An *Efficient* Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information

Manuel Blum

Computer Science Department
University of California at Berkeley

Shafi Goldwasser *

Laboratory for Computer Science
Massachusetts Institute of Technology

Abstract

This paper introduces the first probabilistic public-key encryption scheme which combines the following two properties:

(1) **Perfect secrecy with respect to polynomial time eavesdroppers**: For all message spaces, no polynomial time bounded passive adversary who is tapping the lines, can compute any partial information about messages from their encodings, unless **factoring** composite **integers** is in probabilistic polynomial time.

(2) **Efficiecy**: It compares favorably with the deterministic RSA public-key cryptosystem in both **encoding** and **decoding time** and bandwidth expansion.

The security of the system we propose can also be based on the assumption that the RSA function is intractable, maintaining the same cost for encoding and decoding and the same data expansion. This implementation may have advantages in practice.

1. Introduction

Much attention has been devoted recently to investigating the security of cryptographic protocols, cryptographic encryption techniques, and digital signatures methods. Still, the most important problem in public-key cryptography remains how to encrypt both efficiently and securely.

Key Words: probabilistic encryption, partial information, integer factorization, passive adversaries, chosen cyphertext attack.

* Supported in part by IBM Young Faculty Development Award Dated September 1984

It is customary to call an encryption system secure, when it is infeasible to recover the cleartext x from its encryption $E(x)$. However, this does not necessarily mean that it is infeasible to learn some partial information about x from $E(x)$, for x's of interest. In fact, it has been pointed out in ([L],[GM]) that for any deterministic encryption scheme E, such as the RSA or Rabin's schemes, partial information about x can always be computed from $E(x)$. When E is used in cryptographic protocols, finding out this partial infromation can be detremental to the security of the application, as has been shown for the protocols of "Mental Poker" in [SRA] and "Flipping Coins in Many Pockets" in [BD]. Moreover, whenever the messages to be sent are drawn from special message spaces (e.g. the messages can be expressed as known linear combinations of each other) the secrecy of the messages is in question ([B], [H]).

This issue has been rigorously defined and studied by Goldwasser and Micali in [GM]. They introduced the notion of a Public-Key Encryption scheme that is *polynomially secure*: a polynomial time adversary can not find one message m whose encodings he can distinguish from the encodings of a random message. An equivalent formulation of this security requirement is that after the eavesdropper sees an encrypted message passing in the network, his *a posteriori* probability of computing correctly which message is being sent remains roughly the same as his *a priori* probability of guessing which message is being sent before seeing the encrypted message. This implies that seeing the cyphertext does not help the adversary to compute or guess any function of the cleartext, better than he could have before seeing the cyphertext. Public key encryption schemes have been proposed that achieve this security criteria if the Quadratic Residuosity problem is intractable in [GM], and more generally if there exists any trapdoor function by Yao in [Y].

These schemes are probabilistic and are computed in a "bit-by-bit" fashion. In other words, every message has many possible encodings and every bit of a message is encrypted independently. Due to this last propery, these schemes are highly inefficient. If k is the size of the security parameter (e.g. the size of the modulos in the RSA encryption function) then each bit is encoded individually by a k-bit long string in [GM] and even worse in [Y], resulting in at least a k-bit data expansion factor. Moreover, decoding a single bit in [GM] takes $O(k^3)$ operations. In comparison, the RSA [RSA] or Rabin [Ra] encryption schemes, being deterministic block-ciphers, where a message is encrypted as a block of bits, transfroms k bits of cleartext into k bits of cyphertext using $O(k^3)$ operations.

This paper proposes a simple scheme which combines the polynomial security of the probabilitic schemes ([GM], [G], [Y]) and the efficiency of the deterministic schemes ([RSA] and [Ra]). Thus, with no loss of efficiency, this new scheme can be used in applications such as Mental Poker[GM2], Coin Flipping in a Distributed Network[ABCGM] or whenever messages to be sent come from a special message set and are dependent on one another in a publically known way.

