**Chương trình KC-01:**
**Nghiên cứu khoa học phát triển công nghệ thông tin và truyền thông**

**Đề tài KC-01-01:**
**Nghiên cứu một số vấn đề bảo mật và an toàn thông tin cho các mạng dùng giao thức liên mạng máy tính IP**

**Báo cáo kết quả nghiên cứu**

# ĐẢM BẢO TOÁN HỌC CHO CÁC HỆ MẬT

## Quyển 3A: "Sinh tham số an toàn cho hệ mật RSA"

HÀ NỘI-2003

**Báo cáo kết quả nghiên cứu**

# ĐẢM BẢO TOÁN HỌC CHO CÁC HỆ MẬT

## Quyển 3A: "Sinh tham số an toàn cho hệ mật RSA"

**Chủ trì nhóm nghiên cứu:**
**TS. Lều Đức Tân**

# MỤC LỤC

**PHỤ LỤC- CHƯƠNG TRÌNH NGUỒN**

# CHƯƠNG I

# HỆ TIÊU CHUẨN CHO HỆ MẬT RSA

## 1. MỞ ĐẦU

Hệ mật RSA là một trong những hệ mật có độ an toàn dựa trên quan điểm tính toán do đó một hệ tiêu chuẩn cần thiết để áp đặt cho hệ mật này chính là nhằm cho nó có được tính an toàn cần thiết. Một hiện thực là với các hệ mật có độ an toàn tính toán thì giá trị của nó chỉ được giới hạn trong thời gian mà thông tin do nó bảo mật (thời gian đối phương tìm ra được nội dung thật của thông tin sau khi đã có bản mã). Thời gian trên tùy theo yêu cầu của vấn đề cần bảo mật mà đặt ra cụ thể tuy nhiên chung ta có thể đưa ra một số năm Y khá lớn nào đó (như vài chục năm chẳng hạn). Do thời gian tính toán phụ thuộc vào hai yếu tố quan trọng đó là thuật toán sử dụng và năng lực (cụ thể ở đây là tốc độ tính toán và dung lượng lưu trữ của hệ thống máy tính phục vụ) tính toán.

### 1.1. Thông số an toàn cho một hệ mật có độ an toàn tính toán

Do kiến thức về thuật toán tấn công là chỉ có được tại thời điểm hiện tại, trong khi đó năng lực tính toán luôn được tăng trưởng theo luật Moore (sau 18 tháng thì tốc độ xử lý của máy tính tăng gấp đôi) cho nên khi xem xét thời gian an toàn của hệ mật chúng ta có thể quy chiếu đến tổng số các thao tác cần thiết mà máy phải thực hiện, ký hiệu là $T_0$ và gọi là thông số an toàn của hệ mật. Nếu ký hiệu t là tổng số các thao tác mà hệ thống tính toán được trong 1.5 năm với khả năng tính tại thời điểm hiện hành thì theo luật Moore tổng số thao tác mà nó thực hiện được trong 1.5 kế tiếp là 2t... cho nên sau một thời gian k lần của 1.5 năm hệ thống tính toán của đối phương có thể hoàn thành được tổng số thao tác là T được tính ước lượng như sau.

$$T<2t+t2^2+...+t2^k=t(2^{k+1}-1) \qquad (1\text{-}1)$$

Theo công thức trên ta hoàn toàn có thể dùng giá trị $T_0=t2^k$ để làm thông số an toàn cho hệ mật có thời gian bảo mật là 1.5k năm.

Giá trị t được tính theo công thức

$$t=1.5*365*24*3600*R\approx 2^{26}R \qquad (1\text{-}2)$$

ở trên R là tốc độ xử lý của máy tính tại thời điểm hiện hành. Tại thời điểm hiện hành (năm 2003) thì hệ máy tính có tốc độ xử lý tiên tiến nhất là 2.8Ghz, như vậy với loại máy tính này có tốc độ tính toán vào khoảng 700Mip$\approx 2^{30}$ phép toán trong 1 giây vậy ta thu được $t\approx 2^{56}$.

Để xác định được giá trị $T_0$ tại thời điểm năm y với thời gian an toàn là Y năm ta có thể tính toán chúng theo công thức sau.

$$T_0=2^{56+\frac{Y+y-2003}{1.5}} \qquad (1\text{-}3)$$

Trong những phân tích sau này, chúng ta chỉ cần quan tâm đến số mũ của $T_0$ và ký hiệu là $E_0$, khi này công thức tính $E_0$ là.

$$E_0=56+\frac{Y+y-2003}{1.5} \qquad (1\text{-}4).$$

Một khi đã xác định giá trị E theo yêu cầu bảo mật của hệ một mật có độ an toàn tính toán nói chung và cho hệ RSA nói riêng thì nếu tồn tại một kiểu tấn công đối với nó thì bắt buộc thời gian tấn công đó phải không dưới $O(2^{E_0})$.

Ví dụ. Để có được một sự an toàn trong thời gian Y=15, 30, 45, 60,… năm tính từ 2003 thì $E_0$ tương ứng là:66, 76, 86, 96,….

Trong nhiều tài liệu, khi đánh giá về độ an toàn cuả một hệ mật các tác giả còn đưa ra đơn vị đo khác nhau chẳng hạn như chi phí (theo đơn vị tiền hay thơi gian) phải trả khi muốn phá được hệ mật đó... với phân tích mà chúng ta đã đưa ra ở trên thì thông số thời gian an toàn được xem xét trên đơn vị một máy PC. Hiển nhiên trong một số điều kiện nào đó (chủ yếu là khả năng thuật toán có thể song song được) thì bằng cách thực hiện đồng thời trên nhiều máy thì tổng thời gian thực hiện thuật toán sẽ được giảm đi. Với cách tính trong công thức (1-4) thì với thời gian an toàn trong Y năm khi thuật toán chỉ thực hiện

trên 1 PC vậy để rút ngắn thời gian chỉ trong 1 năm thì số PC cần đến sẽ là $2^{\frac{Y}{1.5}}$. Với Y=45 năm (tương ứng với độ phức tạp $O(2^{86})$ thì nếu liên kết song song được $2^{30}$ máy PC bài toán sẽ giải được trong 1 năm.

Từ nay về sau, trong mọi phân tích chúng tôi sẽ dựa vào số mũ an toàn

$$E_0=86. \qquad (1\text{-}5)$$

## 1.2.Vấn đề xây dựng hệ tiêu chuẩn cho hệ mật RSA

Muốn đưa được hệ mật RSA vào sử dụng thì một trong những công việc phải làm đầu tiên đó là xây dựng những yêu cầu về nó nhằm mục đích loại bỏ những nguy cơ mất an toàn một khi vi phạm các yêu cầu đó- Hệ thống các yêu cầu nói trên được gọi là hệ tiêu chuẩn. Trên thế giới thường xuyên có những công bố về những tấn công đối với các hệ mật nói chung và RSA nói riêng và tương ứng với nhưng công bố đó là các cập nhật về hệ tiêu chuẩn cho RSA. Một cơ sở nổi tiếng nhất và có lẽ là chuyên nghiệp nhất trong lĩnh vực trên là "RSA Laboratories" và đối với họ chuẩn X9.31 công bố năm 1997 cho đến nay vẫn được sử dụng.

### *1.2.1. Chuẩn X9.31*

Chuẩn X9.31 do RSA Laboratories quy định cho việc sinh tham số cho hệ mật RSA, nó bao gồm các tiêu chuẩn sau.

*S1. Các modulo N=pq được sử dụng có số bit là 1024+256x với x=0, 1, 2, ... và như một hệ quả p, q là các số 512+128x bit.*

*S2. Các giá trị p-1, p+1, q-1, q+1 đều có ước nguyên tố lớn không dưới 100 bit.*

*S3. gcd(p-1,q-1) nhỏ.*

*S4. p-q có ước nguyên tố lớn trên 64 bit.*

### *1.2.2. Phương pháp xây dựng chuẩn của chúng ta*

Để có một chuẩn của riêng mình đối với hệ RSA chúng ta tốt nhất nên xuất phát từ chuẩn X9.31, tìm hiểu nguyên do để dưa ra các yêu cầu trong chuẩn đó, bổ xung thêm các thông tin mới hơn liên quan đến RSA vào chuẩn. Bằng cách tiếp cận này, cùng với thông tin về số mũ an toàn $E_0$ được đưa ra trong mục 1.1 chúng tôi đã đưa ra được một hệ tiêu chuẩn phong phú hơn về mặt định tính và rõ ràng hơn về mặt định lượng so với X9.31.

