The Bit Security of Modular Squaring given Partial Factorization of the Modulos

Benny Chor[†] Oded Goldreich[‡] Shafi Goldwasser[*]
MIT, Laboratory for Computer Science
Cambridge, MA 01239

Abstract — It is known that given a composite integer $N = p_1p_2$ (such that $p_1 \equiv p_2 \equiv 3 \pmod 4$), and q a quadratic residue modulo N, guessing the least significant bit of a square root of q with any non-negligible advantage is as hard as factoring N.

In this paper we extend the above result to multi-prime numbers $N = p_1p_2 \cdots p_l$ (such that $p_1 \equiv p_2 \equiv \cdots \equiv p_l \equiv 3 \pmod 4$). We show that given N and q, a quadratic residue mod N, guessing the least significant bit of a square root of q is as hard as completely factoring N. Furthermore, the difficulty of guessing the least significant bit of the square root of q remains unchanged even when all but two of the prime factors of N, $p_3, \ldots, p_l$, are known.
The result is useful in desiging multi-party cryptographic protocols.

1. Introduction

The problem of factoring large composite integers is perhaps the single most important computational problem in public key cryptography, as is evident from the large number of cryptosystems based on it (e.g. RSA [15], Rabin [13], Williams [18], Goldwasser-Micali [10]). The importance of the factoring problem motivated various research efforts. Among those are

1) Designing more efficient factorization algorithms.
2) Investigating the security of specific bits in the modular squaring function.
3) Investigating factorization algorithms given partial information on the factors [14].

Most of these works have concentrated on composite numbers N which are the product of two primes p_1p_2.

In this paper we investigate the problem of bit security for the modular squaring function with respect to multi-prime composites $N = p_1p_2 \ldots p_l$. The salient property of our work is that we investigate the bit security given partial factorization $p_3, \ldots, p_l$ of N (i.e. all but two

[†] Supported in part by an IBM Graduate Fellowship and a Bantrell Postdoctoral Fellowship.

[‡] Supported in part by a Weizmann Postdoctoral Fellowship. On leave from the Computer Sc. Dept., Technion

[*] Supported in part by an IBM Faculty Development Award (1984) and NSF Grant DCR-8509905.

factors are known). We show that *the partial factorization does not help.* More specifically, any non-negligible advantage in guessing the least significant bit in the $x^2 \pmod N$ function is equivalent to factoring the remaining pair $p_1 p_2$ (and thus totally factor N). In other words, if it is infeasible to factor two prime composites, then it is infeasible to guess the least significant bit in the squaring modulo N function even if one has almost all of N's factors.

Our work extends the results of Alexi, Chor, Goldreich and Schnorr [1], who considered the bit security of RSA and Rabin functions. These two functions are defined with respect to two-prime moduli $N = p \cdot q$. The RSA function is defined as raising to a power e and reducing modulo N (where e and $(p-1)(q-1)$ are relatively prime). Rabin's function is squaring modulo N. The RSA is 1-to-1, while Rabin's function is 4-to-1. This difference is crucial in trying to extend the [1] results to multi-prime moduli. Extending the RSA result to multi-prime moduli is easy, since the extended function is still 1-to-1. In the case of Rabin's function, squaring modulo an l-prime moduli is a 2^l-to-1 function, and dealing with it is more complicated. In this paper, we demonstrate how these complications can be resolved.

Our results have applications in the design of multi-party cryptographic protocols. In particular, it is useful in contexts where partial factorization, but not complete factorization, is released to a subset of the participants, while certain information must still be kept secret. Combining our result with techniques of probabilistic encryption [10,5], arbitrary information can be encoded so that it still remain totally secure, in such circumstances.

The remaining of this paper is organized as follows. In section 2 we introduce notations and terminology. In section 3 we review previous related results. In section 4 the main result is proved. In section 5 we mention two applications to the design of multi-party cryptographic protocols. We conclude by proposing an open problem.

2. Terminology

We begin this section by presenting some number theoretic terminology which will be used throughout the paper. We proceed by defining a specific class of composite integers which will constitute the domain of our investigation. We conclude this section by formally defining the notion of a "factoring bit".