Let k be the size of the security parameter. Then, the new scheme transfroms an l-bit long cleartext into an $(l+k)$-bit long ciphertext, with a computational cost of $O(\frac{lk^2}{\log k})$ for encryption and of $O(k^3) + O(\frac{lk^2}{\log k})$ for decryption. When the length of the message $l < k$ our scheme is at least as fast as either the RSA or Rabin schemes and when $l > k$ our scheme is actually slightly faster. Most importantly, it is more secure. We prove that the scheme is polynomially secure, provided that **factoring**(or inverting RSA) is intractable.

A High Level Description of Our System

Our work has gained from and extends the works by Goldwasser-Micali [GM], Blum-Micali [BM], Yao [Y], Blum-Blum-Shub [BBS], Goldwasser-Micali-Tong [GMT] and Chor-Goldreich [CG].

The notion of a cryptographically strong pseudo random bit (CSPRB) generator has been introduced by Blum and Micali in [BM] and extended by Yao in [Y]. Such a generator is a program which takes as input a k-bit random seed and produces as output a k^t-bit sequence, where $t > 0$ is fixed. The output sequences produced by a CSPRB generator are high quality pseudo-random sequences in the following sense: *if the k bit seed is totally unknown*, they cannot be distinguished from truely random sequences of the same length by any statistical test which runs in polynomial in k time.

The key idea in our method will be: to send an l-bit message m, send the exclusive-or of m with an l-bit output of a CSPRB generator on a k-bit random input seed S *along with a public-key encryption of S.* Other implementations of this idea using different CSPRB generators and different types of encodings of S have been proposed by Blum, Blum and Shub in [BBS] and Goldwasser, Micali and Tong in [GMT], but they were not as efficient as our implementation and relied on different number theoretic assumtions.

We show a way to encrypt the seed S for which:

(1) We prove that sending the encrypted seed S along with the exclusive-or of the message and the output of the CSPRB generator on input S, does not compromise the apparent "randomness" of the output of the CSPRB generator. (Yao [Y] proved that sequences outputed by CSPRB generators are polynomial time indistinguishable from truly random sequences, only when the seed is totally unknown).

(2) The encoding and decoding of the seed S take at most $O(k^3)$ steps and k bits of cyphertext, where k is the size of the seed.

2. Number Theoretic Background

Let $|m|$ denote $\log_2 m$ and Z_N^* denote the set of positive integers less than N which are relatively prime with N. Let $(\frac{x}{N})$ denote the Jacobi symbol of $x \in Z_N^*$ mod N. Recall that for

$x \in Z_N^*$ where the prime factorization of N is $N = \prod_i p_i^{\alpha_i}$

$$(\frac{x}{N}) = \prod_i (\frac{p_i^{\alpha_i}}{N})$$

and,

$$(\frac{x}{p_i}) = 1 \text{ if } x \text{ is a quadratic residue mod } N \text{ and} -1 \text{ otherwise}$$

Let $N = pq$, where $p \equiv q \equiv 7 \bmod 8$. Set $QR_N = \{a \mid \exists x, x^2 \equiv a \ mod \ N\}$. Then, any $a \in QR_N$ has exactly two square roots with Jacobi symbol $+1$ of the form x and $-x$ in Z_N^* and exactly one square root with Jacobi symbol $+1$ which is less than $\frac{N}{2}$. Let x be such a square root of $a \in QR_N$.

Define $B(a,N)$ for $a \in QR_N$ as follows:(1)

$$B(a,N) = \textit{least significant bit of } x$$

Recent results by Chor and Goldreich [CG] imply that

Lemma 1[CG]: If there exists a probabilistic polynomial in $|N|$ time algorithm that computes $B(x,N)$ for a fraction $\frac{1}{2} + \frac{1}{poly(log\ N)}$ of $x \in QR_N$ where $N \in H$, then there exists a probabilistic polynomial in $|N|$ time algorithm for factoring $N \in H$.