## 2. MỘT SỐ TIÊU CHUẨN DỰ KIẾN CHO HỆ RSA

### 2.1. Tiêu chuẩn về độ lớn của N

Phương pháp sàng trường số cho đến nay được coi là một phương pháp phân tích số nguyên tiên tiến nhất. Thời gian tính tiệm cận của phương pháp sàng trường số để phân tích được hợp số N được cho bởi đánh giá sau.

$$T_1=\exp\{(1.92+O(1))(\ln N)^{\frac{1}{3}}(\ln\ln N)^{\frac{2}{3}}\} \qquad (1\text{-}6)$$

Như vậy để phân tích được số nguyên N có độ lớn là n bit ($n=\log_2 N+1$) ta cần phải thực hiện một số thao tác như đã đưa ra trong công thức trên. Để cho hệ RSA chống được kiểu tấn công phân tích theo phương pháp sàng trường số thì chúng ta cần chỉ ra được số n tối thiểu để $T_1 \geq T_0$.

Biến đổi $T_1$ theo luỹ thừa của 2 ta được

$$T_1=2^{E(n)} \text{ với}$$

$$E(n) = (1.92+O(1))\, n^{\frac{1}{3}}(\ln 2)^{-\frac{2}{3}}(\ln n+\ln\ln 2)^{\frac{2}{3}}$$

$$\approx 2.46\, n^{\frac{1}{3}}(\ln 2)^{-\frac{2}{3}}(\ln n+\ln\ln 2)^{\frac{2}{3}}$$

$$\approx 4.91\, n^{\frac{1}{3}}(\ln n+\ln\ln 2)^{\frac{2}{3}} \qquad (1\text{-}7).$$

Từ công thức (1-7) chúng ta tính được các giá trị E tương ứng đối với một số modulo RSA có số bit n=512+x*256 (x=0,1,…14) cho bởi bảng 1 dưới đây.

***Bảng 1***.

| n | 512 | 768 | 1024 | 1280 | 1536 | 1792 | 2048 |
|---|---|---|---|---|---|---|---|
| E(n) | 64 | 77 | 87 | 96 | 103 | 110 | 117 |
| 2304 | 2560 | 2816 | 3072 | 3328 | 3584 | 3840 | 4096 |
| 123 | 129 | 134 | 139 | 144 | 148 | 152 | 157 |

Qua các tham số tính được ở bảng 1 thì số modulo N với 1024 bit là phù hợp với yêu cầu có số mũ tấn công E=87 là không dưới $E_0$=86 vậy ta có dự kiến sau

***Dự kiến 1. Số modulo N dùng cho hệ mật RSA phải không dưới 1024 bit.***

## 2.2. Tiêu chuẩn về độ lớn các ước nguyên tố p và q của N

Trong các thuật toán phân tích số nguyên có một lớp thuật toán mà thời gian tính của chúng phụ thuộc vào độ lớn các ước trong số nguyên cần phân tích. Tiêu biểu trong số này là thuật toán phân tích dựa vào đường cong elliptic (ký hiệu là ECM) được mô tả như sau.

Input: N là hợp số

Output: p là ước của N.

1.repeat

1.1. Lấy ngẫu nhiên đường cong E(a,b): $Y^2Z=X^3+aXZ^2+bZ^3$.

1.2. Lấy ngẫu nhiên điểm P=(x,y,z)$\in$E, p$\leftarrow$1,i$\leftarrow$1.

1.3. While (i$\leq$I) and not(N>p>1) do

1.3.1. i$\leftarrow$i+1.

1.3.2. (x,y,z)$\leftarrow$i(x,y,z).

1.3.3. p←gcd(z,N).

2. Until (N>p>1).

3. Output←p.

ở trên I=max{rlog$_r$N: r là các số nguyên tố ≤B}.

Ta biết rằng nếu (x,y,z) tính được tại bước 1.3.2 là điểm O (điểm có toạ độ z=0) của đường cong E trên trường $F_p$ (hoặc $F_q$) thì tại bước 1.3.3 ta sẽ thu được ước không tầm thường của N. Lại biết rằng, nếu i! là bội của số điểm của đường cong trên các trường tương ứng trên thì (x,y,z) tính được tại bước 1.3.2 chính là điểm O cho nên theo định nghĩa của I thì nếu số điểm của đường cong chỉ có các ước nguyên tố không quá B thì cùng lắm là I bước trong vòng "While" nêu trên thuật toán sẽ thành công.

Bằng cách tối ưu hoá giá trị B người ta đã chứng tỏ được rằng phương pháp ECM có thời gian tính tiệm cận là.

$$T_2= O(\exp\{\sqrt{2\ln p \ln\ln p}\}) \qquad (1\text{-}8)$$

Do không có trong tay tài liệu nào phân tích tường minh về số liệu trên nên để bạn đọc yên tâm chúng tôi cố gắng lý giải và thu được một kết quả khiêm tốn hơn như sau.

***Kết quả 1.1.*** *Thời gian tính tiệm cận của ECM là*

$$T_2= O(\exp\{1.5\sqrt{\ln p \ln\ln p}\}) \qquad (1\text{-}9)$$

Chứng minh.

Trước hết chúng ta thấy rằng tham số I= max{rlog$_r$N: r là các số nguyên tố ≤B} được đưa ra trong thuật toán có thể thay bằng tham số M=B! (với chú ý rằng chúng là các vô cùng lớn cùng bậc) và thay vì cho việc lần lượt tính P←iP như đã nêu trong 1.3.2 với i=2,3,…,B ta chỉ cần tính một lần giá trị P←M (ở

đây P=(x,y,z)). Bằng phương pháp xích cộng thì việc tính điểm tích MP cần đến O(lnM) phép cộng hoặc nhân đôi điểm.

Do M=B! mà $B^{0.5B}<B!<B^B$ nên 0.5BlnB<lnM<BlnB hay lnM=cBlnB với c là một hằng số 0.5<c<1. Tóm lại ta có thời gian tính điểm MP là.

$$O(BlnB). \tag{1-10}$$

Trong [N.M.Stephens] (trang 413) cho biết rằng xác suất để số x là B-trơn là

$$\rho \approx u^{-u} \tag{1-11}$$

với $u=\frac{\ln x}{\ln B}$.

Và trong [Blanke-Seroussi-Smart] (bổ đề IX.1 trang 161) cho biết số điểm của đường cong là phân bố đều trên đoạn $[p+1-\sqrt{p}\,;p+1+\sqrt{p}\,]$ cho nên để thuật toán thành công ta cần phải duyệt vào cỡ $O(u^u)$ đường cong hay thời gian thực hiện thuật toán ECM là

$$T_2=O(BlnB.u^u) =O(\exp\{lnB+lnlnB+ulnu\}) \tag{1-12}$$

với $u=\frac{\ln p}{\ln B}$.

Lấy $lnB=\sqrt{\ln p \ln\ln p}$, thì số mũ vế phải của (1-12) là

$$lnB+lnlnB+ulnu=\sqrt{\ln p \ln\ln p}+lnlnB+\frac{\ln p}{\sqrt{\ln p \ln\ln p}}(lnlnp-lnlnB)$$

$$=2\sqrt{\ln p \ln\ln p}-\frac{1}{2}(\ln\ln p-\ln\ln\ln p)\left(\sqrt{\frac{\ln p}{\ln\ln p}}-1\right)$$

$$=1.5\sqrt{\ln p \ln\ln p}-\frac{1}{2}\left(\ln\ln p+\ln\ln\ln p\sqrt{\frac{\ln p}{\ln\ln p}}-\ln\ln\ln p\right)$$

Do $\frac{1}{2}\left(\ln\ln p+\ln\ln\ln p\sqrt{\frac{\ln p}{\ln\ln p}}-\ln\ln\ln p\right)$ là vô cùng lớn bậc thấp hơn so với $\sqrt{\ln p \ln\ln p}$ khi $p\to\infty$ nên

Từ (1-12) ta được $T_2$=O(exp{$1.5\sqrt{\ln p \ln\ln p}$ }) và đây là công thức cần chứng minh.□

Theo công thức trên thì thuật toán sẽ rất có hiệu quả khi N có một ước nhỏ và để chống lại tấn công ECM thì theo công thức (1-8) nếu m là số bit của p ta có độ phức tạp của phép phân tích là

$$T_1 = O(\exp\{\sqrt{2\ln p \ln\ln p}\})$$

$$=O(2^{\log_2 e\sqrt{2\ln p \ln\ln p}})$$

$$=O(2^{\log_2 e\sqrt{2m\ln 2 \ln(m\ln 2)}})$$

$$=O(2^{\sqrt{2m\log_2 e \ln(m\ln 2)}})$$

vậy theo yêu cầu về $E_0$=86 chúng ta thấy rằng nếu

$$\sqrt{2m\log_2 e\ln(m\ln 2)} \geq E_0 \qquad (1\text{-}13)$$

Tuy nhiên nếu q và p xấp xỉ nhau thì phương pháp ECM được đưa về trường hợp khó nhất, vì vậy các tài liệu đề cập đến tiêu chuẩn này luôn lấy q và p xấp xỉ nhau. Tại đây chúng tôi cũng đề nghị một tiêu chuẩn như vậy.