2.1 Preliminaries

Definition 1: Let N be a natural number. $\mathbf{Z}_N$ will denote the ring of integers modulo N, where addition and multiplication are done modulo N. The length of N will be denoted by n.

Definition 2: Let N be a natural number, and x an integer. $[x]_N$ will denote the remainder of x modulo N (notice that for all x, $0 \leq [x]_N < N$). $L_N(x)$ will denote the least significant bit of

$[x]_N$ in the ordinary binary expansion.

Definition 3: Let N be an integer. Then a is said to be a *quadratic residue modulo* N if there exist an integer x such that $x^2 = a \pmod N$. Otherwise, a is said to be a it quadratic non-residue modulo N. Let us denote by Q_N the set of quadratic residues modulo N.

Let $N = p_1 p_2 \cdots p_l$ be a product of l distinct odd primes. Note that a is a quadratic residue modulo N if and only if a is a quadratic residue modulo each of the p_i's.

Definition 4: Let p be an odd prime number, and h an integer relatively prime to p. The *Legendre symbol* $\left(\frac{h}{p}\right)$ is defined to be 1 if h is a quadratic residue modulo p, and -1 otherwise. For $N = p_1 p_2 \cdots p_l$, a product of l distinct odd primes, and h relatively prime to N, the *Jacobi symbol* $\left(\frac{h}{N}\right)$ is defined to be $\prod_{i=1}^{l} \left(\frac{h}{p_i}\right)$.

Even though the definition of the Jacobi symbol uses the factorization of N, it is well known that $\left(\frac{h}{N}\right)$ be easily computed even if N's factorization is not given. Another fact which is used in this paper is the multiplicativity of the Jacobi symbol, namely $\left(\frac{h \cdot h'}{N}\right) = \left(\frac{h}{N}\right) \cdot \left(\frac{h'}{N}\right)$. For further details on these properties and their proofs, see [12, ch. 3].

2.2 Blum Integers

When all the prime factors of $N = p_1 p_2 \ldots p_l$ are congruent to 3 (mod 4), the set of quadratic residues modulo N has an interesting property. Each quadratic residue has ***exactly*** one square root which is a quadratic residue itself. In other words, squaring modulo N is a permutation over Q_N. Blum was the first to point out the cryptographic significance of this fact [3]. Let $BI = \{ N | N = p_1 \cdot p_2 \cdots p_l,\ p_i \equiv 3 \pmod 4,\ 1 \le i \le l \}$, and call $N \in BI$ ***Blum Integers***.

Definition 5: Let $N = p_1 p_2 \cdots p_l$ be in BI, and q be a quadratic residue modulo N. We denote by $\sqrt{q}$ the square root of q which is a quadratic residue itself, namely $\left(\sqrt{q}\right)^2 = q$ and $\sqrt{q} \in Q_N$.

We restrict our attention to $N \in BI$, since for each quadratic residue $q \in Q_N$, $\sqrt{q}$ and the least significant bit of $\sqrt{q}$ are well defined.

2.3 Bit Security for Factoring

Following [6] and [11], we formally define the notion of bit security for factoring. For the definition, recall that n denotes the length of N.

Definition 6: Let O_N be a probabilistic oracle which, given a quadratic residue q (modulo N), outputs a guess, $O_N(q)$, for $L_N(\sqrt{q})$ (this guess might depend on the internal coin tosses of O_N). Let $\epsilon(\cdot)$ be a function from integers into the interval $[0, \frac{1}{2}]$. We say that O_N is a $\epsilon(n)$-oracle if the probability that the oracle is correct, on an input q randomly selected from the set Q_N, is at least $\frac{1}{2} + \epsilon(n)$.

The probability space in the definition is that of all $q \in Q_N$ and all $0-1$ sequences of internal coin tosses, with uniform distribution. Notice that there is no requirements from the oracle if it is fed as input a number in Z_N which is not a quadratic residue.

Definition 7: We say that *the least-significant bit of $\sqrt{\cdot}$ is $\epsilon(n)$-secure* if there is a probabilistic polynomial time algorithm that on input $N, q \in Q_N$ and access to an arbitrary $\epsilon(n)$-oracle for the least significant bit, $\mathcal{O}_N$, computes $\sqrt{q}$.

