Fast Generation of Secure RSA-Moduli with Almost Maximal Diversity

Ueli M. Maurer

Institute for Signal and Information Processing
Swiss Federal Institute of Technology
CH-8092 Zürich, Switzerland

Abstract

This paper describes a new method for generating primes together with a proof of their primality that is extremely efficient (for 100-digit primes the average running time is equal to the average time required for finding a "strong pseudoprime" of the same size that passes the Miller-Rabin test for only four bases), that yields primes that are nearly uniformly distributed over the set of all primes in a given interval, and that is easily modified to yield (with no additional computational effort) primes that are nearly uniformly distributed over the subset of these primes that satisfy certain security constraints for use in the RSA cryptosystem. This method is used to generate, for a given encryption exponent e, an RSA-modulus $m = pq$ that is nearly uniformly distributed over all secure RSA-moduli in a given interval I, i.e., over the set of all integers in I that are (1) the product of exactly two primes p and q none of which is smaller than a given limit L, where (2) $(p-1, e) = (q-1, e) = 1$ and (3) $p-1$ and $q-1$ each contain a prime factor greater than a given limit L', and where (4) for all but a provably (given) small fraction of plaintexts in Z_m^*, the minimum number of iterated encryptions with exponent e required to recover the plaintext, is provably greater than a given limit M. Our method exploits a well-known result due to Pocklington [20] that allows one to prove the primality of p when only a partial factorization of $p-1$ is known. These prime factors of $p-1$ are generated by recursive application of the prime generating procedure. Although the discussion is centered on the RSA system, our method can of course be used in other cryptographic systems, such as the Diffie-Hellman public key distribution system, that require large primes satisfying certain security constraints.

1. Introduction

The primes p and q in the RSA cryptosystem [26] must satisfy certain conditions in order to prevent that the system can easily be broken. Two kinds of attacks are usually considered: factorization of the modulus $m = pq$ and decryption by iterated encryption [25,27,29,31]. To prevent fast factorization of the modulus, p and q should be sufficiently large, and $p-1$ (and similarly $q-1$) should contain a large prime factor p' (q') to make the so-called $p-1$ factoring method (see [24], p.172 and [21]) infeasible. Other conditions, for example that $p+1$, p^2+1 or $p^2 \pm p + 1$ (and similarly for q) each contain a large prime factor can for a similar reason also be placed on the primes. However these additional conditions are less stringent and are usually not considered, with some exceptions for the (p+1)-condition [12,18,28]. The usual way to guarantee that the iterated encryption attack is infeasible for virtually all plaintexts is by requiring that p' (and similarly q') must contain a large prime factor p'' (q''). To satisfy these conditions [25,27] it is usually required that p and q are roughly of the same size (e.g. have equally many digits) and to choose p (and similarly q) of the special form $p = 2ap' + 1$ with $p' = 2bp'' + 1$ where p' and p'' are both primes and where a and b are small integers (often $a = b = 1$). This is a strong restriction on the set of allowed primes as will be shown.

Our approach differs in the way the security constraints are satisfied. We argue that the security requirements should be specified by the security parameters described in Section 3, but that the RSA-modulus should then be randomly chosen from the set of all RSA-moduli in a given interval that satisfy these constraints, in order to achieve maximal diversity and not to give the enemy any a priori information about the special form of the primes. For characterization of the iterated encryption attack, a new result (Theorem 2) is presented in Section 2 showing that the condition that $p'-1$ has a large prime factor p'' is not necessary, although this is in practice virtually always the case. For one reasonable formulation of the security constraints, about 23% of all products of two sufficiently large primes are shown to be acceptable. This fraction is by orders of magnitude greater than that achieved by using the strong restrictions described above. We argue that there is little reason for such strong restrictions when the security requirements can also be met by a much looser restriction. Moreover, even if the general factorization problem were indeed difficult, it is at least conceivable that a specialized fast algorithm exists for factoring numbers that are the product of two primes of the described special form, although at present these instances of the factorization problem seem to be the most difficult ones. However, the security constraints in our method for generating primes can be specified flexibly, and by an appropriate choice the described strong restrictions can also be met if required by the security policy.

The method for generating RSA-moduli described in Section 3 is based on an analysis of the probability distribution of the size of the smaller prime factor of an integer randomly selected from the set of integers of a given size that are the product of exactly two primes, both of which are greater than a given limit. The underlying recursive method for generating primes is based on an analysis of the probability distribution of the size of the largest prime factor of an integer randomly selected from the set of integers n of a certain size, for which $2n+1$ is prime. Other conditions on n, namely that $2an+1$, and possibly also $2b(2an+1)+1$, $2c(2b(2an+1)+1)+1$ etc., are primes, are also considered. These derivations, which are based on evident heuristic arguments similar to those used in [15] for estimating the number of primes p below n for which $(p-1)/2$ is also prime, lead to number-theoretic conjectures about the asymptotic behaviour of the mentioned distributions, which are of independent interest.

There exist two approaches for generating a prime in a given interval: to choose odd integers at random and to apply a primality test until a prime is obtained, or to construct a prime (e.g. by the methods of [18], [19], [28] or [31]). The first approach is not suited for practical applications, because the most efficient presently known general primality tests, which are due to Morain [17], and to Adleman, Pomerance and Rumely [1] as well as Cohen and Lenstra [5,6], are not efficient enough to be suited for implementation on a small computer or on special purpose cryptographic hardware. Another related paper by Goldwasser and Killian [11] shows that almost all primes can be certified in time polynomial in their length, but this approach cannot be efficiently implemented either. This problem of bad efficiency is in practice usually remedied by using instead of a primality test a probabilistic compositeness test [23,30], which is efficient but does not yield provable primes. The construction approach on the other hand allows one to prove the primality of the generated integers, but has either had the disadvantage that the diversity of reachable primes is moderate [18,28,31], that the interval for the primes cannot be chosen flexibly enough [18], or was computationally not efficient enough ([19], method A).

2. Theoretical Results on the Decipherability by Multiple Encryption

In the following, we denote by $ord_n(x)$ the order of x in Z_n^*, the multiplicative group modulo n. A basic fact that will be used repeatedly is: $n|m \implies ord_n(x)|ord_m(x)$. The following lemma, which is the key to our method for constructing primes, is a special case of a well-known theorem due to Pocklington ([20], see also [24], p. 109), which is the basis of the contruction method presented in [28].