***Dự kiến 2. Các số nguyên tố p và q đều xấp xỉ $\sqrt{N}$ (512 bit).***

## 2.3.Tiêu chuẩn về ước nguyên tố của p±1

### *2.3.1.Mở đầu*

Tiêu chuẩn p±1 và q±1 phải có ước nguyên tố lớn được đưa ra nhằm chống lại tấn công phân tích số theo thuật toán p-1 của Pollard và p±1 của Williams. Tất cả các hệ tiêu chuẩn cho hệ RSA đã công bố đều có tiêu chuẩn này tuy nhiên các định lượng về tính "lớn" của các ước thường chưa có một lý giải cụ thể. Trong mục này chúng tôi sẽ trình bày lại phương pháp p±1 của Williams với mục đích làm sáng tỏ điều trên.

### *2.3.2. Cơ sở của thuật toán*

Chú ý rằng thuật toán gốc của Williams là dựa vào các kết quả về các dãy Lucas, còn thuật toán mà chúng tôi sẽ trình bày dưới dây được dựa vào một kết quả đơn giản nhưng tương đương liên quan đến khái niệm bậc mở rộng.

Cho trường $F_p$ với p là số nguyên tố lẻ, D là một phần tử bất kỳ thuộc $F_p$. Ký hiệu hình thức $\sqrt{D}$ là một phần tử nào đó (có thể không thuộc $F_p$) thoả mãn điều kiện $(\sqrt{D})^2$=D.

Xét tập $F[\sqrt{D}]=\{(a,b): a,b\in F_p\}$ với hai phép toán "+" và "." định nghĩa như sau:

$$\begin{cases} (a,b)+(u,v)=(a+u,b+v) \\ (a,b).(u,v)=(au+Dbv,av+bu) \end{cases} \qquad (1\text{-}14)$$

Ta có $F_p[\sqrt{D}]$ là trường mở rộng của $F_p$, hơn nữa nếu D là thăng dư bậc 2 ($\sqrt{D}\in F_p$) thì $F_p[\sqrt{D}]=F_p$ và ngược lại $F_p[\sqrt{D}]$ là trường (với $p^2$ phần tử) mở rộng thực sự của $F_p$.

Với mọi phần tử $0\neq(a,b)\in F_p[\sqrt{D}]$ luôn tồn tại số d sao cho $(a,b)^d\in F_p$, ta gọi giá trị d>0 nhỏ nhất thoả mãn điều kiện trên là bậc mở rộng của (a,b) và kí hiệu là $ord_D(a,b)$. Chúng ta dễ dàng kiểm tra được rằng bậc mở rộng các tính chất cơ bản như bậc thông thường như

-Nếu $(a,b)^d\in F_p$, thì $ord_D(a,b) \mid d$.

-Nếu $ord_D(a,b)=d$, $ord_D(u,v)=e$ với gcd(d,e)=1 thì $ord_D((a,b)(u,v))=de$.

Ngoài ra ta còn có kết quả sau.

***Kết quả 1.2.*** *Với mọi $0\neq(a,b)\in F_p[\sqrt{D}]$ ta có.*

*(i)-Nếu D là thăng dư bậc 2 trên $F_p$ thì $ord_D(a,b) \mid (p-1)$.*

*(ii)-Ngược lại $ord_D(a,b) \mid (p+1)$.*

Chứng minh.

Kết quả (i) là hiển nhiên. Ngược lại do $F_p[\sqrt{D}]$ là trường $p^2$ phần tử nên hiển nhiên ta có bậc thông thường của mọi phần tử khác 0 của trường này đều là ước của $K=p^2-1$ tức là $(a,b)^K=1$. Xét $(u,v)=(a,b)^{p+1}$ ta có $(u,v)^{p-1}=(a,b)^K=1$ vậy (u,v) là nghiệm của phương trình $x^p-x=0$. Biết rằng trong trường $F_p[\sqrt{D}]$ thì mọi nghiệm của phương trình trên đều là phần tử của trường con $F_p$ vậy ta đã có $(a,b)^{p+1}\in F_p$ và kết đã được chứng minh.□

### *2.3.3. Thuật toán Williams*

Input : $N=pq$ với $p\neq q$ và $p-1=\prod_{r_i\le B} r_i^{c_i}$ hoặc $p+1=\prod_{r_i\le B} r_i^{c_i}$ .

Output: p.

1. Do

    1.1. Lấy ngẫu nhiên $D\in Z_N$, $(a,b)\in Z_N[\sqrt{D}]$ $(D,b\neq 0)$.

    1.2. $p\leftarrow gcd(b,N)$, if $(p=1)$ $p\leftarrow gcd(D,N)$; $i\leftarrow 1$.

    1.3. While $(i\le I)$ and not$(N>p>1)$ do

        1.3.1. $i\leftarrow i+1$;

        1.3.2. $(a,b)\leftarrow (a,b)^i$

        1.3.3. $p\leftarrow gcd(b,N)$

8. Until $N>p>1$.

9. Return p.

ở trên $I=\max\{r\log_r N: r \text{ nguyên tố } \le B\}$.

Do các tính toán theo modulo p là phép toán trên trường $Z_p=F_p$ có đặc số p, hơn nữa bộ công thức (1-14) thực chất là cộng và nhân các số dạng $a+b\sqrt{D}$ một cáhc thông thường nên ta có

$$(a,b)^p=(a+b\sqrt{D})^p=a^p+b^p\sqrt{D}^p=a+b\sqrt{D}\,D^{\frac{p-1}{2}} \qquad (1\text{-}15)$$

Nếu D là thặng dư bậc 2 modulo p ta có $D^{\frac{p-1}{2}}$ =1 và ngược lại ta có $D^{\frac{p-1}{2}}$ =-1 như vậy ta có kết quả sau

$$(a,b)^{p+\left(\frac{D}{p}\right)} = (1,0) \qquad (1\text{-}16)$$

với $\left(\frac{D}{p}\right)$ là kí hiệu Legendre.

Kết hợp các điều kiện p-1=$\prod_{r_i \le B} r_i^{c_i}$ , D là thặng dư bậc 2 modulo p với (a,b) được tính theo bước 1.3.2 của thuật toán thì tại giá trị

$$i=\max\{c_i \log_{r_i} p : c_i>0\} \le I$$

ta có i! sẽ là bội của p-1 cho nên theo kết quả trên thì b=0 (mod p) do đó gcd(b,N)>1. thêm vào nữa nếu b≠0 (mod q) ta có ngay p=gcd(b,N).

Hoàn toàn tương tự với p+1=$\prod_{r_i \le B} r_i^{c_i}$ , D là không thặng dư bậc 2 modulo p thuật toán cũng thành công trong việc tìm p.

Rõ ràng thời gian tính của thuật toán sẽ là O(B) với B là ước nguyên tố nhỏ nhất trong các ước nguyên tố lớn nhất của p-1 và của p+1. Với cách tấn công trên, để đảm bảo tính an toàn cho hệ mật RSA chúng ta có thể đưa ra yêu cầu là p±1 cần phải có ước nguyên tố không dưới 86 bit. Tuy nhiên tiếp sau đây chúng ta phân tích thêm một chút về điều kiện này.