They extend Lemma 1 as follows. Let

$$B^k(a,N) = k^{th} \text{ least significant bit of } x \text{ where } x^2 \equiv a \bmod N,\ x < \frac{N}{2}, \text{ and } (\frac{x}{N}) = +1.$$

Lemma 2 [CG], [VV]: Let $1 \le j \le \log\log N$. If there exists a probabilistic polynomial in $|N|$ time algorithm that on inputs $N \in H$ and $S \subseteq \{B^i(x,N) \mid 1 \le i \le \log\log N,\ i \ne j\}$, guesses $B^j(x,N)$ for a fraction greater than $\frac{1}{2} + \frac{1}{poly(log\ N)}$ of $x \in QR_N$, then there exists a probabilistic polynomial in $|N|$ time algorithm for factoring $N \in H$.

3. The Encryption Scheme

Let the public file of user A contain a composite number N product of two primes p and q, both congruent to 7 mod 8. A keeps p and q secret.

Let the security paramter $k = \log N$. Define $G(l,r,N)$ to be the l-bit boolean vector whose $l-i+1$st bit is $B^g(r^{2^{h+1}},N)$ where $h = \left\lfloor \frac{i}{\log k} \right\rfloor$ and $g = i - h \cdot \log k$ for $1 \le i \le l$.

(1) This predicate was first introduced in [GMT] and further analyzed in [BCS] and [VV].

Suppose any user B in the network wants to send an encoding of an l-bit message $m \in M$ to user A.

How to Encode m:

1. Choose $r \in Z_N^*$ at random.
2. Compute $G(l,r,N)$ using algorithm A below.
3. set $f = r^{2^{h+1}} \bmod N$ where $h = \left\lceil \frac{l}{\log k} \right\rceil$
4. Let the encryption of m be the pair ($G(l,r,N) \oplus m, f$).

Algorithm A

```
input: $l,r,N$ where $k = \log N$.
output: $G(l,r,n)$
    1) Let $h = \max\{j \mid j \cdot \log k \geq l\}$.
    2) For j=0 to h-1 do
        Let $temp = r^{2^{j+1}} \bmod N$
        For i= 1 to log $k$, let the $l-(j \cdot \log k + i)+1$ bit of $G(l,r,n)$ be $B^i(temp, N)$
        (namely, the $i^{th}$ least significant bit of $temp$).
```

Note that the size of the encoding of an l-bit message m is $l+k$. Computing $G(l,r,N)$ can be done in $O(\frac{lk^2}{\log k})$ steps.

How to Decode

Let the pair (D,f) be an public-key encoding of an l bit message sent to user A. Then, user A who knows the factors of N does:

1. Let $h = \frac{l}{\log k}$ 2. compute r such that $f^{2^{h+1}} = r \bmod N$ by performing the algorithm B below.
2. Compute $G(l,r,N)$ using algorithm A above.
3. Let $m = G(l,r,N) \oplus D$.

The cost of decoding is $O(k^3) + O(\frac{lk^2}{\log k})$. This is faster than a previous $O(k^4)$ of all previous probabilistic encoding schemes [GM, BBS, GMT]. We achieve this by exploiting the special properties of p and q. We show this in algorithm B below. The main idea is that instead of extracting square roots of f mod N at every step of the computation to retrieve $G(l,r,N)$, we first compute r in one exponentiation and then compute $G(l,r,N)$ by algorithm A.

Algorithm B:

Input: p, q, h and $r^{2^{h+1}} \bmod N$ where $r \in QR_N$ and $k = \log N$.
Output: $r \in QR_N$
Running time: $O(k^3)$ operations.

Let $p=8t+7$ and $q=8l+7$ for some t and l.

1) Let $u \equiv 2^{h+1}(t+1)^{h+1} \bmod p-1$

2) Let $v \equiv 2^{h+1}(l+1)^{h+1} \bmod q-1$

(The above exponents u and v can be precomputed on inputs p, q and $h+1$ alone)

3) Let $a \equiv (r^{2^{h+1}})^{u} \bmod p$

4) Let $b \equiv (r^{2^{h+1}})^{v} \bmod q$

5) Compute $c \in Z_N^*$ such that $c \equiv a \bmod p$ and $c \equiv b \bmod q$ using the Chinese Remainder theorem.

