocol $\mathcal{G}_i$ to prove the relation between the secrets concealed in the input and the output commitments of the gate. Each leaf L_j is associated with a protocol $\mathcal{L}_i$ proving the knowledge of the secret concealed in the associated commitment U_j. Hence, we have the following result:

Lemma 3. *Let θ_i, λ_j, ε and ϵ be the cheating probability of $\mathcal{G}_i$, $\mathcal{L}_j$, the opening commitment required in step 4 and the whole protocol respectively. The following inequality holds*

$$\epsilon \leq \varepsilon + \sum_{i=1}^{v} \theta_i + \sum_{j=1}^{u} \lambda_j$$

From assumption (2) and (3), the cheating probability of the building blocks are negligible, $u + v$ are of polynomial size, thus the cheating probability of the protocol (ϵ) is no greater than a polynomial order of a negligible value. That proves the soundness of the protocol.

B Arithmetic Circuits of Digital Signature Schemes

Let $f()$ be the signature function, m be the message. The function $f()$ takes the signature as it input and outputs m or some isomorphic value of m under some one-way function. The construction of the arithmetic circuit corresponding to $f()$ is as follows:

The Case of RSA Family

`Function f():` $f(x) = x^d \bmod n$
`Verification:` $f(s) = m$, s is the signature
`Circuit construction:` $d = d_1..d_k$ $(d_i \in [0, 1])$.

$$f(x) := \sum_{i=1}^{k} (f_i(x))^{d_i} \bmod n$$
$$f_i(x) := (f_{i-1}(x))^2 \bmod n$$
$$f_1(x) := x$$

The Case of ElGamal Digital Signature Scheme[13]

`Function f():` $f(\alpha, \beta) = y^\alpha \alpha^\beta \bmod p$ (y is public key)
`Verification:` $f(r, s) = g^m$, (r, s) is the signature
`Circuit construction:`

$$f(\alpha, \beta) := f_1(\alpha, \beta) f_2(\alpha, \beta) \bmod p$$
$$f_1(\alpha, \beta) := y^{f_3(\alpha,\beta)} \bmod p$$
$$f_2(\alpha, \beta) := f_4(\alpha, \beta)^{f_5(\alpha,\beta)} \bmod p$$
$$f_3(\alpha, \beta) := \alpha \bmod (p - 1)$$
$$f_4(\alpha, \beta) := \alpha$$
$$f_5(\alpha, \beta) := \beta$$

The circuit construction for Digital Signature Algorithm[18] is identical to that of ElGamal. The only difference in the circuit is that $f_3(\alpha, \beta)$ is computed as $\alpha \bmod q$, not $\alpha \bmod (p - 1)$. Other variants of ElGamal[19] could be proven using similar circuit.

The Case of Nyberg-Rueppel Digital Signature Scheme[20]

`Function f():` $f(\alpha, \beta) = g^\alpha \bmod {}^q y^\beta \beta$
`Verification:` $f(r, s) = m$, (r, s) is the signature
`Circuit construction:`

$$f(\alpha, \beta) := f_1(\alpha, \beta) f_2(\alpha, \beta) f_3(\alpha, \beta) \bmod p$$
$$f_1(\alpha, \beta) := g^{f_4(\alpha,\beta)} \bmod p$$
$$f_2(\alpha, \beta) := y^{f_3(\alpha,\beta)} \bmod p$$
$$f_3(\alpha, \beta) := \beta \bmod q$$
$$f_4(\alpha, \beta) := \alpha$$

Note that the checks $0 \leq \alpha < q$ and $0 < \beta < 0$ are also satisfied. This is due to their commitments are computed in respective finite fields. To prove $\beta > 0$, the modulo protocol should use the range $[1, q - 1]$ instead of $[0, q - 1]$ as presented in section 3.3.