Trước hết theo nghịch lý ngày sinh chúng ta biết rằng để tìm được phần tử cùng số dư theo modulo B thì chỉ cần đến O($\sqrt{B}$) phép tính theo như phương pháp Rho mà Pollard đã chỉ ra cho nên nếu sau khi thực hiện phép tấn công như đã nêu trên, với kết quả thu được tại bước 1.3.2 là $(a_0,b_0)=(a,b)^{I!}$ tất nhiên chỉ khi gcd($b_0$,N)=1 chúng ta sẽ tiếp tục thực hiện như sau.

1.S←{$b_0$}, i←0, p←1.

2. While not(N>p>1) do

2.1. i←i+1;

2.2. Lấy ngẫu nhiên m.

2.3. $(a_i,b_i)\leftarrow(a_0,b_0)^m$

2.4. $S\leftarrow S\cup\{b_i\}$

2.5. $p\leftarrow\max\{\gcd(b_j-b_k,N): \forall b_j,b_k\in S\ 0\le j<k\le i\}$.

3. Return p.

Rõ ràng với phần bổ xung trên thì các ước p với p±1 có dạng sau

$p\pm1=R\prod_{r_i\le B} r_i^{c_i}$ với $r_i$ là các số nguyên tố $\le B_0$ còn R là ước nguyên tố thoả mãn $B_0<<R\le B$ thì phép tấn công trên sẽ tìm được p. Tuy không phải là luôn luôn có hiệu quả trong mọi trường hợp nhưng rõ ràng với dạng nêu trên của p±1 thì thời gian tấn công chỉ còn là $O(\sqrt{B})$. Để đảm bảo cho hệ RSA trước tấn công đã nêu chúng ta đưa ra tiêu chuẩn sau.

***Dự kiến 3. p±1 phải có ước nguyên tố lớn không dưới 172 bit.***

## 2.4. Tiêu chuẩn về ước nguyên tố của p-q

### *2.4.1. Mở đầu*

Trong [Silverman] có đưa ra một tiêu chuẩn là p-q có ước nguyên tố lớn. Tiêu chuẩn được đưa ra trên cơ sở chống lại các tấn công của thuật toán phân tích của Fermat và của Lehman. Các thuật toán này dựa vào ý tưởng chung là cố tìm x, y sao cho $x^2-y^2=N$ với x được tìm trong lân cận của giá trị $\lfloor\sqrt{N}\rfloor$. Trong mục này chúng tôi cố gắng lý giải tiêu chuẩn trên và chuyển thành điều kiện gcd(p-1;q-1) có ước nguyên tố lớn. Chú ý rằng gcd(p-1;q-1) là ước của p-q nên điều kiện của chúng tôi đưa ra là chặt hơn nhưng bù lại ta sẽ có một yên tâm được khẳng định trong bởi định lý 1.3 mà chúng tôi chỉ ra.

### *2.4.2. Tấn công kiểu giải hệ phương trình*

Hiển nhiên rằng nếu tìm được giá trị của p-q hoặc p+q là A chẳng hạn thì cùng

với điều kiện pq=N chúng ta dễ dàng tìm được p và q bằng cách giải một trong hai hệ phương trình tương ứng sau.

$$\begin{cases} pq = N \\ p - q = A \end{cases} \text{ khi biết p-q=A}$$

Rõ ràng kiểu phân tích trên cũng có hiệu lực trong trường hợp tồn tại các số nguyên có trị tuyệt đối nhỏ là a, b và c sao cho

$$ap\text{-}bq=c \qquad (1\text{-}17)$$

Khi này hệ phương trình để tìm p, q sẽ là

$$\begin{cases} (ap)(bq) = abN \\ ap - bq = c \end{cases} \qquad (1\text{-}18)$$

Bằng cách duyệt dần các giá trị a, b, c trong một miền [-B;B] với B nhỏ nào đó chúng ta sẽ có được một hệ có nghiệm.

Vì vậy để chống lại được tấn công kiểu trên thì yêu cầu cần thiết là đẳng thức (1-17) chỉ xảy ra với ít nhất một trong ba tham số a, b, c có trị tuyệt đối lớn, chẳng hạn không dưới $B=2^{E_0}$ với $E_0$ đã đưa ra trước đây. Cũng trong tài liệu trên tác giả Robert D. Silverman đưa ra điều kiện là

"*p-q có ước nguyên tố lớn*" (1-19)

và đồng thời cũng nhận định rằng đây là điều kiện rất khó thực hiện và đề nghị rằng chỉ cần thử thực hiện phân tích p-q bởi phương pháp ECM để đảm bảo rằng giá trị này không chỉ gồm những ước nguyên tố nhỏ?!.

Cố gắng tiếp theo của chúng tôi ở đây là đưa ra được một kết quả khẳng định nếu điều kiện (1-19) đủ chống lại tấn công kiểu giải hệ (1-18).

***Định lý 1.2.***

*Nếu p và q thoả mãn các điều kiện sau*

*(i). p-1 có ước nguyên tố là $p_0>B$ và $p_0$ không là ước của q-1.*

*(ii). p-1 và q-1 có ước nguyên tố là r>4B.*

*Thì không tồn tại a, b, c đồng thời có trị tuyệt đối không quá B để có (1-17).*

Chứng minh.

Từ (ii) ta giả sử p=xr+1 và q=yr+1 cho nên với (1-17) ta có

ap-bq=(ax-by)r+(a-b)=c suy ra

$$c=(a-b)\ (mod\ r) \qquad (1\text{-}20)$$

Không giảm tổng quát ta giả sử c≥0 nên (1-20) dẫn đến $c=\begin{bmatrix} a-b \\ r-(a-b) \end{bmatrix}$.

Rõ ràng từ r>4B còn (a-b)≤2B nên trường hợp c=r-(a-b) bị loại bỏ và (1-17) trở thành ap-bq=a-b hay

$$a(p-1)=b(q-1)$$

Từ điều kiện (i) thì b phải là bội của $p_0$>B và định lý đã được chứng minh.□

Với dự kiến 3 đưa ra trước đây thì điều kiên (i) trong định lý 1.3 đã được thoả mãn cho nên để chống lại tấn công vừa đưa ra ta chỉ cần bổ xung thêm điều kiện (ii) đó là.

***Dự kiến* 4. *gcd(p-1, q-1) phải có ước nguyên tố lớn không dưới 86 bit.***

## 2.5. Tiêu chuẩn về gcd(p±1,q±1)

### *2.5.1. Mở đầu*

Trong [Silverman] có đưa ra tiêu chuẩn gcd(p-1,q-1) phải có giá trị nhỏ với lập luận dựa trên phân tích xác suất gặp phải số mũ công khai có bậc thấp là cao nếu giá trị gcd(p-1,q-1) lớn. Trong phân tích của tác giả có dẫn đến những kết quả có trong tài liệu [RivestSilverman] nhưng rất tiếc là chúng tôi chưa có tài liệu này trong tay nên bù lại chúng tôI sẽ trình bày theo phân tích của riêng mình theo một tiếp cận khác. Bằng những phân tích mà chúng tôi sẽ đưa ra sau

này, không những cần quan tâm đến gcd(p-1,q-1) mà ta còn phải quan tâm đến các giá trị gcd(p+1,q-1), gcd(p-1,q+1) và gcd(p+1,q+1).

### *2.5.2. Phân tích số nguyên dựa vào gcd(p±1,q±1)*

Xét biểu diễn

$$N=\alpha F^3+AF^2+BF\pm 1 \text{ với } 0\le A,B<F. \quad (1\text{-}21)$$

Trong luận văn phó tiến sĩ của mình, tác giả Lều Đức Tân đã chỉ ra rằng nếu các ước nguyên tố của N có dạng xF±1 thì với không quá 2α bước là tìm được các ước của N (xem [Lều Tân], định lý 1.2 trang 23-24, định lý 4.3 trang 43-44).

Ta lại thấy rằng từ N=pq nên rõ ràng gcd(p-1,q-1) và gcd(p+1,q+1) đều là ước của N-1 và tương ứng gcd(p-1,q+1) và gcd(p+1,q-1) đều là ước của N+1. Như vậy nếu một trong bốn ước chung lớn nhất trên là lớn, giả sử đó là F thì giá trị F này có thể tìm trong các ước tương ứng của N-1 hoặc N+1.