<u>Lemma 1:</u> *Let $n-1 = RF$ where $F = \prod_{j=1}^{r} q_j^{\beta_j}$ with $q_1, \ldots, q_r$ distinct primes. If there is an integer a, satisfying $a^{n-1} \equiv 1 \pmod n$ and $(a^{(n-1)/q_j} - 1, n) = 1$ for $j = 1, \ldots, r$, then each prime factor p of n has the form $p = Fm + 1$ for some integer $m \geq 1$. Moreover, if $F > \sqrt{n}$, then n is prime.*

<u>Proof:</u> Let p be a prime divisor of n and let α be its multiplicity. We prove that $q_j^{\beta_j}|(p-1)$ for $j = 1, \ldots, r$. We have

$$a^{n-1} \equiv 1 \ (mod\ n) \implies ord_n(a)|(n-1) \implies ord_{p^\alpha}(a)|(n-1), \tag{1}$$

$$ord_{p^\alpha}(a)|\varphi(p^\alpha) \quad \text{where} \quad \varphi(p^\alpha) = (p-1)p^{\alpha-1}, \tag{2}$$

$$p \nmid (n-1) \overset{\text{by(1)}}{\implies} p \nmid ord_{p^\alpha}(a) \overset{\text{by(2)}}{\implies} ord_{p^\alpha}(a)|(p-1), \text{ and} \tag{3}$$

$$(a^{(n-1)/q_j} - 1, n) = 1 \implies a^{(n-1)/q_j} \not\equiv 1 \ (mod\ p) \implies ord_{p^\alpha}(a) \nmid \frac{n-1}{q_j}. \tag{4}$$

Using (1) and (3) and the trivial fact that for a prime q, $q^\beta|y, z|y$ and $z \nmid (y/q)$ together imply $q^\beta|z$, finally yields

$$(4) \overset{\text{by(1)}}{\implies} q_j^{\beta_j}|ord_{p^\alpha}(a) \overset{\text{by(3)}}{\implies} q_j^{\beta_j}|(p-1) \ \text{ for } 1 \leq j \leq r \implies F|(p-1).$$

It is obvious that if every prime factor of a number is greater than its square root then the number itself must be prime. □

More sophisticated conditions that guarantee the primality of a number p if the factored part of $p-1$ is less than $\sqrt{p}$ are described in [4] and [7] but these conditions are computationally more costly to verify. Note that Lemma 1 is usually stated in a way that allows the base a to be different for each q_j [4,7,28]. We will only make use of the special case because the verification of the conditions is first computationally more efficient and secondly allows to prove, as will be demonstrated by Theorem 2, that iterated encryption is infeasible to recover the plaintext when a is chosen to be the encryption exponent.

A few basic facts about Euler's totient function $\varphi(.)$ are:

$$\varphi(ab) \geq \varphi(a)\varphi(b) \text{ with equality if and only if } (a,b)=1, \tag{5}$$

$$\sum_{d|n} \varphi(d) = n, \quad \text{and} \tag{6}$$

$$p \text{ prime and } d|(p-1) \implies \#\left\{x \in Z_p^* : ord_p(x) = d\right\} = \varphi(d). \tag{7}$$

<u>Lemma 2:</u> *Let p be a prime and d a divisor of $p-1$. Then*

$$\#\left\{x \in Z_p^* : d|ord_p(x)\right\} \geq \frac{\varphi(d)}{d}(p-1)$$

with equality if and only if $(d,(p-1)/d)=1$.

<u>Proof:</u>
$$\#\left\{x \in Z_p^* : d|ord_p(x)\right\} \overset{\text{by}(7)}{=} \sum_{d':d|d',d'|(p-1)} \varphi(d') = \sum_{k:k|\frac{p-1}{d}} \varphi(kd)$$
$$\overset{\text{by}(5)}{\geq} \sum_{k:k|\frac{p-1}{d}} \varphi(k)\varphi(d) = \varphi(d)\sum_{k:k|\frac{p-1}{d}} \varphi(k) \overset{\text{by}(6)}{=} \varphi(d)\frac{p-1}{d}.$$
The inequality holds with equality if and only if $(d,(p-1)/d)=1$. □

The following new result shows that virtually any a can be used in Lemma 1 if all q_j's are large.

<u>Lemma 3:</u> *Let $p = RF+1$ be a prime with $F = \prod_{j=1}^{r} q_j^{\beta_j}$, $F > R$ and $(R,F)=1$, where q_j, $1 \leq j \leq r$, are primes. Then the fraction of elements $a \in Z_p^*$ that are successful in proving the primality of p by Lemma 1 is $\varphi(F)/F \geq 1 - \sum_{j=1}^{r} 1/q_j$.*

<u>Proof:</u> $a^{p-1} \equiv 1 \ (mod\ p)$ is satisfied for every $a \in Z_p^*$. If $(R,F)=1$ then $F|ord_p(a)$ and $a^{(p-1)/q_j} \not\equiv 1 \ (mod\ p)$ for $1 \leq j \leq r$ are equivalent statements. Application of Lemma 2 with $d = F$ completes the proof. □

The following theorem will be required to prove Theorem 2, the final result:

<u>Theorem 1:</u> *Let $m = pq$ be an RSA-modulus where $p-1 = R_pF_p$ and $q-1 = R_qF_q$ and where the prime factorizations of F_p and F_q are $F_p = \prod_{i=1}^{r} p_i'^{\alpha_i}$ and $F_q = \prod_{i=1}^{s} q_i'^{\beta_i}$, respectively. Then the fraction of plaintexts $x \in Z_m^*$ for which $ord_m(x)$ is not a multiple of $lcm(F_p,F_q)$ is upper bounded by $\sum_{i=1}^{r} 1/p_i' + \sum_{i=1}^{s} 1/q_i'$.*