Remarks: As is customary, we say that an algorithm is polynomial time if its running time is polynomial in its input length. In particular, the run time will be polynomial in n, the length (in binary) of the modulus N. In the last definition, the specific polynomial might depend on $\epsilon(\cdot)$. The same applies to the next definition.

Definition 8: We say that *the least-significant bit of $\sqrt{\cdot}$ is $\epsilon(n)$-secure even if the factorization of N is partially known* if there is a probabilistic polynomial time algorithm that on input $N, q \in Q_N$, some (but not all) the prime factors of N and access to an arbitrary $\epsilon(n)$-oracle for the least significant bit, $\mathcal{O}_N$, computes $\sqrt{q}$.

We will subsequently replace $\epsilon(n)$ by ϵ for notational convenience. However, ϵ will still be a function of n.

3. Previous Results

In this section, we briefly review related previous results by Rabin [13], Blum, Blum and Shub [4], Alexi, Chor, Goldreich and Schnorr [1] and Vazirani and Vazirani [17].

3.1 The Equivalence of Factoring and Extracing Square Roots

Theorem 1 (Rabin): The following problems are probabilistic polynomial time equivalent

1) Factoring a composite integer N product of two primes.
2) Given N and $q \in Q_N$, finding a square root of q.

This Theorem easily extends to multi-prime integers.

3.2 Reducing Square Root Extraction to a Strange Oracle

Following a sequence of results in [11,2,16,9], Alexi, Chor, Goldreich and Schnorr [1] proved $1/poly(n)$-security results for the least significant bit of a variant of the squaring modulo $N = p_1 p_2$ function. Their proof can be broken into two parts. First, a special type oracle, called (ϵ, q)-oracle is defined (see below). It is shown that factoring is in polynomial-time given access to an (ϵ, q)-oracle. Next, it was shown that an $(\epsilon/2, q)$-oracle can be implemented using any ϵ-oracle for the least significant bit of a particular square root.

Definition 9: Let $N \in BI$ and $q \in Q_N$ be a quadratic residue. An (ϵ, q)-oracle is an oracle that on input $s \in Z_N$ outputs $L_N(s \cdot \sqrt{q})$ with probability at least $\frac{1}{2} + \epsilon$. (Here the probability is taken over all possible choices of s and the internal coin tosses of the oracle with uniform probability distribution.)

The following Theorem is implicit in [1].

Theorem 2 (Alexi, Chor, Goldreich and Schnorr): There exists a probabilistic polynomial time algorithm that on input $N = p_1 p_2 \in BI$, $q \in Q_N$ and access to an arbitrary (ϵ, q)-oracle, finds $\sqrt{q}$.

The proof of Theorem 2 is almost identical to the proof in [1] of equivalence between inverting the RSA and guessing its least significant bit. While Theorem 2 deals with two prime composites, it extends to multi-prime composites. Combining the extended Theorems 1 and 2, we get

Corollary 1: There exists a probabilistic polynomial time algorithm that on input $N = p_1 p_2 \cdots p_l$, $q \in Q_N$ and access to an (ϵ, q)-oracle, completely factors M.

It is left to be shown that on input $N \in BI, q \in Q_N$ and access to an ϵ-oracle for $L_N(\sqrt{\cdot})$, an $(\epsilon/2, \cdot)$-oracle can be implemented. This will be discussed in the next subsection.

3.3 Reducing the Strange Oracle to LSB Oracle when $N = p_1 p_2$

In this subsection we deal with implementing an (ϵ, q)-oracle, given access to an ϵ-oracle for $L_N(\sqrt{\cdot})$. The main difficulty lies in the fact that an (ϵ, q)-oracle must perform well when s ranges over Z_N while the ϵ-oracle is guaranteed to perform well only when its input ranges over Q_N.

The approach taken in resolving this difficulty is to map the queries to the (ϵ, q)-oracle into "queries" and "non-queries" to the ϵ-oracle. "Queries" are answered by invoking the ϵ-oracle, while "non-queries" are answered by flipping a coin. This requires the ability to distinguish "queries" from "non-queries". For $N = p_1 p_2$, two alternative implementations of this abstract approach were suggested.