6) Output r = c.

Steps 1 and 2 above can be precomputed when the cryptosystem is set up. The cost of steps 3 -4 are two modular exponentiations and one gcd computation of k bit numbers. The complexity of the naive algorithm for performing these operations is $O(k^3)$.

Decoding Lemma: The decoding algorithm is correct.

Proof: To compute square roots of $a \in QR_p$, note that $(\frac{a}{p}) = a^{4t+3} = +1$ implies $(a^{2t+2})^2 = a \bmod p$. Thus to compute the square root of $a \bmod p$ which is a square itself, simply compute $a^{2t+2} \bmod p$. Finally, to compute the 2^{h+1}-th root of $a^{2^{h+1}} \bmod N$ which is a square, compute $a^{(t+1)^{h+1}2^{h+1}} \bmod p$ and $a^{(l+1)^{h+1}2^{h+1}} \bmod q$, and take the Chinese reminder of the results mod pq. The exponents $(t+1)^{h+1}2^{h+1} \bmod p-1$ and $(l+1)^{h+1}2^{h+1} \bmod q-1$ do not depend on r and can be precomputed, knowing p, q and the length of the message.

4. Security Analysis

4.1 The Model

Let M be any message space. We think of the sender as a stochastic process which produces $m \in M$ in accordance with a polynomial time computable system of probabilities. Thus every $m \in M$ has an a priori probability of being sent. The adversary knows and can compute the probability distribution of the message space. When he intercepts the cyphertext he can calculate from it a set of a posteriori probabilities of the various messages which may have produced this cyphertext. Shannon defines a cypher to be *perfect secure* if: after the adversary intercepts the cyphertext, the a posteriori probability of the cyphertext representing some message must be the same as the a priori probability of the same message before interaction.

We adopt Shannon's perfect secrecy criteria to polynomial time bounded adversaries. See also [GM] and [Y].

Let k be the security parameter of our system. We think of a public-key encryption scheme as a probabilistic, polynomial time algorithm Π that receives k as input from each user in the system, and outputs a pair of polynomial in k time computable encryption and decryption algorithms $E:M \rightarrow C$ and $D:C \rightarrow M$ where C is the set of cyphertexts. The encryption algorithm E may be probabilistic. We let $E(m)$ denote the set of possible encodings of $m \in M$. Note that when $E = RSA$, $E(m)$ is unique for any $m \in M$.

The measure of security we would like to enforce is the following.

Informally, no polynomial time passive adversary which can compute $x \in E(m)$ for any $m \in M$, should be able to come up with even one message m whose encodings he can even distinguish from the encodings of a random message $r \in M$. This implies that seeing an encryption of a message does not help the adversary to compute or guess with any significant advantage any partial information about the message itself.

Formally, Let A be a polynomial time, probabilistic algorithm that takes as input a description of E and outputs a message $m_{E,A} \in M$. Let B be a polynomial time, probabilistic algorithm that takes as input $x \in E(m)$ and outputs 1 or 0 (this is B's guess as to whether $m = m_{E,A}$ or not). Note that B may even know $m_{E,A}$. Let p_m denote the probability that B guesses 1 on $x \in E(m)$ (the probability here is taken over the encodings $x \in E(m)$ and B's coin tosses).

Definition: We say that Π is *polynomially secure* if for all M, for all A, B, for all polynomials Q, for all sufficiently large k, $| p_{m_{A,E}} - p_r | < \frac{1}{Q(k)}$ for a random $r \in M$ and random E generated by Π.

This notion of security was introduced by Goldwasser and Micali in [GM]. Another notion of security was later proposed by Yao[Y] using time-bounded information theory. Rackoff [R] showed the two notions equivalent, providing some evidence that polynomial security is the correct notion for security for a public-key cryptosystem.

4.2 Proof of Security

Factoring Assumption (FA): Let F_k^A denote be the fraction of k-bit integers factored by probabilistic, polynomial in k time algorithm A. Then, for all polynomials Q, for sufficiently large k, $F_k^A < \frac{1}{Q(k)}$.

