Note: Sections 2.5 and 2.6 are based on the first edition of "Cryptography Theory and Practice", which was published in 1995, and they have not been updated for this course.

2.5 The Okamoto Identification Scheme

In this section, we present a modification of the **Schnorr Scheme** due to Okamoto. This modification can be proved secure, assuming the intractibility of computing a particular discrete logarithm in $\mathbb{Z}_p$.

To set up the scheme, the TA chooses p and q as in the **Schnorr Scheme**. The TA also chooses two elements $\alpha_1, \alpha_2 \in {\mathbb{Z}_p}^*$ both having order q. The value $c = \log_{\alpha_1} \alpha_2$ is kept secret from all the participants, including Alice. We will assume that it is infeasible for anyone (even a coalition of Alice and Olga, say) to compute the value c. As before, the TA chooses a signature scheme and hash function. The certificate issued to Alice by the TA is constructed as described in Protocol 2.9.

Protocol 2.9: ISSUING A CERTIFICATE TO ALICE

1. The TA establishes Alice's identity and issues an identification string $\mathrm{ID}(\mathrm{Alice})$.
2. Alice secretly chooses two random exponents a_1, a_2, where $0 \leq a_1, a_2 \leq q-1$. Alice computes
$$v = {\alpha_1}^{-a_1}{\alpha_2}^{-a_2} \bmod p$$
and gives v to the TA.
3. The TA generates a signature
$$s = sig_{\mathrm{TA}}(\mathrm{ID}(\mathrm{Alice}), v).$$
The certificate
$$\mathbf{C}(\mathrm{Alice}) = (\mathrm{ID}(\mathrm{Alice}), v, s)$$
is given to Alice.

The **Okamoto Identification Scheme** is presented in Protocol 2.10.

Protocol 2.10: THE OKAMOTO IDENTIFICATION SCHEME

1. Alice chooses random numbers k_1, k_2, where $0 \leq k_1, k_2 \leq q-1$, and computes
$$\gamma = {\alpha_1}^{k_1}{\alpha_2}^{k_2} \bmod p.$$
2. Alice sends her certificate $\mathbf{C}(\mathrm{Alice}) = (\mathrm{ID}(\mathrm{Alice}), v, s)$ and γ to Bob.
3. Bob verifies the signature of the TA by checking that $ver_{\mathrm{TA}}(\mathrm{ID}(\mathrm{Alice}), v, s) = \mathrm{true}$.
4. Bob chooses a random number r, $1 \leq r \leq 2^t$ and gives it to Alice.
5. Alice computes
$$y_1 = k_1 + a_1 r \bmod q$$
and
$$y_2 = k_2 + a_2 r \bmod q$$
and gives y_1 and y_2 to Bob.
6. Bob verifies that
$$\gamma \equiv {\alpha_1}^{y_1}{\alpha_2}^{y_2}v^r \pmod p.$$

Here is an example of the **Okamoto Scheme**.

Example 2.3 As in previous examples, we will take $p = 88667$, $q = 1031$, and $t = 10$. Suppose $\alpha_1 = 58902$ and $\alpha_2 = 73611$ (both α_1 and α_2 have order q in ${\mathbb{Z}_p}^*$). Now, suppose $a_1 = 846$ and $a_2 = 515$; then $v = 13078$.

Suppose Alice chooses $k_1 = 899$ and $k_2 = 16$; then $\gamma = 14574$. If Bob issues the challenge $r = 489$ then Alice will respond with $y_1 = 131$ and $y_2 = 287$. Bob will verify that

$$58902^{131} 73611^{287} 13078^{489} \equiv 14574 \pmod{88667}.$$

So Bob will accept Alice's proof of identity. $\square$

The proof that the protocol is complete (i.e., that Bob will accept Alice's proof of identity) is straightforward. The main difference between Okamoto's and Schnorr's scheme is that we can prove that the **Okamoto Scheme** is secure provided that the computation of the discrete logarithm $\log_{\alpha_1} \alpha_2$ is intractible.

The proof of security is quite subtle. Here is the general idea: As before, Alice identifies herself to Olga polynomially many times by executing the protocol. We then suppose (hoping to obtain a contradiction) that Olga is able to learn some information about the values of Alice's secret exponents a_1 and a_2. If this is so, then we will show that (with high probability) Alice and Olga together will be able to compute the discrete logarithm c in polynomial time. This contradicts the assumption made above, and proves that Olga must be unable to obtain any information about Alice's exponents by taking part in the protocol.