<u>Proof:</u> By Lemma 2,
$$n_p \overset{\text{def}}{=} \#\left\{x \in Z_p^* : F_p|ord_p(x)\right\} \geq (p-1)\frac{\varphi(F_p)}{F_p} = (p-1)\prod_{i=1}^{r}(1-\frac{1}{p_i'}), \quad \text{and}$$
$$n_q \overset{\text{def}}{=} \#\left\{x \in Z_q^* : F_q|ord_q(x)\right\} \geq (q-1)\frac{\varphi(F_q)}{F_q} = (q-1)\prod_{i=1}^{s}(1-\frac{1}{q_i'}).$$
$F_p|ord_p(x)$ and $F_q|ord_q(x)$ for $x \in Z_m^*$ together imply $lcm(F_p,F_q)|ord_m(x)$. Since $Z_m^* = Z_p^* \times Z_q^*$ we have
$$\#\left\{x \in Z_m^* : lcm(F_p,F_q)|ord_m(x)\right\} \geq n_pn_q \geq (p-1)(q-1)\prod_{i=1}^{r}(1-\frac{1}{p_i'})\prod_{i=1}^{s}(1-\frac{1}{q_i'})$$
$$\geq |Z_m^*|\left(1-\sum_{i=1}^{r}\frac{1}{p_i'}-\sum_{i=1}^{s}\frac{1}{q_i'}\right). \qquad □$$

Iterated t-fold encryption in an RSA cryptosystem reveals the plaintext x if and only if $x^{e^u} \equiv x \ (mod\ m)$ for some $u \leq t$, i.e., if and only if $e^u \equiv 1 \ (mod\ ord_m(x))$ for some $u \leq t$. Hence the minimal number of encryptions needed to recover the plaintext is $ord_{ord_m(x)}(e)$; it is required for security that this number be large for virtually all x.

Theorem 2: *Let $m = pq$ be an RSA-modulus where $p-1 = R_pF_p$ and $q-1 = R_qF_q$, and where the prime factorizations of F_p and F_q are $F_p = \prod_{i=1}^{r} p_i'^{\alpha_i}$ and $F_q = \prod_{i=1}^{s} q_i'^{\beta_i}$, respectively. Further let $p_i'-1 = R_{p_i'}F_{p_i'}$ for $1 \leq i \leq r$ and $q_i'-1 = R_{q_i'}F_{q_i'}$ for $1 \leq i \leq s$ where the prime factorizations of $F_{p_i'}$ and $F_{q_i'}$ are $F_{p_i'} = \prod_{j=1}^{r_i} p_{ij}''^{\alpha_{ij}}$ for $1 \leq i \leq r$ and $F_{q_i'} = \prod_{j=1}^{s_i} q_{ij}''^{\beta_{ij}}$ for $1 \leq i \leq s$. For every integer a relatively prime to $(p-1)(q-1)$ and satisfying*

$$\text{for } 1 \leq i \leq r: \quad a^{(p_i'-1)/p_{ij}''} \not\equiv 1 \ (mod\ p_i') \ \text{ for } 1 \leq j \leq r_i,$$
$$\text{and for } 1 \leq i \leq s: \quad a^{(q_i'-1)/q_{ij}''} \not\equiv 1 \ (mod\ q_i') \ \text{ for } 1 \leq j \leq s_i,$$

the fraction of plaintexts $x \in Z_m^*$ for which $ord_{ord_m(x)}(a)$ is not a multiple of $lcm(F_{p_1'}, \ldots, F_{p_r'}, F_{q_1'}, \ldots, F_{q_s'})$ is upper bounded by $\sum_{i=1}^{r} 1/p_i' + \sum_{i=1}^{s} 1/q_i'$.

Proof: Similar arguments as used in the proof of Lemma 1 allow one to show that $F_{p_i'} | ord_{p_i'}(a)$ and hence that $F_{p_i'} | ord_{p_i'^{\alpha_i}}(a)$ and also that $F_{p_i'} | ord_{F_p}(a)$ for $1 \leq i \leq r$. Thus $lcm(F_{p_1'}, \ldots, F_{p_r'}) | ord_{F_p}(a)$. Simililarly one obtains $lcm(F_{q_1'}, \ldots, F_{q_s'}) \mid ord_{F_q}(a)$ and hence that $lcm(F_{p_1'}, \ldots, F_{p_r'}, F_{q_1'}, \ldots, F_{q_s'})$ divides $ord_{lcm(F_p,F_q)}(a)$. From $lcm(F_p, F_q) \mid ord_m(x)$ it follows that $ord_{lcm(F_p,F_q)}(a) \mid ord_{ord_m(x)}(a)$ and hence application of Theorem 1 completes the proof. □

Theorem 2 illustrates that, in order to prevent decipherability by iterated encryption, the condition that $p'-1$, where p' is the largest prime factor of $p-1$, must again have a very large prime factor p'', is not necessary.

3. Recursive Algorithm for Generating Cryptographically Secure Primes and RSA-Moduli with Almost Maximal Diversity

Lemma 1 suggests a method for constructing large primes. One can form the product $F = \prod_{j=1}^{r} q_j^{\beta_j}$ of some known primes q_j, raised to some powers β_j, and repeatedly choose an integer $R < F$ at random until $n = 2RF + 1$ can be proved to be prime by an appropriate choice of the base a. Lemma 3 shows that if n is indeed prime and if all q_j's are large, then virtually every base a can be used for certifying the primality of n. On the other hand, if n is composite but does not contain a small prime factor that allows to detect this by trial division, then virtually every base a will satisfy $a^{n-1} \not\equiv 1 \ (mod\ n)$ and hence reveal the compositeness of n, unless n is of a very special form.

When this construction approach is used for generating RSA-primes p and q (see also [28]), the major problem is to avoid that the generated primes are of a special form, i.e., that the enemy has a priori information about the generated primes that could help him to factor $m = pq$. Therefore the prime factors of F must not be chosen from a small set of "base primes". Instead, in order to generate candidates n that are random odd integers, these prime factors should rather be selected according to their actual probability distribution. This can be achieved by selecting the sizes of the prime factors according to their probability distribution and then generating primes of the selected sizes. Note that by doing this, we are solving the original problem of generating a prime by reduction to itself, i.e., by recursion. It is somewhat surprising that this recursive construction turns out to be even faster than the generation of a strong pseudoprime that passes the Miller-Rabin test for an appropriate number of bases.