Theo biểu diễn (1-21) thì nếu n là số bit của N và m là số bit của F thì

$$\alpha=\left\lfloor \frac{N}{F^3} \right\rfloor=O(2^{n-3m}).$$

Với yêu cầu về số mũ luỹ thừa 2 của độ phức tạp phép tấn công phải không dưới $E_0$=86, với n=1024 ta dễ dàng thu được m không quá $\frac{1024-86}{3}$=312.

Phân tích thêm về sự có mặt của tiêu chuẩn 2 là hai ước nguyên tố p và q của modulo N cùng số bit chúng ta thu được kết quả sau.

***Định lý 1.4.*** *Cho N=pq với p, q cùng số bit và F là giá trị xác định như sau.*

$$F=max\{gcd(p-1,q-1),gcd(p-1,q+1),gcd(p+1,q-1),gcd(p+1,q+1)\}$$

*Khi đó thời gian phân tích N là* $T= O(2^{\frac{n}{2}-2m})$ *với n là số bit của N và m là số bit của F.*

Chứng minh.

Ta dễ dàng nhận ra rằng, nếu F=gcd(p-1,q-1) hoặc F=gcd(p+1,q+1) thì N-1 là bội của F, giả sử

$$N=AF^2+BF+1 \text{ với } 0\le B<F \tag{1-22}$$

Trong khi đó p=xF+1 và q=yF+1 hoặc p=xF-1 và q=yF-1, khi này ta có.

$$N=pq=xyF^2\pm(x+y)F+1 \tag{1-23}$$

Đặt

$$\delta=\pm(x+y)\ (div\ F) \tag{1-24}$$

thì từ (1-22) và (1-23) ta thu được hệ phương trình sau

$$\begin{cases} B = x + y - \delta \\ A = xy + \delta F \end{cases} \tag{1-25}$$

Rõ ràng rằng nếu xác định được δ thì từ (1-25) ta luôn tìm được x và y và từ đó tính được p và q nên bài toán phân tích N được đưa về bài toán xác định δ.

Bây giờ từ p, q có cùng số bit giả sử p<q ta có p<q<2p suy ra x≤y≤2x hay 2x≤x+y≤3x mà $x\approx\frac{\sqrt{N}}{F}$ và $|\delta|\approx\frac{x+y}{F}$ ta có $2\frac{\sqrt{N}}{F^2}\le|\delta|\le3\frac{\sqrt{N}}{F^2}$ hay $|\delta|$ chỉ nhận trong số $M=\frac{\sqrt{N}}{F^2}$ giá trị khác nhau. Bằng cách vét cạn ta có thể tìm được số đúng của δ vậy thời gian thực hiện của thuật toán sẽ là $O(\frac{\sqrt{N}}{F^2})=O(2^{\frac{n}{2}-2m})$.

Trường hợp F=gcd(p-1,q+1) hoặc F=gcd(p+1,q-1) cũng được xét tương tự với F là ước của N+1. Vậy ta đã chứng minh xong định lý.□

Để chống lại được tấn công trên thì ta cần có $\frac{n}{2}$-2m≥$E_0$ hay

$$m\le\frac{n-2E_0}{4} \tag{1-27}$$

Với n=1024, $E_0$=86 ta có m≥213, vậy chúng ta đưa ra tiêu chuẩn sau đây về các giá trị gcd(p±1,q±1) đó là

***Dự kiến 5. max{gcd(p±1,q±1)} phải nhỏ hơn 213 bit.***

## 2.6. Tiêu chuẩn về các ước nguyên tố của λ(p±1)

Để đưa ra một điều kiện về các ước nguyên tố của λ(p±1) chúng ta cần xem xét đến một tấn công được gọi là tấn công số mũ lặp lại được mô tả như sau.

Input : N=pq với p≠q và $\lambda(p-1)=\prod_{r_i \le B} r_i^{c_i}$ hoặc $\lambda(p+1)=\prod_{r_i \le B} r_i^{c_i}$ sao cho $\prod_{c_i \ne 0} r_i \le K$.

Output: p.

1. Do

    2. Lấy ngẫu nhiên D,a,b∈$Z_N$ (D,b≠0).

    3. p←gcd(b,N), if (p=1) p←gcd(D,N); k←1.

    4. While not(N>p>1) and (k<K) do

        5.Lấy ngẫu nhiên e∈$Z_N$.

        6.t←1, (x,y)=(a,b)

        7. While (t≤$\log_2$N ) and not(N>p>1) do

            8. t←t+1;

            9. (x,y)←$(x,y)^e$

            10. p←max{gcd(x-a,N), gcd(b,N)}

        11.k←k+1.

8. Until N>p>1.

9. Return p.

Bây giờ chúng ta phân tích những trường hợp thành công của phép tấn công trên.

***Trường hợp thứ nhất.***

Nếu e là bội của $\prod_{c_i \neq 0} r_i$. Khi này rõ ràng luôn tồn tại $t \leq \log_2 N$ sao cho $e^t$ là bội của $\lambda(p\pm1)=\prod_{r_i \leq B} r_i^{c_i}$ hay ta đã có $e^t=0 \pmod{\lambda(p\pm1)}$ và khi này ta có luôn gcd(b,N) là bội của p.

***Trường hợp thứ hai.***

Nếu e là phần tử có ord(e) modulo $\lambda(p\pm1)$ thấp. Khi này sẽ tồn tại t sao cho $e^t=1 \pmod{\lambda(p\pm1)}$ và như vậy gcd(x-a,N) là bội của p.

Để đảm bảo an toàn cho hệ mật RSA trước tấn công trên chúng ta cần đưa ra một yêu cầu làm sao cho xác suất xảy ra hai trường hợp trên là rất nhỏ. Một lần nữa viện đến nghịch lý ngày sinh trước các phân tích kiểu xác suất cái gọi là rất nhỏ ở đây mà chúng ta phải đưa ra là không quá $2^{-160}$.

Một liên quan giữa yêu cần đưa ra ở trên và các ước nguyên tố của $\lambda(p\pm1)$ được cho bởi bổ đề sau.

***Bổ đề 1.5.*** *Xét vành $Z_N$. Giả sử r là ước nguyên tố của $\lambda(N)$, khi đó ta có:*

*(i). Xác suất để phần tử e có bậc là bội của r là* $1-\frac{1}{r}$

*(ii). Xác suất để phần tử e với gcd(e,N)=1 là bội của r là* $\frac{1}{r}$.□

Từ bổ đề trên ta thấy rằng xác suất để e có tính chất nêu trong điều kiện thứ nhất là không quá $\frac{1}{r}$ với r là ước nguyên tố của $\lambda(p\pm1)$ cho nên nếu giá trị này có ước nguyên tố r không dưới 172 bit thì hiển nhiên yêu cầu thứ nhất của chúng ta đã được thoả mãn. Đồng thời khi này điều kiện thứ hai nêu trên tối chỉ xảy ra khi bậc của e không là bội của r do đó xác suất để e thoả mãn điều

kiện này cũng không quá $\frac{1}{r}$. Tóm lại, nếu λ(p±1) có ước nguyên tố r không dưới 172 bit thì hệ mật RSA của chúng ta sẽ chống được tấn công số mũ lặp lại nêu trên.

***Dự kiến 6. λ(p±1) phải có ước nguyên tố lớn không dưới 172 bit.***

## 3. HỆ TIÊU CHUẨN CHO HỆ MẬT RSA

Tại phần 2, lần theo các tấn công đã có đối với hệ mật RSA chúng ta đã ghi nhận được 6 dự kiến đối với các tham số nguyên tố được phép sử dụng cho hệ mật này đó là.

*Dự kiến 1*. Số modulo N dùng cho hệ mật RSA phải không dưới 1024 bit

*Dự kiến 2*. Các số nguyên tố p và q đều xấp xỉ $\sqrt{N}$ (512 bit).

*Dự kiến 3*. p±1 phải có ước nguyên tố lớn không dưới 172 bit.

*Dự kiến 4*. gcd(p-1;q-1) phải có ước nguyên tố lớn không dưới 86 bit

*Dự kiến 5*. max{gcd(p±1,q±1)} phải dưới 213 bit.