Theorem: The FA implies that the scheme described in section 3 is polynomially secure

Idea of the proof: Let Π be public-key encryption scheme that produces encryption algorithms of the form we proposed in section 3. Namely, a user in the system gives input k to Π and receives in return two k-bit prime numbers p and q such that $p \equiv q \equiv 7 \bmod 8$. The user publicizes $N = pq$ and keeps p and q secret. To send the user messages one encodes as we suggested in section 3. For the duration of the proof, lets denote this encryption algorithm by E_N, and the set of possible encryptions of message m by $E_N(m)$.

Now, suppose for contradiction, that for some pair A,B of polynomial-time probabilistic algorithms and for some polynomial Q, for infinitely many k, on a random E_N output by Π on k

and a random $r \in M$, algorithm A on input E_N outputs an $m_{A,E} \in M$ such that

$$| p_{m_{A,E}} - p_r | > \frac{1}{Q(k)} \quad \text{(a)}$$

Pick a k for which (a) holds. Assume without loss of generality that for all $m \in M$, $|m| = l$ and $l < P(k)$ for polynomial P. Pick a random N (product of two k-bit primes p and q such that $p \equiv q \equiv 7 \bmod 8$) whose factors you don't know. Input N to algorithm A to receive $m_{E,A} \in M$. Now, let's define two types of experiments.

Type (1): Pick $x \in QR_N$ at random and an l-bit random R. Let $h = \max\{j \mid j \cdot \log k \geq l\}$. Feed algorithm B, $x^{2^{h+1}} \bmod N$ and $R \oplus G(l,x,N)$ (i.e a random member of $E_n(R)$). The probability that B answers 1 here is p_R.

Type (2): Pick $x \in QR_N$ at random. Feed algorithm B, $x^{2^{h+1}} \bmod N$ and $m_{A,E} \oplus G(l,x,N)$ (i.e a random member of $E_n(m_{A,E})$). The probability that B answers 1 here is $p_{m_{A,E}}$.

By assumption, $| p_{m_{A,E}} - p_R | > \frac{1}{Q(k)}$. Without loss of generality, let $p_{m_{A,E}} - p_R > \frac{1}{Q(k)}$. By the weak law of large numbers, in polynomial in $Q(k)$ number of type (1) and type (2) experiments we can estimate $p_{m_{A,E}}$ and p_R.

We now use a combination of the proof techniques of Goldwasser and Micali in [GM] and Yao in [Y]. Let the i-type experiment be defined as follows. Feed algorithm B, $r^{2^{h+1}} \bmod N$ and $G(l,r,N) + m_i$ where m_i consists of the concatenation of the first i bits of $m_{A,E}$ and $l-i$ random bits (i.e. feed B a random member of $E_N(m_i)$). Let p_i be the probability that during the i-type experiment B outputs 1. (this probability is taken over B's coin tosses, choices of r and the $l-i$ random bits). There must exist an $1 \leq i \leq l$ such that $p_{i+1} - p_i > 1/lQ(k)$ and it can be found by running a polynomial in $l \cdot Q(k)$ number of i-type experiments and $i+1$-type experiments for all $1 \leq i \leq l$. Say we found such an i. There are two cases to look at: $i \leq \log k$ and $i > \log k$. The first case yields a method for predicting $B^i(x,N)$ with probability greater than $\frac{1}{lQ(k)}$ on inputs N and $\{B^j(x,N) \mid 2 \leq j \leq 1\}$, which by lemma 2 implies a probabilistic factoring algorithm for $N \in H$. This contradicts the factoring assumption. In the second case $i > \log k$. The rest of the proof deals with this case. Pick an a in Z_N^* such that $(\frac{a}{N}) = -1$. Let G consist of the concatenation of $G(i,a^2,N)$ with $l-i$ random bits. Let $h' = \max\{j \mid j\log k \geq l\}$. Feed algorithm B as input, $a^{2^{h'+1}} \bmod N$ and $G \oplus m^0$ where m^0 consists of the concatenation of the first i bits of $m_{A,E}$,0, and $l-i-1$ random bits. Denote B's output by b_0. (Note that this is an i-type experiment). Now feed B with $a^{2^{h'+1}} \bmod N$ and $G + m^1$ where m^1 consists of the concatenation of first i bits of $m_{A,E}$,1, and $l-i-1$ random bits. Denote B's output by b_1. (Note that this is an $i+1$-type experiment). Let $j = \log k$ and let b_{i+1} denote the $i+1$'st bit of $m_{A,E}$. If $b_0 = b_1$ then choose at random $c \in \{0,1\}$ and predict that $B^j(a,N) = c$, otherwise let $c = 0 \cdot b_0 + 1 \cdot b_1$ and predict $B^j(a,N) = b_{i+1} \oplus c$. The probability of predicting $B^j(a,N)$ correctly according to this rule is greater than $\frac{1}{2} + \frac{1}{Q(k)l}$. This yields by Lemma 2 a probabilistic time algorithm for factoring $N \in H$, which again contradicts the factoring assumption.