Let again $m = pq$ denote an RSA-modulus where $p < q$, $p'_1, \ldots p'_r$ are the r largest distinct prime factors of $p-1$ in decreasing order and $q'_1, \ldots, q'_s$ are the s largest distinct prime factors of $q-1$ in decreasing order. Our aim is to randomly generate secure RSA-moduli $m = pq$ for a given encryption exponent e, i.e., to randomly select with uniform distribution one of the integers that (1) lie in a given interval I centered at N, (2) are the product of exactly two primes p and q satisfying $(p-1, e) = (q-1, e) = 1$, and (3) satisfy certain security constraints. According to the results in Section 2 and recommendations in the literature, these conditions are (1) that p as well as q must be greater than a given limit $L = N^{\gamma}$ for some γ with $0 < \gamma < 1/2$ (this is to prevent factorization by a method specialized for finding small prime factors), (2a) that $p-1$ and $q-1$ must contain large prime factors p'_1 and q'_1, respectively, with $p'_1 \geq L'$ and $q'_1 \geq L'$ for a given limit $L' = N^{\gamma'}$ for some γ' with $0 < \gamma' < \gamma$ (this is to prevent factorization by the so-called $p-1$ factoring method [24]), (2b) that the remaining $r-1$ largest prime factors $p'_2 \ldots, p'_r$ of $p-1$ and $s-1$ largest prime factors $q'_2, \ldots, q'_s$ of $q-1$, needed for the application of Lemma 1 and in order to satisfy condition (3) below, are each greater than a given limit $L'' = N^{\gamma''}$ for some γ'' with $0 < \gamma'' < \gamma'$ (this is to make the bound $\sum_{i=1}^{r} 1/p'_i + \sum_{i=1}^{s} 1/q'_i$ given in Theorem 2 sufficiently small), and (3) that $lcm(F_{p'_1}, \ldots, F_{p'_r}, F_{q'_1}, \ldots, F_{q'_s}) \geq M$ (see Theorem 2) for a given limit $M = N^{\delta}$ with $0 < \delta < 2\gamma$, where $F_{p'_1}, \ldots, F_{p'_r}$ and $F_{q'_1}, \ldots, F_{q'_s}$ are the factored parts of $p'_1-1, \ldots, p'_r-1$ and $q'_1-1, \ldots, q'_s-1$, respectively. Condition (3) can be satisfied by requiring that for some ρ with $0 < \rho < 1$, $F_{p'_i} \geq p_i'^{\rho}$ for $1 \leq i \leq r$ and $F_{q'_i} \geq q_i'^{\rho}$ for $1 \leq i \leq s$ as well as $F_p = \prod_{i=1}^{r} p'_i \geq p^{\rho}$ and $F_q = \prod_{i=1}^{s} q'_i \geq q^{\rho}$. If $p'_1, \ldots, p'_r$ and $q'_1, \ldots, q'_s$ are distinct, and $F_{p'_1}, \ldots, F_{p'_r}$ and $F_{q'_1}, \ldots, F_{q'_s}$ are pairwise relatively prime (as is virtually automatically the case and can be easily tested during the generation process), these conditions guarantee that $lcm(F_{p'_1}, \ldots, F_{p'_r}, F_{q'_1}, \ldots, F_{q'_s}) \geq N^{\rho^2}$, hence the choice $\rho = \sqrt{\delta}$ guarantees that condition (3) is satisfied. Note that $\rho \geq 1/2$ is required for the primality proof by Lemma 1 for the primes $p'_1, \ldots, p'_r, q'_1, \ldots, q'_s$, p and q, and in general the choice $\rho = 1/2$ guarantees a sufficiently large lower bound of $N^{1/4}$ on $lcm(F_{p'_1}, \ldots, F_{p'_r}, F_{q'_1}, \ldots, F_{q'_s})$, the guaranteed number (for virtually all plaintexts) of iterated encryptions required to recover the plaintext. However, other values for δ can easily be accommodated as well if necessary.

For practical applications where the RSA-moduli must be in a given interval $[M_1, M_2]$ centered at N (e.g. $N = 10^{200}$), we recommend to choose $\gamma \approx 0.4, \gamma' \approx 0.3, \rho = 1/2$ and $\delta = 1/4$. The choices $\gamma = 0.4$ and $\gamma' = 0.3$ guarantee that $p'_1 > \sqrt{p-1}$ and $q'_1 > \sqrt{q-1}$, and hence that if $\rho = 1/2$ then only one prime factor p'_1 of $p-1$ and one prime factor q'_1 of $q-1$ must be generated.

In the following we describe our algorithm for generating secure RSA-moduli with uniform distribution over a given interval $[M_1, M_2]$ centered at N, as well as the underlying recursive algorithm for generating primes in a given interval that satisfy certain conditions. The algorithm has been implemented on a VAX 8650 [8] which demonstrates its practicality and the correctness of the running time analysis. Many important and interesting implementation details can only be mentioned in this paper and are not discussed in detail. The algorithm is very well suited for implementation on a small computer like a PC or on special-purpose cryptographic hardware.

Assume the public encryption exponent e, an interval $[M_1, M_2]$ centered at N, and the security parameters γ, γ', γ'' and δ have been (arbitrarily) specified. The primes p and q will be generated to be compatible with e, i.e., to satisfy the contitions $(p-1, e) = 1$ and $(q-1, e) = 1$, and to satisfy the security constraints. In particular they satisfy the conditions given in Theorem 2 for a replaced by e. Note that e must be used as the base when Lemma 1 is applied for the primality proof of the largest prime factors of $p-1$ and $q-1$, i.e., of p'_1 and q'_1, in order to allow application of Theorem 2. The first step is to choose the relative size $\sigma_p = \log_N p$ of the smaller prime factor p according to its probability distribution $P[\sigma_p \geq \beta] \approx [\log(1-\beta) - \log\beta]/[\log(1-\gamma) - \log\gamma] \approx (1-2\beta)/(1-2\gamma)$ (see [16] for a derivation). Note that for $\gamma > 0.3$, σ_p is almost uniformly distributed over the interval $[\gamma, 1/2]$. The second step is to generate p at random in a small interval centered at N^{σ_p} by the recursive procedure

RANDOMPRIME described below. The last step is the generation of q at random in the interval $[M_1/p, M_2/p]$ by another application of the recursive procedure.