*Dự kiến 6*. λ(p±1) phải có ước nguyên tố lớn không dưới 172 bit.

Trong việc xác định về lượng cho các dự kiến trên chúng ta đã dựa vào thông số số mũ an toàn $E_0$=86 được tính cho thời gian an toàn là 45 năm xét tại thời điểm hiện tại là năm 2003. Trong hệ tiêu chuẩn cho hệ mật RSA mà chúng tôi đưa ra dưới đây được xác định theo tham số số mũ an toàn E tổng quát.

Trong 6 dự kiến được đưa ra trên, hiển nhiên dự kiến thứ 6 là kéo theo dự kiến thứ 3 nên hệ tiêu chuẩn của chúng ta sẽ không cần đến tiêu chuẩn theo dự kiến 3.

Tóm lại đến đây chúng ta đưa ra được hệ tiêu chuẩn cụ thể là (xem trang sau).

## HỆ TIÊU CHUẨN CHO HỆ MẬT RSA DÙNG CHO THỜI ĐIỂM NĂM y VỚI THỜI GIAN AN TOÀN Y NĂM.

Số mũ an toàn E được tính theo công thức sau

$$E=56+\frac{Y+y-2003}{1.5}$$

***Tiêu chuẩn 1***. Số modulo N dùng cho hệ mật RSA phải có độ lớn cỡ n bit với n thoả mãn bất đẳng thức

$$4.91\, n^{\frac{1}{3}}(\ln n+\ln\ln 2)^{\frac{2}{3}}\geq E.$$

***Tiêu chuẩn 2***. Các số nguyên tố p và q đều xấp xỉ $\sqrt{N}$ .

***Tiêu chuẩn 3***. gcd(p-1;q-1) phải có ước nguyên tố lớn không dưới E bit

***Tiêu chuẩn 4***. max{gcd(p±1,q±1)} không quá $\frac{n-2E}{4}$ bit.

***Tiêu chuẩn 5***. λ(p±1) phải có ước nguyên tố lớn không dưới 2E bit.

# CHƯƠNG II

# XÂY DỰNG PHẦN MỀM SINH SỐ NGUYÊN TỐ DÙNG CHO HỆ MẬT RSA

## MỞ ĐẦU

Theo hệ tiêu chuẩn đưa ra cho hệ mật RSA tại chương trước bao gồm.

***Tiêu chuẩn 1***. Số modulo N dùng cho hệ mật RSA phải có độ lớn cỡ n bit với n thoả mãn bất đẳng thức

$$2.46\, n^{\frac{1}{3}} (\ln 2)^{-\frac{2}{3}} (\ln n + \ln\ln 2)^{\frac{2}{3}} \geq \text{E}.$$

***Tiêu chuẩn 2***. Các số nguyên tố p và q đều xấp xỉ $\sqrt{N}$ .

***Tiêu chuẩn 3***. gcd(p-1;q-1) phải có ước nguyên tố lớn không dưới E bit

***Tiêu chuẩn 4***. max{gcd(p±1,q±1)} không quá $\frac{n-2E}{4}$ bit.

***Tiêu chuẩn 5***. λ(p±1) phải có ước nguyên tố lớn không dưới 2E bit.

Với E được tính theo công thức (1-4) sau

$$\text{E}=56+\frac{Y+y-2003}{1.5}$$

Trong 5 tiêu chuẩn trên thì tiêu chuẩn thứ 2 và thứ 5 là liên quan đến tính chất riêng rẽ cần phải có đối với các số nguyên tố còn hai tiêu chuẩn 3 và 4 quy định cho quan hệ giữa cặp số nguyên tố p, q được dùng tạo nên mỗi modulo N=pq. Như vậy chương trình sinh số nguyên tố dùng cho hệ mật RSA phải được thiết kế theo yêu cầu sau.

Input: Hai số nguyên n và E được quy định tại tiêu chuẩn 1 và công thức (1-4).

Output: số nguyên tố p có kích thước $\frac{n}{2}$ bit và $\lambda(p\pm1)$ có ước nguyên tố lớn không dưới 2E bit.

Những số nguyên tố thoả mãn điều kiện tại đầu ra của thuật toán trên từ nay chúng ta sẽ gọi là các số "RSA-mạnh".

Ngoài ra cặp số nguyên tố p, q dùng để tạo ra mỗi modulo N=pq cần thoả mãn hai tiêu chuẩn 3 và 4. Cặp các số RSA-mạnh thoả mãn hai điều kiện nêu trên được gọi là cặp có "quan hệ mạnh".

Toàn bộ các trình bày trong chương này nhằm giải quyết yêu cầu trên. Đóng góp chính của chúng tôi là xây dựng được một công cụ đơn giản nhưng hiệu quả trong việc sinh tham số cho hệ mật RSA. Thuật toán sinh số cặp số có quan hệ mạnh mà chúng tôi tìm ra được có lẽ là một đóng góp mới mặc dù ý tưởng để thực hiện nó cũng chỉ là mô phỏng theo Gordon. Một chú ý có tính thực tiễn là với thuật toán mà chúng tôi xây dựng được thì mọi tiêu chuẩn đều được thoả mãn trong các tham số được sinh.

## 1. THUẬT TOÁN KIỂM TRA TÍNH NGUYÊN TỐ

### Mở đầu

Thông thường để sinh các số nguyên tố lớn người ta thường dựa trên một thuật toán kiểm tra tính nguyên tố của các số nguyên. Thuật toán kiểm tra này được hiểu như một hàm

$$PrimeTest:\mathrm{N}\rightarrow\{\mathrm{TRUE};\ \mathrm{FALSE}\}$$

với *PrimeTest*(p)=TRUE khi và chỉ khi hay ít ra cũng phải là khi p là số nguyên tố.

Khi này thuật toán sinh sinh các số nguyên tố n bit được thực hiện như sau.

Input : số nguyên n.

Output: số nguyên tố p có kích thước n bit.

1.Do

1.1.p=*Random*($2^{n-1}$, $2^n$).

2.Until *PrimeTest*(p)==TRUE

3.Return p.

ở trên hàm *Random* với đầu vào là cặp các số nguyên dương a<b và đầu ra là một số ngẫu nhiên y thoả mãn a<y<b.

Thực tế là trên các chương trình sinh số nguyên tố lớn cung cấp miễn phí trên thế giới chỉ sử dụng thuật toán Muller-Rabin để xây dựng hàm *PrimeTest*(.) mà chúng đều biết rằng thuật toán này chỉ thoả mãn yêu cầu đưa ra của chúng ta chỉ trong trường hợp giả thiết Riemann mở rộng là đúng rất tiếc là giả thiết này cho đến nay vẫn chưa được chứng minh! Trong các nghiên cứu của riêng mình, chúng tôi chọn cách tiếp cận khác với cách đã đưa ra ở trên để thiết kế cho thuật toán sinh số nguyên tố lớn đó là phương pháp sinh truy hồi theo độ dài của số cần sinh. Nói một cách khác là để sinh các số nguyên tố n bit chúng tôi sẽ sinh một số nguyên tố <n bit rồi từ các số nguyên tố sinh được này tiến hành sinh số n bit.

## 1.1. Một số kết quả chuẩn bị

Trước hết chúng ta làm quen với hai định lý rất kinh điển trong lý thuyết số dưới đây (xem [Riesel], định lý 4.3 trang 103 và định lý 4.8 trang 121-123).

***Định lý Pocklington***

*Cho số N=RF+1 với gcd(R,F)=1. Nếu mỗi ước nguyên tố p của F tìm được số a sao cho.*

*(i).* $a^{N-1} \equiv 1 \pmod N$

*(ii).* $gcd(a^{\frac{N-1}{p}} - 1, N) = 1$

*Thì mọi ước nguyên tố q của N đều có dạng q=xF+1.*

***Định lý Lucas-Pocklington***

*Cho số N=RF-1 với gcd(R,F)=1 và D với gcd(D,N)=1 và* $\left(\frac{D}{N}\right)$*=-1. Nếu mỗi ước nguyên tố p của F tìm được số* $u=a+b\sqrt{D} \in Z_N(\sqrt{D})$ *sao cho.*

*(i).* $u^{N+1} \in Z_N$

*(ii).* $u^{\frac{N-1}{p}} = A+B\sqrt{D}$ *với gcd(B,N)=1*

*Thì mọi ước nguyên tố q của N đều có dạng q=xF±1 trong đó ít nhất một ước có dạng xF-1.*

Từ hai định lý trên chúng ta thu được hệ quả sau.