The first part of this procedure is similar to the soundness proof for the **Schnorr Scheme**.

THEOREM 2.1 *Suppose Olga knows a value γ for which she has probability $\epsilon \geq 1/2^{t-1}$ of successfully impersonating Alice in the verification protocol. Then, in polynomial time, Olga can compute values b_1 and b_2 such that*

$$v \equiv {\alpha_1}^{-b_1}{\alpha_2}^{-b_2} \pmod{p}.$$

PROOF For a fraction ϵ of the 2^t possible challenges r, Olga can compute values y_1, y_2, z_1, z_2, r and s with $r \neq s$ and

$$\gamma \equiv {\alpha_1}^{y_1}{\alpha_2}^{y_2}v^r \equiv {\alpha_1}^{z_1}{\alpha_2}^{z_2}v^s \pmod{p}.$$

Define

$$b_1 = (y_1 - z_1)(r - s)^{-1} \bmod q$$

and

$$b_2 = (y_2 - z_2)(r - s)^{-1} \bmod q.$$

Then it is easy to check that

$$v \equiv {\alpha_1}^{-b_1}{\alpha_2}^{-b_2} \pmod{p},$$

as desired. ∎

We now proceed to show how Alice and Olga can together compute the value of c.

THEOREM 2.2 *Suppose Olga knows a value γ for which she has probability $\epsilon \geq 1/2^{t-1}$ of successfully impersonating Alice in the verification protocol. Then, with probability $1 - 1/q$, Alice and Olga can together compute $\log_{\alpha_1} \alpha_2$ in polynomial time.*

PROOF By Theorem 2.1, Olga is able to determine values b_1 and b_2 such that

$$v \equiv {\alpha_1}^{-b_1}{\alpha_2}^{-b_2} \pmod{p}.$$

Now suppose that Alice reveals the values a_1 and a_2 to Olga. Of course

$$v \equiv {\alpha_1}^{-a_1}{\alpha_2}^{-a_2} \pmod{p},$$

so it must be the case that

$${\alpha_1}^{a_1-b_1} \equiv {\alpha_2}^{b_2-a_2} \pmod{p}.$$

Suppose that $(a_1, a_2) \neq (b_1, b_2)$. Then $(a_2 - b_2)^{-1} \bmod q$ exists, and the discrete log

$$c = \log_{\alpha_1} \alpha_2 = (a_1 - b_1)(b_2 - a_2)^{-1} \bmod q$$

can be computed in polynomial time.

There remains to be considered the possibility that $(a_1, a_2) = (b_1, b_2)$. If this happens, then the value of c cannot be computed as described above. However, we will argue that $(a_1, a_2) = (b_1, b_2)$ will happen only with very small probability $1/q$, so the procedure whereby Alice and Olga compute c will almost surely succeed.

Define

$$\mathcal{A} = \{(a'_1, a'_2) \in \mathbb{Z}_q \times \mathbb{Z}_q : {\alpha_1}^{-a'_1}{\alpha_2}^{-a'_2} \equiv {\alpha_1}^{-a_1}{\alpha_2}^{-a_2} \pmod p\}.$$

That is, $\mathcal{A}$ consists of all the possible ordered pairs that could be Alice's secret exponents. Observe that

$$\mathcal{A} = \{(a_1 - c\theta, a_2 + \theta) : \theta \in \mathbb{Z}_q\},$$

where $c = \log_{\alpha_1} \alpha_2$. Thus $\mathcal{A}$ consists of q ordered pairs.

The ordered pair (b_1, b_2) computed by Olga is certainly in the set $\mathcal{A}$. We will argue that the value of the pair (b_1, b_2) is independent of the value of the pair (a_1, a_2) that comprises Alice's secret exponents. Since (a_1, a_2) was originally chosen at random by Alice, it must be the case that the probability that $(a_1, a_2) = (b_1, b_2)$ is $1/q$.

So, we need to say what we mean by (b_1, b_2) being "independent" of (a_1, a_2). The idea is that Alice's pair (a_1, a_2) is one of the q possible ordered pairs in the set $\mathcal{A}$, and no information about which is the "correct" ordered pair is revealed by Alice identifying herself to Olga. (Stated informally, Olga knows that an ordered pair from $\mathcal{A}$ comprises Alice's exponents, but she has no way of telling which one.)