The probabilistic recursive procedure RANDOMPRIME generates a prime p at random with (virtually) uniform distribution over the interval $[P_1, P_2] = [P/c, cP]$, where c is a given interval span constant with $c \approx 1.05 \ldots 2$. The interval is specified equivalently by P_1 and P_2 or by P and c. If the upper interval boundary P_2 is below a certain limit P_{lim} (e.g. $P_{lim} = 10^7$), then p is generated by repeatedly choosing integers from $[P_1, P_2]$ at random until one is shown to be prime by trial division by all primes below its square root, and the procedure is finished. If $P_2 > P_{lim}$ then the first step is to select the relative size $\sigma_{p'_1} = \log_{P/2} p'_1$ of the largest prime factor p'_1 of $(p-1)/2$ according to the conditional probability distribution of the size of the largest prime factor of an integer n on the order of $P/2$, given that $2n+1$ is prime. This distribution will be discussed in Section 4. In this first step, a lower bound L' on p'_1 prescribed by security constraints can easily be accommodated by using the appropriate conditional distribution, given that $p'_1 > L'$, which can easily be obtained from the unconditional distribution by a normalization step. The largest prime factor p'_1 is then generated by a <u>recursive</u> call to RANDOMPRIME, where the interval for p'_1 is $[P'/c', c'P']$ with $P' = (P/2)^{\sigma_{p'_1}}$ and where c' is a small interval span constant (e.g. $c' \approx 1.05...2$, preferably $c' = c$). If $p'_1 > \sqrt{(P_2-1)/2}$ (i.e., roughly if $\sigma_{p'_1} > 1/2$), which happens with probability close to 70%, then an integer $R \in [(P_1-1)/2p'_1, (P_2-1)/2p'_1]$ is repeatedly chosen at random until $p = 2Rp'_1 + 1$ can be proved to be prime by application of Lemma 1 for some base a. In this case the procedure is finished. If $p'_1 < \sqrt{(P_2-1)/2}$, then the relative size $\sigma_{p'_2} = \log_{P/2} p'_2$ of the second largest prime factor p'_2 of $(p-1)/2$ is selected according to the conditional probability distribution of the size of the second largest prime factor of an integer n on the order of $P/2$, given that the largest prime factor of n is p'_1 and given that $2n+1$ is prime. This conditional distribution is equal to the conditional probability distribution of the size of the largest prime factor of a number l on the order of $P/(2p'_1)$, given that it is smaller than p'_1 and given that $2p'_1 l + 1$ is prime. This distribution is dicussed in [16]. Then p'_2 is generated, also by a recursive call to RANDOMPRIME where the interval for p'_2 is $[P''/c'', c''P'']$ with $P'' = (P/2)^{\sigma_{p'_2}}$ and where c'' is again a small interval span constant (e.g. $c'' = c$). This process continues until the product $x = \prod_{i=1}^{r} p'_i$ of the generated r distinct largest prime factors of $(p-1)/2$ satisfies $(P_2-1)/2x < p'_r$. This condition, which is stronger than the condition $x > \sqrt{(P_2-1)/2}$ suggested by the fact that the factored part of a number n in Lemma 1 must be greater than its square root, is necessary to ensure that a randomly chosen integer $R \in [(P_1-1)/2x, (P_2-1)/2x]$ cannot have a prime factor greater than p'_r, a circumstance that would violate the condition that p'_r is the r-th largest prime factor, and thus would falsify the final distribution of the sizes of the prime factors of $(p-1)/2$. Note that in order to test $2Rx+1$ for primality one first rules out all primes below a certain limit as divisors. (The optimal trial division limit is discussed in Section 5.) Instead of dividing $2Rx+1$ by these small primes we suggest to divide the much smaller integer R by these small primes and checking whether R has a remainder modulo one of the small primes that would result in $2Rx+1 \equiv 0$ modulo this prime. This gives an improvement over trial division of $2Rx+1$ which contributes to the surprising efficiency of the method.

The size of the primes to be generated by RANDOMPRIME decreases with increasing level of recursion. The end of the recursion is reached as soon as the upper boundary P_2 of the interval for the prime is below P_{lim} (see above). The final result of RANDOMPRIME is hence a <u>tree</u> where the root is the generated prime $p \in [P_1, P_2]$, the intermediate and terminal nodes are the largest prime factors of the immediate predecessors in the tree reduced by 1, and where the terminal nodes are primes less than P_{lim}. This tree allows one to give a simple proof of the primality of all nodes, starting with the terminal nodes. Note that the obtained primality proof for p is an even more succinct certificate for p than the ones proposed by Pratt [22] and Plaisted [19]. The procedure RANDOMPRIME can also be modified to efficiently solve the problem of randomly generating an integer with uniform distribution over a given interval, such that its factorization, or partial factorization, is known. This is a problem that has been considered by Bach [2], whose results were applied by Blum and Micali [3] to show that, according to their definition, the set B of predicates related to the quadratic residuosity problem is accessible. Note that our method differs from Bach's method [2], which is of theoretical rather than of practical interest, in that it is computationally efficient but on the other hand sacrifices the rigorously provable uniform distribution.

As mentioned in Section 1, common security constraints placed on the primes p and q are that they be of about equal size (e.g. have equally many digits) and that $p = 2p' + 1$, $q = 2q' + 1$, $p' = 2p'' + 1$ and $q' = 2q'' + 1$, where p', q', p'' and q'' are primes. In the following we compare the restrictiveness of this choice with that of our method. One can show by convincing heuristic arguments [16] that the probability that a prime on the order of P be of this special form is $\Pr[p' \text{ is prime} | 2p' + 1 \text{ is prime}] \times \Pr[p'' \text{ is prime} | 2p'' + 1 \text{ and } 4p'' + 3 \text{ are primes}] \approx C_2/\log(P/2) \times 3C_3/2\log(P/4)$ where $C_2 = \prod_{q\,\text{prime}, q\geq 3} q(q-2)/(q-1)^2 \approx 0.66016$ and $C_3 = \prod_{q\,\text{prime}, q \geq 5} q(q-3)/(q-1)(q-2) \approx 0.722$. Hence for example, among all numbers of size roughly N that are the product of exactly two roughly equally large primes, only a fraction $[C_2/\log(\sqrt{N}/2) \times 3C_3/2\log(\sqrt{N}/4)]^2$ has both prime factors of the special form. For $N = 10^{200}$, this fraction is one out of 5.4 billion. We argue that there is little reason for such a strong restriction, because the security requirements can also be met by a much looser restriction.