5. Eficiency Analysis and Comparison

Let k be the security parameter (i.e. the size of the composite number in the public file). To send an k bit message m where $k = \log N$ using RSA requires $O(k^3)$ operations to encode and decode and k-bit long encryption. In our scheme encoding requires $O(\frac{k^3}{\log k})$ operations, decoding requires $O(k^3) + O(\frac{k^3}{\log k})$ operations and the encryption of a k-bit message is $2k$-bit long.

Note that the time bounds for RSA encoding and decoding have been calculated for RSA(x) $= x^s \bmod N$ where $1 < s < \varphi(N)$ is picked at random as proposed in the original RSA paper. It has been suggested to let $s = 3$, then encoding is of k^2 complexity, while decoding still remains of k^3 complexity. However, Blum in [B] and Hastad in [H] point out a problem arising with $s = 3$(or any $s < \log k$) : if the same message is sent encrypted to 3(s respectively) separate people in the network each owning his own N_i, then an adversary tapping the lines can decode the message.

6. Other models of adversaries

The security analysis performed in section 4 was done with respect to passive adversaries. However, the scheme descibed above is not secure against more powerful than passive adversaries such as adversaries which can perform chosen cypher text attacks(CCA). In this attack, the adversary may have temporary access to the decoding equipment, afterwhich he tries to decode. No public key encryption scheme has been **proved** secure against such an attack even under the assumption that certain number theoretic problems are intractable.

On the other hand, no effective way of inverting the RSA function using a CCA on the RSA public-key-cryptosystem is known. We can modify our scheme to achieve the same security against CCA, as the deterministic RSA cryptosystem. We do this by modifying our system to be based on the assumption that the RSA function is intractable. This implementation maintains the same cost of encoding and decoding, same data expansion, and the same security against partial information attacks as our factoring based scheme. In addition it is as secure against CCA as the deterministic RSA system. We briefly describe this implementation in section 7.

Note that what we are interested in is an encryption scheme which is *1-pass.* That is, to send a message to user A we need only look up his public file, compute the encryption of our message and send it, and there is no need of further interaction with A. Solving the problem of public-key encryption which is secure against chosen-cyphertext attacks using a *2-pass* system (where to send a secret message to A two levels of interaction are allowed) is much easier and can be achieved.

7. Implementation of the Scheme based on RSA intractability assumption

Let N denote the modulos in the RSA scheme. That is $N = pq$ where p and q are primes of the same size. Let the public file of user A contain such a composite number N whose factors p and q only A knows. Let s_i denote the inverse of 3^i mod $\varphi(N)$ for $1 < i < \varphi(N)$. Let $k = \log N$.

Define $G(l,r,N)$ of section 4 to be an l-bit vector whose $l-i+1$th bit is the $i - \log k \left\lfloor \frac{i}{\log k} \right\rfloor$ least significant bit of r^{3^j} mod N where $j = \left\lfloor \frac{i}{\log k} \right\rfloor$ Note that computing $G(l,r,N)$ can be done without knowing the factors of N in $O(\frac{lk^2}{\log k})$ time.