***Hệ quả 2.1***

*Cho* $a \equiv 1 \pmod{F_1}$ *và* $a \equiv -1 \pmod{F_2}$ *với* $\gcd(F_1,F_2)=1$ *và* $0 \le a < F = F_1F_2$ *, khi đó.*

*(i). Mọi số N với* $N \equiv 1 \pmod{F_1}$ *và* $N \equiv -1 \pmod{F_2}$ *đều có dạng* $N \equiv a \pmod{F}$*.*

*(ii). Nếu N thoả mãn giả thiết của định lý Pocklington đối với* $F_1$ *và giả thiết Lucas-Pocklington đối với* $F_2$ *thì mọi ước nguyên tố q của N đều có một trong các dạng sau:*

$$q=xF+a \text{ hoặc } q=xF+1 \qquad (2\text{-}1)$$

*trong đó có ít nhất một ước có dạng xF+a.*

Với những kết quả trên chúng ta thu được một số điều kiện đủ để kiểm tra tính nguyên tố của một số lớp số nguyên như sau.

***Điều kiện Pocklington 2.2***

*Cho* $N=xF+1 \le F^3$*. Nếu N thoả mãn điều kiện của định lý Pocklington, khi đó.*

*(i).Nếu* $N \le F^2$ *thì N là số nguyên tố.*

*(ii).Nếu* $F^2 \le N \le F^3$ *và* $B^2-4A$ *không chính phương thì N là số nguyên tố.*

*ở đây $N=AF^2+BF+1$ với $0\le B<F$.□*

***Điều kiện Lucas 2.3***

*Cho $N=xF-1\le F^3$. Nếu N thoả mãn điều kiện của định lý Lucas-Pocklington, khi đó.*

*(i).Nếu $N\le F^2$ thì N là số nguyên tố.*

*(ii).Nếu $F^2\le N\le F^3$ và nếu cả hai $B^2+4A$ và $(F-B)^2+4(A-1)$ đều không chính phương thì N là số nguyên tố.*

*ở đây $N=AF^2+BF-1$ với $0\le B<F$.□*

***Điều kiện Lucas-Pocklington 2.4***

*Cho $N=xF+a\le F^2$ với $0\le a<F=F_1F_2$ sao cho $a\equiv 1 \pmod{F_1}$, $a\equiv -1 \pmod{F_2}$ và $gcd(F_1,F_2)=1$.*

*Nếu N thoả mãn điều kiện của hệ quả 2.1 thì là số nguyên tố.□*

## 1.2. Một số thuật toán kiểm tra tính nguyên tố

Các thuật toán kiểm tra tính nguyên tố mà chúng tôi trình bày trong mục này chỉ kiểm tra được chính xác tính nguyên tố của các số nguyên với một số dạng nhất định. Chúng được trình bày nhằm phục vụ việc tạo ra các số nguyên tố trong từng lớp số tương ứng đó cho nên việc trình các thuật toán này được thể hiện dưới dạng các hàm với đầu ra là một biến boolean TRUE hoặc FALSE.

### *1.2.1. Hàm PocklingtonPrimeTest*

Hàm

$$PocklingtonPrimeTest(.,F): \mathbf{N}_{Pock}(F)\rightarrow\{\text{TRUE; FALSE}\}$$

với $\mathbf{N}_{\text{Pock}}(F)=\{x=AF^2+BF+1: 0\le A,B<F, F=\prod_{i=1...r} p_i^{c_i}\}$.

là hàm kiểm tra tính nguyên tố của các số nguyên $x\in\mathbf{N}_{\text{Pock}}(F)$ trên cơ sở của của điều kiện Pocklington. Thuật toán để thực hiện hàm này như sau.

Input: $x=AF^2+BF+1$ với $0\le A,B<F$ và $F=\prod_{i=1...r} p_i^{c_i}$

Output: TRUE khi và chỉ khi x là số nguyên tố.

1. $i\leftarrow 0$;

2. Do

    2.1. $i\leftarrow i+1$; $p\leftarrow p_i$; ok←FALSE;

    2.2. Do

        2.2.1. a=*Random*(0;x);

        2.2.2. if $a^{x-1}\neq 1 \pmod x$ return FALSE; exit;

        2.2.3. $d\leftarrow \gcd(a^{\frac{x-1}{p}}-1,x)$;

        2.2.4. if (1<d<x) return FALSE; exit;

        2.2.5. ok←(d==1);

    2.3. Until (ok==TRUE);

3. Until (i==r).

4. if (B==0) return TRUE; exit;

   else  if ($B^2-4A==Q^2$) return FALSE; exit;

         else return TRUE; exit;

### *1.2.2. Hàm LucasPrimeTest*

Hàm

*LucasPrimeTest (.,F)*: $\mathbf{N}_{\text{Lucas}}(F)\rightarrow\{\text{TRUE; FALSE}\}$

với $\mathbf{N}_{\text{Lucas}}(F)=\{x=AF^2+BF-1: 0\le A,B<F, F=\prod_{i=1\ldots r} p_i^{c_i}\}$.

là hàm kiểm tra tính nguyên tố của các số nguyên $x\in\mathbf{N}_{\text{Lucas}}(F)$ trên cơ sở của của điều kiện Lucas. Thuật toán để thực hiện hàm này như sau.

Input: $x=AF^2+BF-1$ với $0\le A,B<F$ và $F=\prod_{i=1\ldots r} p_i^{c_i}$

Output: TRUE khi và chỉ khi x là số nguyên tố.

1. i←0; D with ((gcd(D,N)==1) && ($\left(\frac{D}{N}\right)$==-1));

2. Do

    2.1. i←i+1; p←$p_i$; ok←FALSE;

    2.2. Do

        2.2.1. a=*Random*(0;x); b=*Random*(0;x);

        2.2.2. if $(a+b\sqrt{D})^{x+1}=A+0\sqrt{D}$ return FALSE; exit;

        2.2.3. $(a+b\sqrt{D})^{\frac{x-1}{p}}=A+B\sqrt{D}$; d←gcd( B,x);

        2.2.4.if (1<d<x) return FALSE; exit;

        2.2.5. ok←(d==1);

    2.3. Until (ok==TRUE);

3. Until (i==r).

4. if (B==0) return TRUE; exit;

   else  if ($B^2-4A==Q^2$) return FALSE; exit;

         else return TRUE; exit;

### *1.2.3. Hàm LucasPocklingtonPrimeTest*

Hàm

*LucasPocklingtonPrimeTest*(.,$F_1$,$F_2$): $\mathbf{N}_{\text{LucasPock}}(F_1,F_2)\rightarrow$ {TRUE; FALSE}

với

$\mathbf{N}_{\text{LucasPock}}(F_1,F_2)$=

$$\{x=AF+a: 0\le A<F, F=F_1F_2, F_1=\prod_{i=1\ldots r} p_i^{c_i}, F_2=\prod_{i=1\ldots s} q_i^{d_i}, \gcd(F_1,F_2)=1\}.$$

là hàm kiểm tra tính nguyên tố của các số nguyên $x\in\mathbf{N}_{\text{LucasPock}}(F_1,F_2)$ trên cơ sở của của điều kiện Lucas-Pocklington. Thuật toán để thực hiện hàm này như sau.

Input: $x\in\mathbf{N}_{\text{LucasPock}}(F_1,F_2)$.

Output: TRUE khi và chỉ khi x là số nguyên tố.

1. if ($x\le F_1^3$) return *PocklingtonPrimeTest (x,$F_1$)*

   else if ($x\le F_2^3$) return *LucasPrimeTest (x,$F_2$)*)

   else return

   ((*PocklingtonPrimeTest(x,$F_1$)*)&&( *LucasPrimeTest(x,$F_2$)*));

## 2. THUẬT TOÁN SINH SỐ NGUYÊN TỐ BẰNG PHƯƠNG PHÁP TĂNG DẦN ĐỘ DÀI

Phần này chúng ta đề cập đến một phương pháp sinh các số nguyên tố cỡ n bit thông qua việc sinh các số nguyên tố có độ dài bit nhỏ hơn n mà chúng ta gọi là phương pháp tăng dần độ dài. Một thực tế là việc sinh các số nguyên tố nhỏ hơn là dễ hơn nên phương pháp dãn dần độ dài sẽ hứa hẹn cho chúng ta một thuật toán nhanh. Hình thức trình bày bày các thuật toán cũng giống như các mục trước đó là chúng tôi cố gắng diễn đạt chúng dưới dạng các hàm với đầu ra là các số nguyên tố.