The First Alternative

In [1], a slightly different predicate was considered (and shown to be equivalent to factoring). Instead of $L_N(\sqrt{\cdot})$ (the least significant bit of the square root which is a quadratic residue itself), they considered $B_N(\cdot)$, the least significant bit of the square root which has Jacobi Symbol 1 and is smaller than $N/2$. In the setting of [1] it is easy to test whether $[s \cdot \sqrt{q}]_N < N/2$ and whether the Jacobi Symbol $\left(\frac{s}{N}\right)$ equals 1. Such s's are mapped to "queries".

In the case of two-prime moduli each quadratic residue has a unique square root which satisfies the two conditions. However, in case the modulus has $l > 2$ factors, each quatratic residue has 2^{l-2} roots which satisfy the above two conditions. Thus, the solution of [1] to implementing the (ϵ, q)-oracle does not seem to extend to multi-prime moduli.

The Second Alternative

A different method of implementing the (ϵ, q)-oracle was suggested by Vazirani and Vazirani [17]. They observed that by Blum, Blum and Shub [4], the quadratic residuosity of s modulo a two-prime composite N can be determined by using an ϵ-oracle $\mathcal{O}_N$ for the least significant bit. If $s \in Q_N$ then the ϵ-oracle for $L_N(s\sqrt{q})$ else a coin is flipped.

The advantage of this method is that the square root which is a quadratic residue itself is well defined also for multi-prime Blum integers. So there is hope of extending this method.

Let us recall how quadratic residuosity can be tested using an ϵ-oracle for $L_N(\sqrt{\cdot})$.

Theorem 3 (Blum, Blum and Shub): Let $N = p_1 p_2 \in BI$. There exist a probabilistic polynomial time algorithm that, on input $N, s \in Z_N$ and access to any ϵ-oracle for the least significant bit, $\mathcal{O}_N$, determines whether $s \in Q_N$.

Proof's sketch: If $\left(\frac{s}{N}\right) = -1$ then answer "$s \notin Q_N$". We are left with the case that $\left(\frac{s}{N}\right) = 1$. Consider the following experiment. Randomly select $r \in Q_N$ with uniform probability distribution (this is done by choosing an element in Z_N, with uniform probability, and squaring it). Let b be the oracle's answer on query $[(r \cdot s)^2]_N$. Clearly,

$$s \in Q_N \text{ implies } Pr(b = L_N(r \cdot s)) \geq \frac{1}{2} + \epsilon .$$

On the other hand, if $s \notin Q_N$ then $-r \cdot s \in Q_N$. As is always the case, $L_N(r \cdot s) = 1 - L_N(-r \cdot s)$ and thus

$$s \notin Q_N \text{ implies } Pr(b = L_N(r \cdot s)) \leq \frac{1}{2} - \epsilon .$$

So the two cases $s \in Q_N$ and $s \notin Q_N$ can be distinguished (with high probability) by sampling polynomially many r's. ▮

A crucial point in the proof is that for two-prime moduli $N = p_1 p_2$, $q \in Q_N$ has only two square roots with Jacobi Symbol +1. One of them is $\sqrt{q}$ and the other is $-\sqrt{q}$. This is not the case when N has more then two prime factors. In fact, q has 2^{l-1} square roots which have Jacobi symbol +1. In the next section we show "a way around" this last problem.

4. The Main Result

In this section we implement an (ϵ, q)-oracle, given access an ϵ-oracle to $\mathcal{O}_N$, where N is a multi-prime Blum integer. This, in turn, implies that an ϵ-oracle for the least significant bit, $\mathcal{O}_N$, enables the complete factorization of N.

Theorem 4: Let $N = M p_3 p_4 \cdots p_l$, $M = p_1 p_2$ and $N \in BI$, where the p_i's are distinct odd primes. Then there is a probabilistic polynomial time algorithm that on input N, $q \in Q_N$, $p_3, p_4, \ldots, p_l$ and access to an arbitrary ϵ-oracle for the least significant bit, $\mathcal{O}_N$, implements an $(\epsilon/(2^l + 1), q)$-oracle.

