THE BLUM-GOLDWASSER PKC

1. Description

Efficient probabilistic technique, semantically secure assuming the intractability of integer factorization. Smaller message expansion than Goldwasser-Micali — only $\leq \lfloor \lg n \rfloor$ additional bits.

Idea: a pseudorandom bit stream (from the Blum-Blum-Shub pseudorandom number generator) is XORed with the plaintext. The private key is used to recover the random seed used by the sender to initialize the PRNG.

Public key: $\{n\}$, where $n = pq$ for p, q prime $p \equiv q \equiv 3 \pmod 4$. Such an n is said to be a *Blum integer*.

Private key: $\{p, q, a, b\}$, where $ap + bq = 1$ with $a, b \in \mathbb{Z}$.

B encrypts M to send to A as follows:

(1) Let $k = \lfloor \lg n \rfloor$ and $h = \lfloor \lg k \rfloor \geq 1$. Represent M as a string $M = (m_1 m_2 \ldots m_t)$ of length t where each m_i is a binary string of length h.
(2) Select a seed x_0 which is a random quadratic residue modulo n (simply select a random $r < n$ and put $x_0 \equiv r^2 \pmod{n}$).
(3) For $i = 1, \ldots, t$:
  (a) Compute $x_i \equiv x_{i-1}^2 \pmod{n}$.
  (b) Let p_i be the least h significant bits of x_i.
  (c) Compute $c_i = m_i \oplus p_i$.
(4) Compute $x_{t+1} \equiv x_t^2 \pmod{n}$.
(5) Send $C = (c_1 c_2 \ldots c_t, x_{t+1})$ to A.

Note. Only $\lfloor \lg x_{t+1} \rfloor \leq \lfloor \lg n \rfloor$ additional bits transmitted.

A decrypts M from C as follows:

(1) Compute
$$d_1 \equiv \left(\frac{p+1}{4}\right)^{t+1} \pmod{p-1}, \quad d_2 \equiv \left(\frac{q+1}{4}\right)^{t+1} \pmod{q-1}$$
(2) Compute $u \equiv x_{t+1}^{d_1} \pmod{p}$ and $v \equiv x_{t+1}^{d_2} \pmod{q}$. Note that $u \equiv x_0 \pmod{p}$ and $v \equiv x_0 \pmod{q}$, because $p \equiv q \equiv 3 \pmod 4$ and $x_{i-1} = x_i^{(p+1)/4} \pmod{p}$ for $i = 1, \ldots, t+1$.
(3) Compute $x_0 \equiv vap + ubq \pmod{n}$ (application of CRT).
(4) For $i = 1, \ldots, t$:
  (a) Compute $x_i \equiv x_{i-1}^2 \pmod{n}$.
  (b) Let p_i be the h least significant bits of x_i.
  (c) Compute $m_i = p_i \oplus c_i$.
(5) $M = (m_1 m_2 \ldots m_t)$.

Proof that decryption is correct. Since $x_t \in QR_n$, we have $x_t \in QR_p \longrightarrow x_t^{\frac{p-1}{2}} \equiv 1 \pmod{p}$. Thus
$$x_{t+1}^{\frac{p+1}{4}} \equiv (x_t^2)^{\frac{p+1}{4}} \equiv x_t^{\frac{p+1}{2}} \equiv x_t^{\frac{p-1}{2}} x_t \equiv x_t \pmod{p} .$$
Similarly, $x_t^{\frac{p+1}{4}} \equiv x_{t-1} \pmod{p}$, and repeating this argument yields
$$u \equiv x_{t+1}^{d_1} \equiv x_0 \pmod{p}, \quad v \equiv x_{t+1}^{d_2} \equiv x_0 \pmod{q} .$$

By the CRT we get

$$vap + ubq \equiv x_0 \pmod{n},$$

and thus A creates the same random seed x_0 used by B to encrypt. Hence, A can now decrypt C. $\square$

2. Security

Note that any method that breaks the scheme must reveal the parity bit of the x_i (the key).

Theorem 2.1. *Let A_n be an algorithm which given any $x \in QR_n$ returns the parity bit of y where $y^2 \equiv x \pmod{n}$ and $y \in QR_n$. Then A_n can be used to solve the QRP for any $[a] \in \mathbb{Z}_n^*$ with $\left(\frac{a}{n}\right) = 1$.*

Note. The theorem states that if you have an algorithm A_n that can predict the *previous* bit in the key stream, then this algorithm can be used to solve the QRP.

- it can be shown that previous bit prediction resistance provides the same level of security as next bit prediction resistance
- hence, breaking BBS is at least as hard as the QRP.

Proof. Suppose we wish to solve the QRP for some $[a] \in \mathbb{Z}_n^*$. We first determine $x \equiv a^2 \pmod{n}$. We apply A_n to x to get $b = A_n(x)$. Now b is the parity bit of some y where $y^2 \equiv x \pmod{n}$ and $y \in QR_n$. We know $y^2 \equiv a^2 \pmod{n} \longrightarrow n = pq \mid (y-a)(y+a)$. Suppose $p \mid y - a$ and $q \mid y + a$. Then

$$p \mid y - a \longrightarrow y \equiv a \pmod{p} \longrightarrow 1 = \left(\frac{y}{p}\right) = \left(\frac{a}{p}\right)$$

and similarly

$$q \mid y + a \longrightarrow y \equiv -a \pmod{q} \longrightarrow 1 = \left(\frac{y}{q}\right) = \left(\frac{-a}{q}\right) = -\left(\frac{a}{q}\right)$$

and thus $\left(\frac{a}{pq}\right) = \left(\frac{a}{n}\right) = -1$, which is a contradiction. Hence $y \equiv \pm a \pmod{n}$.

- If $y \equiv a \pmod{n}$, then b is the parity bit of a and $a \in QR_n$.
- If $y \equiv -a \pmod{n}$, then $y = n - 1$ and b is the parity bit of y and is *not* the parity bit of a (since n is odd).

Thus, if the parity bit of a equals b, then $a \in QR_n$ and if it does not equal b, then $a \notin QR_n$. $\square$

Disadvantage: scheme is vulnerable to a chosen ciphertext attack. For example, an adversary who wants the decryption of (C, X_{t+1}) can mount a chosen ciphertext attack by obtaining the decryption M' of (A, X_{t+1}) for some random string A of the same length as C. Then $K = A \oplus M'$ is the keystream used to produce C, and $M = C \oplus K$.