Let's look at the information that is exchanged during the identification protocol. Basically, in each execution of the protocol, Alice chooses a γ; Olga chooses an r; and Alice reveals y_1 and y_2 such that

$$\gamma \equiv {\alpha_1}^{y_1}{\alpha_2}^{y_2}v^r \pmod p.$$

Recall that Alice computes

$$y_1 = k_1 + a_1 r \bmod q$$

and

$$y_2 = k_2 + a_2 r \bmod q,$$

where

$$\gamma = {\alpha_1}^{k_1}{\alpha_2}^{k_2} \bmod p.$$

But note that k_1 and k_2 are not revealed (nor are a_1 and a_2).

The particular quadruple (γ, r, y_1, y_2) that is generated during one execution of the protocol appears to depend on Alice's ordered pair (a_1, a_2), since y_1 and y_2 are defined in terms of a_1 and a_2. But we will show that each such quadruple could equally well be generated from any other ordered pair $(a'_1, a'_2) \in \mathcal{A}$. To see this, suppose $(a'_1, a'_2) \in \mathcal{A}$, i.e., $a'_1 = a_1 - c\theta$ and $a'_2 = a_2 + \theta$, where $0 \leq \theta \leq q - 1$. We can express y_1 and y_2 as follows:

$$\begin{aligned} y_1 &= k_1 + a_1 r \\ &= k_1 + (a'_1 + c\theta)r \\ &= (k_1 + rc\theta) + a'_1 r, \end{aligned}$$

and

$$\begin{aligned} y_2 &= k_2 + a_2 r \\ &= k_2 + (a'_2 - \theta)r \\ &= (k_2 - r\theta) + a'_2 r, \end{aligned}$$

where all arithmetic is performed in $\mathbb{Z}_q$. That is, the quadruple (γ, r, y_1, y_2) is also consistent with the ordered pair (a'_1, a'_2) using the random choices $k'_1 = k_1 + rc\theta$ and $k'_2 = k_2 - r\theta$ to produce (the same) γ. We have already noted that the values of k_1 and k_2 are not revealed by Alice, so the quadruple (γ, r, y_1, y_2) yields no information regarding which ordered pair in $\mathcal{A}$ Alice is actually using for her secret exponents. This completes the proof. ∎

This security proof is certainly quite elegant and subtle. It would perhaps be useful to recap the features of the protocol that lead to the proof of security. The basic idea involves having Alice choose two secret exponents rather than one. There are a total of q pairs in the set $\mathcal{A}$ that are "equivalent" to Alice's pair (a_1, a_2). The fact that leads to the ultimate contradiction is that knowledge of two different pairs in $\mathcal{A}$ provides an efficient method of computing the discrete logarithm c. Alice, of course, knows one pair in $\mathcal{A}$; and we proved that if Olga can impersonate Alice, then Olga is able to compute a pair in $\mathcal{A}$ which (with high probability) is different from Alice's pair. Thus Alice and Olga together can find two pairs in $\mathcal{A}$ and compute c, which provides the desired contradiction.

Here is an example to illustrate the computation of $\log_{\alpha_1} \alpha_2$ by Alice and Olga.

Example 2.4 As in Example 2.3, we will take $p = 88667, q = 1031$ and $t = 10$, and assume that $v = 13078$.

Suppose Olga has determined that

$$\alpha_1{}^{131}\alpha_2{}^{287}v^{489} \equiv \alpha_1{}^{890}\alpha_2{}^{303}v^{199} \pmod{p}.$$

Then she can compute

$$b_1 = (131 - 890)(489 - 199)^{-1} \bmod 1031 = 456$$

and

$$b_2 = (287 - 303)(489 - 199)^{-1} \bmod 1031 = 519.$$

Now, using the values of a_1 and a_2 supplied by Alice, the value

$$c = (846 - 456)(519 - 515)^{-1} \bmod 1031 = 613$$

is computed. This value c is in fact $\log_{\alpha_1} \alpha_2$, as can be verified by calculating

$$58902^{613} \bmod 88667 = 73611.$$

□

Finally, we should emphasize that, although there is no known proof that the **Schnorr Scheme** is secure (even assuming that the discrete logarithm problem is intractible), neither is there any known weakness in the scheme. Actually, the **Schnorr Scheme** might be preferred in practice to the **Okamoto Scheme** simply because it is somewhat faster.