To encode m where $|m| = l$: pick $r \in Z_N^*$ at random, compute $G(l,r,N)$, let $h = \left\lceil \frac{l}{\log k} \right\rceil$ and $f = r^{3^{h+1}}$ mod N, let the encoding of m be the pair ($G(l,r,N) \oplus m, f$).

To decode (D,f) where $|D| = l$, recall that $s_{h+1} 3^{h+1}$ mod $\varphi(N) \equiv 1$. Compute $r = f^{s_{h+1}}$ mod N, compute $G(l,r,N)$, and let $m = G(l,r,N) \oplus D$.

8. Remarks and Open problems

This paper presented a probabilistic encryption scheme which is secure against all partial information attacks in presence of passive adversaries, provided factoring is hard, whose cost of encoding and decoding is fast, and has constant factor data expansion. An interesting open question remains:

Given x^2 mod N, are ½ of the bits of x such that $x < \frac{N}{2}$ and $(\frac{x}{N}) = +1$ as hard to compute as x? If so, then one may build an extremeley efficient encryption scheme which requires only 2 multiplications to encode, and is secure against all partial information attacks.

Acknowledgements

We thank Ron Rivest, Dana Angluin, Don Coppersmith, Oded Goldreich and Silvio Micali for their helpful comments.

References

[ABCGM]Awerbach, Blum, Chor, Goldwasser, Micali, *A Provably Fair Coin Toss in A Byzantine Network*, Submitted to PODC 1985.

[CG] Chor, Goldreich, *RSA/Rabin Bits are 1/2 +* $\frac{1}{poly(|N|)}$ secure, Proc. of Crypto 84, Santa Barbara.

[B] M. Blum, private communication .

[BBS]L. Blum, M. Blum and M. Shub, *A simple secure pseudo random number generator*, Advances in Cryptology: Proc. of CRYPTO-82, ed. D. Chaum, R.L. Rivest and A.T. Sherman. Plenum press 1983, pp 61-78.

[BCS]Ben-Or, Chor, Shamir, *On the Security of RSA Bits*, Proceedings of 15th ACM symposuim on Theory of Computation, April 1983, pp. 421-430

[BD] A. Broder, and D.Dolev, *On Flipping Coins in Many Pockets*, 25th IEEE FOCS, 1984.

[BM] M. Blum and S. Micali, *How to generate cryptographically strong sequences of pseudo-random bits*, Proc. 23rd IEEE Symp. on Foundations of Computer Science, 1982, pp 112-117

[DH] Diffie and Hellman, *New Directions in Cryptography*, IEEE Transactions on Infromation Theory.

[GM]S. Goldwasser and S. Micali, *Probabilistic Encryption*, JCSS 28(2), 1984. References

[GM2]Goldwasser and Micali, *Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Infromation*, 1982 14th STOC.

[GMT]S. Goldwasser, S. Micali and P. Tong, *Why and how to establish a private code on a public network*, Proc. 23rd IEEE Symp. on Foundations of Computer Science, 1982, pp 134-144

[GMR]S. Goldwasser, S. Micali and R.Rivest, *Probabilistic Signature SEcure Against Chosen Cyphertext Attack*, In Preparation.

[H] J. Hastad, *On Using RSA with Low Exponent in A Public Key Network*, In Preparation.

[L] D. Lipton, *How to Cheat in Mental Poker.*

[Ra] M. Rabin, *Digital Signatures as Intractable as Factorization.*

[RSA]R. Rivest, A. Shamir, and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, Commun. ACM, vol. 21, Feb. 1978, pp 120-126

[Y] A.C. Yao, *Theory and applications of trapdoor functions*, Proc. 23rd IEEE Symp. on Foundations of Computer Science, 1982, pp 80-91.

[Sh] C. Shannon, *A Mathematical Theory of Cryptography* , 1945.

[VV] V. Vazirani, U. Vazirani, *Trapdoor Pseudo-Random Number Generators, with Applications to Protocol Design* , 1983.

SECTION IV

ANALYSIS AND CRYPTANALYSIS