Proof: Let $Q'_N = \{e : \left(\frac{e}{p_1}\right) = \left(\frac{e}{p_2}\right) = -1$ and $\left(\frac{e}{p_i}\right) = 1$ for every $3 \leq i \leq t\}$.
Given $q \in Q_N$, and access to the ϵ-oracle for the least significant bit, $\mathcal{O}_N$, we implement an $(2^{-l} \cdot \epsilon, q)$-oracle as follows. On query $s \in Z_N$, we first compute the Jacobi Symbol $\left(\frac{s}{M}\right)$ and the Legendre Symbols $\left(\frac{s}{p_3}\right), \left(\frac{s}{p_4}\right), \ldots, \left(\frac{s}{p_l}\right)$. If either of the above equals -1 then $s \notin Q_N \bigcup Q'_N$, and we return the outcome of an unbiased coin flip. It remains to deal with $s \in Q_N \bigcup Q'_N$. We consider two cases:

- Case I: The oracle $\mathcal{O}_N$ answers to $L_N(s\sqrt{q})$ are considerably worse for $s \in Q'_N$, compared to $s \in Q'_N$. In this case we first use $\mathcal{O}_N$ to test whether $s \in Q_N$. Our answer to $L_N(s\sqrt{q})$ is $\mathcal{O}_N(s^2 \cdot q)$ if $s \in Q_N$, and a flip of a coin if $s \in Q'_N$.
- Case II: The oracle $\mathcal{O}_N$ answers to $L_N(s\sqrt{q})$ are not considerably worse for $s \in Q'_N$, compared to $s \in Q'_N$. In this case, we answer to $L_N(s\sqrt{q})$ by $\mathcal{O}_N(s^2 \cdot q)$. Intuitively, it does not matter here whether $s \in Q_N$ or $s \in Q'_N$.

To treat the above cases formally, we define the success probabilities of $\mathcal{O}_N$ on query $[r^2]_N$ where $r \in Q_N$ (correspondingly $r \in Q'_N$) is randomly chosen. (The probabilities are taken over $\mathcal{O}_N$'s internal coin tosses.) Let

$$f = Pr(\mathcal{O}_N(r^2) = L_N(r)) \qquad \text{where } r \text{ is randomly chosen in } Q_N$$

$$f' = Pr(\mathcal{O}_N(r^2) = L_N(r)) \quad . \quad \text{where } r \text{ is randomly chosen in } Q'_N \, .$$

By $\mathcal{O}_N$'s definition, $f \geq \frac{1}{2} + \epsilon$, but no a-priori bounds on f' are known.

With overwhelmingly high probability (say $1 - 2^{-n}$), both f and f' can be approximated with good accuracy (say $\epsilon/8$) by the following polynomial time Monte Carlo experiments: To approximate f, randomly select many independent $r \in Q_N$ with uniform probability distribution. (A random $r \in Q_N$ is selected by picking an element of Z_N at random and squaring it modulo N.) Compare $\mathcal{O}_N$'s answer on $[r^2]_N$ with the known $L_N(r)$. To approximate f', randomly select many independent $r \in Q'_N$ with uniform probability distribution, and compare $\mathcal{O}_N$'s answer on $[r^2]_N$ with the known $L_N(r)$. A random $r \in Q'_N$ is selected by picking $r' \in Q_M$ and $r'' \in Q_{p_3 p_4 \cdots p_l}$, at random, setting $r \equiv -r' \pmod{M}$ and $r \equiv r'' \pmod{p_3 p_4 \cdots p_l}$, and computing r by the Chinese Reminder Theorem.

Let us denote the above approximations by $\tilde{f}$ and $\tilde{f}'$ respectively (i.e. $|f - \tilde{f}| < \epsilon/8$ and $|f' - \tilde{f}'| < \epsilon/8$ with overwhelming high probability). We now consider two cases

Case I: $\tilde{f}' < \tilde{f} - \epsilon/2$.