2.6 The Guillou-Quisquater Identification Scheme

In this section, we describe another identification scheme, due to Guillou and Quisquater, that is based on **RSA**.

The set-up of the scheme is as follows: The TA chooses two primes p and q and forms the product $n = pq$. The values of p and q are secret, while n is public. As is usually the case, p and q should be chosen large enough that factoring n is intractible. Also, the TA chooses a large prime integer b which will function as a security parameter as well as being a public RSA encryption exponent; to be specific, let us suppose that b is a 40-bit prime. Finally, the TA chooses a signature scheme and hash function.

The certificate issued to Alice by the TA is constructed as described in Protocol 2.11.

Protocol 2.11: ISSUING A CERTIFICATE TO ALICE

1. The TA establishes Alice's identity and issues an identification string $\mathrm{ID(Alice)}$.
2. Alice secretly chooses an integer u, where $0 \le u \le n-1$. Alice computes
$$v = (u^{-1})^b \bmod n$$
and gives v to the TA.
3. The TA generates a signature
$$s = sig_{\mathrm{TA}}(\mathrm{ID(Alice)}, v).$$
The certificate
$$\mathbf{C}(\mathrm{Alice}) = (\mathrm{ID(Alice)}, v, s)$$
is given to Alice.

When Alice wants to prove her identity to Bob, say, Protocol2.12 is executed.

Protocol 2.12: THE GUILLOU-QUISQUATER IDENTIFICATION SCHEME

1. Alice chooses a random number k, where $0 \le k \le n-1$ and computes
$$\gamma = k^b \bmod n.$$
2. Alice gives Bob her certificate $\mathbf{C}(\mathrm{Alice}) = (\mathrm{ID(Alice)}, v, s)$ and γ.
3. Bob verifies the signature of the TA by checking that $ver_{\mathrm{TA}}(\mathrm{ID(Alice)}, v, s) = \mathrm{true}$.
4. Bob chooses a random number r, $0 \le r \le b-1$ and gives it to Alice.
5. Alice computes
$$y = ku^r \bmod n$$
and gives y to Bob.
6. Bob verifies that
$$\gamma \equiv v^r y^b \pmod{n}.$$

We will prove that the **Guillou-Quisquater Scheme** is sound and complete. However, the scheme has not been proved to be secure (even assuming that the **RSA** cryptosystem is secure).

Example 2.5 Suppose the TA chooses $p = 467$ and $q = 479$, so $n = 223693$. Suppose also that $b = 503$ and Alice's secret integer $u = 101576$. Then she will compute

$$\begin{aligned} v &= (u^{-1})^b \bmod n \\ &= (101576^{-1})^{503} \bmod 223693 \\ &= 89888. \end{aligned}$$

Now, let's assume that Alice is proving her identity to Bob and she chooses $k = 187485$; then she gives Bob the value

$$\begin{aligned} \gamma &= k^b \bmod n \\ &= 187485^{503} \bmod 223693 \\ &= 24412. \end{aligned}$$

Suppose Bob responds with the challenge $r = 375$. Then Alice will compute

$$\begin{aligned} y &= ku^r \bmod n \\ &= 187485 \times 101576^{375} \bmod 223693 \\ &= 93725 \end{aligned}$$

and gives it to Bob. Bob then verifies that

$$24412 \equiv 89888^{375} 93725^{503} \pmod{223693}.$$

Hence, Bob accepts Alice's proof of identity. $\square$

As is generally the case, proving completeness is quite simple:

$$\begin{aligned} v^r y^b &\equiv (u^{-b})^r (ku^r)^b \pmod n \\ &\equiv u^{-br} k^b u^{br} \pmod n \\ &\equiv k^b \pmod n \\ &\equiv \gamma \pmod n. \end{aligned}$$

Now, let us consider soundness. We will prove that the scheme is sound provided that it is infeasible to compute u from v. Since v is formed from u by RSA encryption, this is a plausible assumption to make.