It can be shown again by convincing heuristic arguments (see [16]) that the fraction of numbers on the order of N that are the product of exactly two primes, none of which is less than N^{α}, is for $\alpha \in [0.03, 0.5]$ well approximated by $[\log(1-\alpha) - \log\alpha]/\log N$. We conjecture that for $0 < \alpha < 1/2$, $\lim_{N\to\infty} \#\{x \leq N : x = pq \text{ with } p \text{ and } q \text{ primes}, p \geq N^{\alpha}, q \geq N^{\alpha}\} \sim N(\log(1-\alpha) - \log\alpha)/\log N$. The fraction of numbers on the order of N that are the product of exactly two primes, p and q, both of which are greater than $L = N^{\gamma}$ and such that both $p-1$ and $q-1$ have a prime factor greater than $L' = N^{\gamma'}$ is approximately given by $\int_{\gamma}^{1/2}[1 + \log(\gamma'/x)][1 + \log(\gamma'/(1-x))]/x(1-x) \cdot dx/\log N$, which for $\gamma = 0.4$ and $\gamma' = 0.3$ becomes $0.094/\log N$. These numbers make up approximately 23% of all products on the order of N of exactly two primes both greater than $N^{0.4}$. (Note that the condition $lcm(F_{p'_1}, \ldots, F_{p'_r}, F_{q'_1}, \ldots, F_{q'_s}) \geq N^{\delta}$ is no essential additional restriction on the set of allowed RSA-moduli pq. It is only a restriction on the method for finding such primes.) Thus we can claim, unlike for the case where primes of a very special form are used, that factoring our RSA-moduli is about equivalent to solving the general problem of factoring the product of two large primes. Note that breaking the RSA system is not more difficult than the problem of factoring the product of two primes, which is not necessarily more and possibly much less, difficult than the problem of factoring general integers of comparable size. By allowing the RSA-modulus to be the product of more than 2 primes, i.e., a general integer, one could possibly obtain a cryptosystems which is as difficult to break as the general problem of factoring integers is difficult to solve. However this author does not suggest such a generalization for practical applications. To prove the equivalence of breaking the system and factoring would probably be even more difficult than for the original RSA-system.

4. On the Size of the Prime Factors of Certain Numbers

The basic probability distribution required by the procedure RANDOMPRIME is (for all relevant orders of magnitude, N) the probability distribution of the relative size $\sigma_{p'} = \log_N p'$ of the largest prime factor p' of an integer n on the order of N, given that n satisfies a certain condition. For the generation of the largest prime factor of $(p-1)/2$ in the procedure RANDOMPRIME, this condition is simply that $2n+1$ is prime. For the generation of the second and further largest prime factors of $(p-1)/2$ (which is only necessary with probability 0.3, namely when the largest prime factor p' of $(p-1)/2$ is less than $\sqrt{p}$) this condition is that $2an+1$ is prime, where a is the product of the prime factors generated so far. In the i-th level of the recursion, that is at depth i in the prime generation tree, the condition on n is of the form that $2a_1+1$, $2a_2(2a_1+1)+1$, $2a_3(2a_2(2a_1+1)+1)+1, \ldots$, $2a_{i-1}(2a_{i-2}(\ldots(2a_1+1)\ldots)+1)+1$ must all be prime.

All the arguments used in the derivation of these probability distributions are heuristic rather than mathematically rigorous, but they are empirically and numerically evident enough to be convincing. The arguments are similar to those used by Koblitz [15] for estimating the number $S(n)$ of primes p less than n for which $(p-1)/2$ is prime as $S(n) \sim C_2 n/(\log n)^2$, where $C_2 = \prod_{q \text{ prime}, q \geq 3} q(q-2)/(q-1)^2 \approx 0.66016$. Similar heuristics can also be used to derive the conjecture stated by Hardy and Littlewood [13] that the number $P_2(n)$ of prime-pairs (primes p for which $p+2$ is prime) below n is asymptotically given as $P_2(n) \sim 2C_2 n/(\log n)^2$. Note that, as is also true for our analysis, the gap is wide between these precise and numerically evident conjectures and the best result that can be rigorously proved. It is namely not even proved yet that there exist infinitely many prime-pairs or infinitely many primes p for which $(p-1)/2$. The basic result of our analysis is that the distribution of the sizes of the prime factors of a number n on the order of N is virtually independent of the different conditions that can be placed on n, although for example the probabilities that n is prime, or that $n = kq$ with q prime for a specific k, can strongly depend on the condition.

In this paper we only briefly consider the probability distribution of the largest prime factor of a number n on the order of N, given that $2n+1$ is prime. For further information we refer to a forthcoming paper [16]. Let $p_N(k)$ denote the probability that a randomly selected integer n on the order of N is of the form $n = kp'$ where p' is the largest prime factor of n, let $\overline{p}_N(k)$ denote the corresponding conditional probability, given that $2n+1$ is prime, and let $\overline{F}_1^N(\alpha)$ and $F_1^N(\alpha)$ denote the probability, with and without the above condition, respectively, of the event that the largest prime factor p' of n is smaller or equal to N^α. Heuristic reasoning suggests for $k \leq \sqrt{N}$ that $p_N(k) \approx 1/k \cdot 1/\log(N/k)$, where $1/k$ is the probability that k divides n and $1/\log(N/k) = 1/(\log N - \log k)$ is the probability that a number on the order of N/k is prime. Similar heuristic reasoning as used in [15] (for the case $k=1$) suggests that $\overline{p}_N(k) \approx A_k p_N(k)$ where $A_k = C_2 \prod_{q \text{ prime}, q \geq 3, q|k} (q-1)/(q-2)$ and where C_2 has been defined above. We conjecture from similar heuristic evidence that for every fixed k the number $S_k(n)$ of primes $p \leq n$ for which $(p-1)/2 = kq$ with q prime satisfies $S_k(n) \sim A_k n/k(\log n)^2$.