In this case we will use $\mathcal{O}_N$ to test whether $s \in Q_N$. To do that, randomly select $r \in Q_N$ with uniform probability distribution. Let b be the oracle's answer on query $[(r \cdot s)^2]_N$. If $s \in Q_N$ then

$Pr(b = L_N(r \cdot s)) = f$, while if $s \in Q'_N$ then $Pr(b = L_N(r \cdot s)) = f'$. Since $|f - f'| > \epsilon/4$ (with overwhelming probability), the two cases can be distinguished by a Monte-Carlo experiment.

If we have decided that $s \in Q_N$ then we query the oracle on s^2q and return whatever it has answered (i.e. we return $\mathcal{O}_N(s^2q)$). Otherwise, we flip an unbiased coin and return its outcome.

Case II: $\tilde{f}' \geq \tilde{f} - \epsilon/2$.

In this case we will not try to test whether $s \in Q_N$ or $s \in Q'_N$, but rather query $\mathcal{O}_N$ on s^2q and return $\mathcal{O}_N(s^2q)$. Here $f' \geq \frac{1}{2}$ with overwhelming probability.

Probability Analysis

We now analyze the probability that the answer to $L_N(s\sqrt{q})$ produced by the above procedure is correct. The probability space is that of all choices of $s \in Z_N$ and all internal coin tosses with uniform distribution.

The event $s \notin Q_N \bigcup Q'_N$ occurs with probability $1 - 2 \cdot 2^{-l}$ and is always detected. In this case the above procedure is correct with probability exactly one half.

The event $s \in Q_N \bigcup Q_N$ occurs with probability 2^{-l+1}. In Case I, the answer is correct with probability $\frac{1}{2}(\frac{1}{2} + f) \geq \frac{1}{2} + \frac{\epsilon}{2}$ (up to the overwhelmingly small error term of the approximations). In Case II, the answer is correct with probability $\frac{1}{2}(f + f') \geq \frac{1}{2} + \frac{\epsilon}{2}$ (with the same qualification). The overall probability that our procedure is correct is therefore bounded below by

$$\frac{1}{2} + \frac{1}{2^l} \cdot \epsilon - 2^{-n} \quad .$$

Thus, we have implemented an $(\epsilon/(2^l + 1), q)$-oracle ∎

The proof of Theorem 4 shows how to implement an $(\epsilon/2^l, q)$-oracle given an ϵ-oracle for the least significant bit $L_N(\cdot)$, where N has l prime factors. Thus, when $l = O(\log n)$ the advantage of the new oracle is polynomially (in n) related to the advantage of the original one. Combining Corollary 1 and Theorem 4, we get

Corollary 2: Let $N, M \in BI$ such that M divides N. Suppose that M has two prime factors and N has $l = O(\log n)$ distinct prime factors, where n is the length of N. Then the following two tasks (1) and (2) are computational equivalent, and both are polynomial-time reducible to (3).

1) Factoring M.
2) Given M, p_3, p_4,..., p_l (a partial factorization of $N = Mp_3p_4 \cdots p_l$) and $q \in Q_N$, guess $L_N(\sqrt{q})$ with success probability exceeding $\frac{1}{2} + \frac{1}{poly(n)}$.
3) Let $1 \leq k < l$, $N_1, N_2, \ldots, N_k$ such that $N = N_1N_2 \cdots N_k$ and M divides N_1. Given N_1, N_2,..., N_k and $q \in Q_N$, guess $L_N(\sqrt{q})$ with success probability exceeding $\frac{1}{2} + \frac{1}{poly(n)}$.

5. Applications to Protocols Design

Chor, Goldwasser, Micali, and Awerbuch [7] suggested to use a composite number N product of $l = 2^t + 1$ primes in order to "verifiably share" a secret bit among many players, t of which can be untrusty. They suggested two implementations of this scheme: One is based on the RSA, while the other is based on modular squaring. The security of the second implementation relies on the result of this paper. A brief description of the scheme follows.

The secret is the least significant bit of $\sqrt{q}$, where $q \in Q_N$ is a quadratic residue modulo N. After establishing the secret, the dealer distributes "pieces" of it to every participant (one piece per participant). A random split of N corresponds to one piece of the secret bit. Since N has $2^t + 1$ prime factors, it cannot be totally factored with only t pieces. By our result, it is infeasible for t participants to guess the secret $L_N(\sqrt{q})$ with any non-negligible advantage. On the other hand, with overwhelmingly high probability, $3t$ pieces yield the complete factorization of N and allow the recovery of the secret bit.