THEOREM 2.3 *Suppose Olga knows a value γ for which she has probability $\epsilon > 1/b$ of successfully impersonating Alice in the verification protocol. Then, in polynomial time, Olga can compute u.*

PROOF For some γ, Olga can compute values y_1, y_2, r_1, r_2 with $r_1 \neq r_2$, such that

$$\gamma \equiv v^{r_1} {y_1}^b \equiv v^{r_2} {y_2}^b \pmod{n}.$$

Suppose, without loss of generality, that $r_1 > r_2$. Then we have

$$v^{r_1 - r_2} \equiv (y_2/y_1)^b \pmod{n}.$$

Since $0 < r_1 - r_2 < b$ and b is prime, $t = (r_1 - r_2)^{-1} \bmod b$ exists, and it can be computed in polynomial time by Olga using the Euclidean algorithm. Hence, we have that

$$v^{(r_1 - r_2)t} \equiv (y_2/y_1)^{bt} \pmod{n}.$$

Now,

$$(r_1 - r_2)t = \ell b + 1$$

for some positive integer ℓ, so

$$v^{\ell b + 1} \equiv (y_2/y_1)^{bt} \pmod{n},$$

or equivalently,

$$v \equiv (y_2/y_1)^{bt} (v^{-1})^{\ell b} \pmod{n}.$$

Now raise both sides of the congruence to the power $b^{-1} \bmod \phi(n)$, to get the following:

$$u^{-1} \equiv (y_2/y_1)^t (v^{-1})^{\ell} \pmod{n}.$$

Finally, compute the inverse modulo n of both sides of this congruence, to obtain the following formula for u:

$$u = (y_1/y_2)^t v^{\ell} \bmod n.$$

Olga can use this formula to compute u in polynomial time. ∎

Example 2.6 As in the previous example, suppose that $n = 223693$, $b = 503$, $u = 101576$ and $v = 89888$. Suppose Olga has learned that

$$v^{401} 103386^b \equiv v^{375} 93725^b \pmod{n}.$$

She will first compute

$$\begin{aligned} t &= (r_1 - r_2)^{-1} \bmod b \\ &= (401 - 375)^{-1} \bmod 503 \\ &= 445. \end{aligned}$$

Next, she calculates

$$\begin{aligned}\ell &= \frac{(r_1 - r_2)t - 1}{b}\\ &= \frac{(401 - 375)445 - 1}{503}\\ &= 23.\end{aligned}$$

Finally, she can obtain the secret value u as follows:

$$\begin{aligned}u &= (y_1/y_2)^t v^\ell \bmod n\\ &= (103386/93725)^{445} 89888^{23} \bmod 223693\\ &= 101576.\end{aligned}$$

Thus Alice's secret exponent has been compromised. □

2.6.1 Identity-based Identification Schemes

The **Guillou-Quisquater Identification Scheme** can be tranformed into what is known as an *identity-based identification scheme*. This basically means that certificates are not necessary. Instead, the TA computes the value of u as a function of Alice's ID string, using a public hash function h with range $\mathbb{Z}_n$. This is done as indicated in Protocol 2.13.

Protocol 2.13: ISSUING A VALUE u TO ALICE

1. The TA establishes Alice's identity and issues an identification string $\mathrm{ID(Alice)}$.
2. The TA computes

$$u = (h(\mathrm{ID(Alice)})^{-1})^a \bmod n$$

   and gives u to Alice.

The identification protocol now works as described in Figure 2.14.

Protocol 2.14: THE GUILLOU-QUISQUATER IDENTITY-BASED SCHEME

1. Alice chooses a random number k, where $0 \leq k \leq n-1$ and computes
$$\gamma = k^b \bmod n.$$
2. Alice gives $\mathrm{ID}(\mathrm{Alice})$ and γ to Bob.
3. Bob computes
$$v = h(\mathrm{ID}(\mathrm{Alice})).$$
4. Bob chooses a random number r, $0 \leq r \leq b-1$ and gives it to Alice.
5. Alice computes
$$y = ku^r \bmod n$$
and gives y to Bob.
6. Bob verifies that
$$\gamma \equiv v^r y^b \pmod{n}.$$

The value v is computed from Alice's ID string via the public hash function h. In order to carry out the identification protocol, Alice needs to know the value of u, which can be computed only by the TA (assuming that the **RSA** cryptosystem is secure). If Olga tries to identify herself as Alice, she will not succeed because she does not know the value of u.