$\overline{F}_1^N(\alpha)$ (and similarly $F_1^N(\alpha)$) can be computed for $\alpha \geq 1/2$ as $\overline{F}_1^N(\alpha) = \sum_{k=1}^{N^{1-\alpha}} \overline{p}_N(k)$. It is interesting to note that $E[A_k] = 1$ for randomly chosen integers k. Because $\sum_{k=1}^{N^{1-\alpha}} \overline{p}_N(k) \approx \sum_{k=1}^{N^{1-\alpha}} A_k p_N(k) \approx E[A_k] \sum_{k=1}^{N^{1-\alpha}} p_N(k) = E[A_k] F_1^N(\alpha) = F_1^N(\alpha)$, it follows that $\overline{F}_1^N(\alpha) \approx F_1^N(\alpha)$. That is, the distribution of the size of the largest prime factor of a random integer n does not depend on the condition that $2n+1$ is prime, although for example the probability that n is prime is 34% smaller when the condition "$2n+1$ prime" is given. Similar results can be obtained if additional conditions are placed on n, for example that "2a(2n+1)+1 must also be prime" for a given constant a, "$2b(2a(2n+1)+1)+1$ must also be prime for a given constant b, etc.. Knuth and Trabb Pardo [14] proved that the limiting function $F_1(\alpha) = \lim_{N\to\infty} F_1^N(\alpha)$ exists, where n is assumed to be uniformly distributed in $[1, N]$. $F_1(\alpha)$ is defined by $F_1(\alpha) = 1 - \int_\alpha^1 F_1(x/(1-x))/x \cdot dx$ with $F_1(\alpha) = 1$ for $\alpha \geq 1$. In particular, $F_1(\alpha) = 1 + \log \alpha$ for $1/2 \leq \alpha \leq 1$. The probability that the largest prime factor of a randomly chosen large integer is greater than its square root is hence $\log 2 \approx 0.69$. We conjecture on the basis of substancial heuristic and empirical evidence that $\lim_{N\to\infty} \overline{F}_1^N(\alpha) = F_1(\alpha)$.

$\overline{F}_1^N(\alpha)$ can be very well approximated for finite N by $F_1(\alpha) - \epsilon/\log N$ for $1/2 \leq \alpha < 1$ where ϵ is a computable positive constant less than one, and by $(1 - \epsilon/(\log 2 \log N))F_1(\alpha)$ for $0 \leq \alpha \leq 1/2$.

For small values of k below a certain limit, the cases $(p-1)/2 = kq$ with q prime must be considered separately and a different type of recursion of RANDOMPRIME than the one described above must be used. k must be selected according to its discrete distribution $\bar{p}_N(k)$. Then p_1' must be generated by a recursive call to RANDOMPRIME with interval $[(P_1-1)/2k, (P_2-1)/2k]$. Note that this type of recursion, which occurs seldomly, requires the repeated generation of p_1' until $2kp_1'+1$ is prime, as opposed to the reselection of a random integer R until $2Rp_1'+1$ is prime. Hence the computation time for finding the desired prime p is considerably greater if this second type of recursion has to be used. However measures can be taken to keep the increase in computation time reasonably small. We recommend for the sake of speed to make use only of the fast type of recursion. Note that this slightly changes the a priori probabilities of those primes p in the given interval for which k is very small, i.e., $p-1$ has a very large prime factor, which is of no relevance for practical applications.

5. Comments on the Average Running Time and Conclusions

We can only present some main results of the running time analysis because the number of pages of this paper is limited. For more details we refer to a forthcoming paper to be submitted to the Journal of Cryptology. The aim of the running time analysis of the procedure RANDOMPRIME is to determine the ratios $c_{psp}(n) = T_{psp}(n)/T_{exp}(n)$ and $c_{pri}(n) = T_{pri}(n)/T_{exp}(n)$, where $T_{psp}(n)$, $T_{pri}(n)$ and $T_{exp}(n)$ are the expected times required for finding an n-bit strong pseudoprime that passes the Miller-Rabin test for one base, for generating one n-bit prime by our algorithm and for one full modular n-bit exponentiation, respectively. Since modular exponentiation is the most time consuming operation, $c_{psp}(n)$ and $c_{pri}(n)$ are almost implementation independent.

In order to find a strong pseudo-prime that passes the Miller-Rabin test for one base one can repeatedly select integers at random, rule out all primes below a certain limit as possible divisors by trial division, and if the trial division test is passed apply the Miller-Rabin test which costs one modular exponentiation. The optimal upper limit for trial division, i.e., the limit that minimizes the expected time for revealing the composite nature of an n-bit integer, can be shown to be $L_{opt}(n) = T_{exp}(n)/T_{div}(n)$, where $T_{div}(n)$ is the time for one division by a small integer (less than $L_{opt}(n)$). When this optimal trial division limit is used, $c_{psp}(n)$ can be shown to be a function growing as $n/\log n$; and, as an example, for $n = 332$ (100 decimal digits) it equals 14.5. Under the plausible simplifying assumption that in the procedure RANDOMPRIME the selected relative size $\sigma_{p_1'}$ of the largest prime factor is always equal to its average, i.e., $\sigma_{p_1'} = 0.624$, one can show that $c_{pri}(n) = 1.18 \cdot c_{psp}(n)$. In other words, the average running time for generating a prime is only roughly 20% greater than the time required for finding a strong pseudoprime that passes the Miller-Rabin test for only one base. For 100-digit integers, the expected time for generating a prime is equivalent to only 17.5 exponentiations (i.e., $c_{pri}(332) = 17.5$) compared to 14.5 for a pseudoprime. If the pseudoprime is tested for four different bases then the two methods are equally fast, but if more than four bases have to be tested (for a practical implementation 20 to 50 is reasonable in order to achieve a sufficient level of confidence), our method is considerably faster. The asymptotic running time is $n^4/\log n$.