6. An Open Problem

A crucial condition for proof of Corollary 2, is that the number of prime factors is logarithmic in the length of the modulus. The reason being that the inverting algorithm needs answers for random elements in Z_N, while the ϵ-oracle for least significant bit answers only on $q \in Q_N$. Thus, only a 2^{-l} fraction of the queries are answered, where l is the number of primes in N. Getting around this difficulty will require either a different inverting algorithm or a better analysis of what happens when the oracle is asked on $q \in Z_N - Q_N$.

References

[1] Alexi, W., B. Chor, O. Goldreich, and C.P. Schnorr, "RSA and Rabin Fuctions: Certain Bits are As Hard As The Whole", to appear in *SIAM Jour. on Computing.* Extended abstract in *Proc. of 25th FOCS,* 1984, pp. 449-457.

[2] Ben-Or, M., B. Chor, and A. Shamir, "On the Cryptogrsphic Security of Single RSA Bits", *15th ACM Symp. on Theory of Computation,* April 1983, pp. 421-430.

[3] Blum, M., "Coin Flipping by Telephone", *IEEE Spring COMCON,* 1982.

[4] Blum, L., M. Blum, and M. Shub, "Comparison of Two Pseudo-Random Number Generators", *Advances in Cryptology: Proceedings of Crypto82,* Chaum, D., et al. eds., Plenum Press, 1983, pp. 61-79.

[5] Blum, M., and S. Goldwasser, "An Efficient Probabilistic PKCS as Secure as Factoring", *Advances in Cryptography: Proceedings of Crypto 84,* Springer Verlag, Lecture Notes in

Computer Science (196), 1985, pp. 289-299.

[6] Blum, M., and S. Micali, "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits", *SIAM Jour. on Computing*, Vol. 13, No. 4, November 1984, pp. 850-864.

[7] Chor, B., S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults", *Proc. of 26th FOCS*, 1985, pp. 383-395.

[8] Diffie, W., and M.E. Hellman, "New Directions in Cryptography", *IEEE Trans. on Inform. Theory*, Vol. IT-22, No. 6, November 1976, pp. 644-654.

[9] Goldreich, O., "On the Number of Close-and-Equal Pairs of Bits in a String (with Implications on the Security of RSA's L.s.b.)", MIT/LCS/TM-256, March 1984.

[10] Goldwasser, S., and S. Micali, "Probabilistic Encryption", *Jour. of Computer and System Science*, Vol. 28, No. 2, 1984, pp. 270-299.

[11] Goldwasser, S., S. Micali, and P. Tong, "Why and How to Establish a Private Code on a Public Network", *Proc. of the 23rd IEEE Symp. on Foundation of Computer Science*, November 1982, pp. 134-144.

[12] Niven, I., and H.S. Zuckerman, *An Introduction to the Theory of Numbers*, John Wiley & Sons Inc., (1980).

[13] Rabin, M.O., "Digital Signatures and Public Key Functions as Intractable as Factorization", MIT/LCS/TR-212, 1979.

[14] Rivest, R.L., and A. Shamir, "An Efficient Factoring Algorithm Based on Partial Information", presented in Eurocrypt85, Linz, Austria, April 1985.

[15] Rivest, R.L., A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signature and Public Key Cryptosystems", *Comm. of the ACM* ,Vol.21, February 1978, pp. 120-126.

[16] Vazirani, U.V., and V.V. Vazirani, "RSA Bits are .732 + ϵ Secure", *Advances in Cryptology: Proceedings of Crypto83*, Chaum,D. ed, Plenum Press, 1984, pp. 369-375.

[17] Vazirani, U.V., and V.V. Vazirani, "Efficient and Secure Pseudo-Random Number Generation", *Proc. of 25th FOCS*, 1984, pp. 458-463.

[18] Williams, H.C., "A Modification of the RSA Public-Key Encryption Procedure", *IEEE Trans. Info. Th.*, IT-26 (1980), pp. 726-729.