Thus, our new method for generating primes not only offers the advantages of yielding provable primes that are virtually uniformly distributed over the set of primes in a given interval satisfying the flexibly specified RSA-security constraints, but moreover it is also faster than previous methods used to generate only "probable primes". Of course it is not restricted to the RSA cryptosystem, but it can be used in other cryptocraphic systems that require large primes satisfying certain security constraints, such as the Diffie-Hellman public key distribution system [9], the El-Gamal cryptosystem and signature scheme [10], the Blum-Micali pseudorandom sequence generator [3], etc..

Acknowledgement

I would like to thank Jim Massey for his valuable assistance during the preparation of this paper.

References

[1] L.M. Adleman, C. Pomerance, and R.S. Rumely, *On distinguishing prime numbers from composite numbers*, Annals of Mathematics, Vol. 117, 1983, pp.173-206.

[2] Eric Bach, *How to generate random integers with known factorization*, Proc. 15th annual ACM Symp. on Theory of Computing, 1983, pp. 184-188.

[3] M. Blum and S. Micali, *How to generate cryptographically strong sequences of pseudo-random bits*, SIAM J. Comput., Vol. 13, No. 4, Nov. 1984, pp. 850-864.

[4] J. Brillhart, D.H. Lehmer and J.L. Selfridge, *New primality criteria and factorizations of* $2^m \pm 1$, Math. Comp., Vol. 29, 1975, pp. 620-647.

[5] H. Cohen and W. Lenstra, Jr., *Primality testing and Jacobi sums*, Math. Comp., Vol. 42, 1984, pp. 297-330.

[6] H. Cohen and A.K. Lenstra, *Implementation of a new primality test*, Mathematics of Computation, Vol. 48, No. 177, Jan. 1987, pp. 103-121.

[7] C. Couvreur and J.J. Quisquater, *An introduction to fast generation of large prime numbers*, Philips Journal of Research, Vol. 37, 1982, pp. 231-264, (errata: id, Vol. 38., 1983, p. 77).

[8] R. De Moliner, *Effiziente Konstruktion zufälliger grosser Primzahlen*, Diploma Project, Institute for Signal and Information Processing, Swiss Federal Institute of Technology, Zurich, 1989.

[9] W. Diffie and M.E. Hellman, *New directions in cryptography*, IEEE Trans. Info. Th., Vol. IT-22, Nov. 1976, pp. 644-654.

[10] T. El-Gamal, *A public key cryptosystem and a signature scheme based on the discrete logarithm*, IEEE Trans. Info. Th., Vol. IT-31, No. 4, July 1985, pp. 469-472.

[11] S. Goldwasser and J. Kilian, *Almost all primes can be quickly certified*, Proc. of the Annual ACM Symp. on the Theory of Computing, 1986, pp.316-329.

[12] J. Gordon, *Strong primes are easy to find*, Proc. EUROCRYPT'84, Lecture Notes in Computer Science, Vol. 209, pp. 216-223, Springer Verlag.

[13] G.H. Hardy and J.E. Littlewood, *Some problems of 'partitio numerorum'; III: on the expression of a number as a sum of primes*, Acta Math., Vol. 44, 1923, pp. 1-70.

[14] D.E. Knuth and L. Trabb Pardo, *Analysis of a simple factorization algorithm*, Theoretical Computer Science, Vol. 3, 1976, pp. 321-348.

[15] N. Koblitz, *Primality of the number of points on an elliptic curve over a finite field*, Pacific J. of Mathematics, Vol. 131, No. 1, 1988, pp. 157-165.

[16] U.M. Maurer, *Some number-theoretic conjectures and their relation to the generation of cryptographic primes*, submitted to the 2nd IMA Symposium on Cryptography and Coding, Dec. 1989, Cirencester, UK.

[17] F. Morain, *Primality testing: news from the front*, these proceedings.

[18] M. Ogiwara, *A method for generating cryptographically strong primes*, Research Reports on Information Sciences, No. C-93, Dept. of Information Sciences, Tokyo Institute of Technology, April 1989.

[19] D.A. Plaisted, *Fast verification, testing, and generation of large primes*, Theoretical Computer Science, Vol. 9, 1979, pp. 1-16, (errata: id., Vol 14., 1981, p. 345).

[20] H.C. Pocklington, *The determination of the prime or composite nature of large numbers by Fermat's theorem*, Proc. Cambridge Philos. Soc., Vol. 18, 1914-1916, pp. 29-30.

[21] J.M. Pollard, *Theorems on factorization and primality testing*, Proc. Cambridge Philos. Soc., Vol. 76, 1974, pp. 521-528.

[22] V.R. Pratt, *Every prime has a succinct certificate*, SIAM J. Computing, Vol. 4, No. 3, Sept. 1975, pp.214-220.

[23] M.O. Rabin, *Probabilistic algorithm for testing primality*, Journal on Number Theory, Vol. 12, 1980, pp. 128-138.

[24] Hans Riesel, *Prime numbers and computer methods for factorization*, Boston, Basel, Stuttgart: Birkhäuser, 1985.

[25] R.L. Rivest, *Remarks on a proposed cryptanalytic attack on the M.I.T. public key cryptosystem*, Cryptologia, Vol. 2, No. 1, Jan. 1978, pp. 62-65.

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

[27] C.P. Schnorr, *Zur Analyse des RSA-Schemas*, Preprint: Fachbereich Mathematik, Universität Frankfurt, Dec. 1981.

[28] J. Shawe-Taylor, *Generating strong primes*, Electronics Letters, Vol. 22, No. 16, July 1986, pp. 875-877.

[29] G. Simmons and M. Norris, *Preliminary comments on the M.I.T public key cryptosystem*, Cryptologia, Vol. 1, No. 4, Oct. 1977, pp. 406-414.

[30] R. Solovay and V. Strassen, *A fast Monte-Carlo test for primality*, SIAM J. Computing, Vol. 6, No.1, March 1977, pp. 84-85.

[31] H.C. Williams and B. Schmid, *Some remarks concerning the M.I.T. public-key cryptosystem*, BIT, Vol. 19, 1979, pp. 525-538.