## 2.1. Một số hàm sinh số nguyên tố đơn giản

### *2.1.1. Hàm sinh các số nguyên tố không qua 32 bit*

*SmallPrimeGenerator*:{17,18,...,32}→**P**.

Input: k∈{17,18,...,32}.

Output: x∈**P** với độ dài k bit.

Hàm được thực hiện theo phương pháp sàng với cơ sở là tất cả các số nguyên tố không quá $2^{16}$.

### *2.1.2. Hàm sinh các số nguyên tố từ k+1 đến 3k bit từ nhân nguyên tố k bit*

*LucasPockPrimeGenerator(p,.,.)*:{k+1, k+2,...,3k-1}x{0;1}→**P**. với p là số nguyên tố và k là độ dài bit của p.

Input: n∈{k+1, k+2,...,3k-1} và b∈{0;1}.

Output: x∈**P** với độ dài n bit.

1. $R_{min}$←min{r: rp≥$2^{n-1}$, r chẵn}; $R_{max}$←max{r: rp≤$2^n$, r chẵn}; ok←FALSE;

2. do

    2.1. r←*Random*($R_{min}$;$R_{max}$);

    2.2. if (b==0) x←ry+1

        else x←ry-1;

    2.3. if (b==0) ok←*PocklingtonPrimeTest*(x,p)

        else ok←*LucasPrimeTest*(x,p);

3. until (ok==TRUE);

4. return x;

## 2.2. Thuật toán

Thuật toán dãn dần độ dài để sinh số nguyên tố lớn được xây dựng thành một hàm ký hiệu là *PrimeGenerator*(.) với

Input: $n \in \mathbf{N}$.

Output: $x \in \mathbf{P}(n)$.

1. k←*Random*(17;33);

2. x←*SmallPrimeGenerator*(k);

3. while ($k < \frac{n}{3}$) do

    3.1. k←*Random*(k+1;3k-1);

    3.2. b←*Random*(2);

    3.3. x←*LucasPockPrimeGenerator*(x,k,b);

4. b←*Random*(2);

5. x←*LucasPockPrimeGenerator*(x,n,b);

6. return x;

## 2.3. Đánh giá thuật toán

Trong mục này chúng ta xem xét thuật toán sinh số nguyên tố lớn theo phương pháp dãn dần độ dài theo góc độ chính là mật độ của các số nguyên tố có thể sinh được theo thuật toán trong tập số các số nguyên tố. Để thuận lợi cho việc đánh giá trên trước hết chúng ta phân tích và rút ra một số kết luận chung phục vụ cho việc xem xét nói trên.

### *2.3.1. Số lần dãn trung bình*

Ta biết rằng để có được một số nguyên tố n bit theo thuật toán dãn dần độ dài

mô tả ở trên thì:

*Tại bước 2 của thuật toán ta đã có một số nguyên tố k bit với 16<k<33.

*Để đạt được số nguyên tố đúng n bit tại đầu ra thì giả sử rằng chúng ta cần thực hiện m lần dãn độ dài trong bước 3 thì rõ ràng số lần dãn độ dài trong thuật toán sẽ là m-1.

*Độ dài sau mỗi lần dãn theo bước 3.1 thì được lấy ngẫu nhiên trong khoảng (k+1;3k-1) với k là độ dài cũ, như vậy độ dài trung bình sau mỗi lần dãn là tăng gấp đôi. Tóm lại số lần dãn trung bình của thuật toán ký hiệu là d sẽ được tính theo công thức.

$$d=log_2(n-16)+1. \qquad (2\text{-}2)$$

### *2.3.2. Mật độ số nguyên tố sinh được*

***Kết quả 2.5***. *Tỷ lệ số nguyên tố n bit sinh được từ thuật toán trên tổng số các số nguyên tố n bit là*

$$\rho(n) \approx \left(\frac{728}{729}\right)^{\log_2(n-16)+1}. \qquad (2\text{-}3)$$

Chứng minh.

Theo công thức (1-11) của chương trước ta biết rằng xác suất để một số x là B-trơn bằng $\rho(B,x)\approx\left(\frac{\ln x}{\ln B}\right)^{-\left(\frac{\ln x}{\ln B}\right)}$, với $B=2^k$ và $x=2^{3k}$ ta có $\rho(2^k, 2^{3k}) \approx \frac{1}{27}$.

Như vậy xác suất để số $2^{3k-1}<x<2^{3k}$ có x-1 và x+1 đều là $2^k$-trơn sẽ là.

$$\left(\frac{1}{27}\right)^2 = \frac{1}{729}.$$

Theo thuật toán dãn dần độ dài thì tất cả các số nguyên tố x sinh được sau mỗi lần dãn đều có tính chất là có ước nguyên tố của x-1 hoặc của x+1 với độ dài ít nhất là $\frac{1}{3}$ độ dài của số sinh được. Như vậy xác suất để xuất hiện những số này trong tổng số các số nguyên tố sẽ vào khoảng

$$1-\frac{1}{729}=\frac{728}{729}>99.8\%.$$

Như vậy sau d lần dãn độ dài thì ta có

$$\rho(n)\approx\left(\frac{728}{729}\right)^{d}.$$

Theo (2-2) thì số lần thực hiện việc dãn độ dài là $d=\log_2(n-16)+1$, nên ta có ngay điều cần chứng minh.□

## 3. SINH SỐ NGUYÊN TỐ RSA-MẠNH

### 3.1. Mở đầu

Một thuận lợi cho việc xây dựng phần mềm sinh các số nguyên tố RSA-mạnh là có được thuật toán do Gordon đưa ra từ năm 1984 (xem [Gordon]). Mặc dù rằng khái niệm mạnh cho các số nguyên tố dùng trong hệ mật RSA trong mỗi tài liệu có đôi chút khác nhau tuy nhiên có điểm tương đồng là đều quan tâm chủ yếu và trước hết đến tính có ước nguyên tố lớn của p-1 và p+1. Ý tưởng của thuật toán Gordon là sinh trước các nhân nguyên tố $p_0\neq p_1$ thoả mãn điều kiện về độ lớn (không dưới một ngưỡng nào đó chẳng hạn là E bit với E chọn trước) rồi thực hiện tìm số nguyên tố trong lớp các số nguyên y thoả mãn điều kiện

$$y\equiv 1 \pmod{p_0} \text{ và } y\equiv -1 \pmod{p_1}. \qquad (2\text{-}4)$$

Các số nguyên nói trên có dạng

$$y=xF+a \qquad (2\text{-}5)$$

với $F=p_0p_1$ và $a\equiv(p_1^{p_0-1}-p_0^{p_1-1}) \pmod F$.

Rõ ràng giá trị a là thoả mãn điều kiện (2-4) và do đó mọi số y trong (2-5) đều thoả mãn điều kiện (2-4). Mặt khác theo định lý Dirichlet cho phép ta luôn tìm được các số nguyên tố trong lớp (2-5) với xác suất là

$$\rho \approx \frac{F}{\varphi(F)\ln y} = \frac{p_0 p_1}{(p_0 - 1)(p_1 - 1)\ln y}. \qquad (2\text{-}6)$$

## 3.2. Thuật toán Gordon

### *3.2.1. Hàm PrimeP-1Generator(k)*

Để sinh được các số nguyên tố RSA-mạnh, chúng ta cần đến một hàm sinh các số nguyên tố p với p-1 có ước nguyên tố k bit. Hàm này có một biến đầu vào là số nguyên dương k và được ký hiệu là *PrimeP-1Generator*(.) với.

Input : số tự nhiên k.

Ouput: p nguyên tố với p-1 có ước nguyên tố đúng k bit.

1. r←*PrimeGenerator*(k);

2. x←2;

3. do

    3.1. p←x*r+1;

    3.2. x←x+2;

4. until (*PocklingtonPrimeTest*(p,r)==TRUE);

5. return p;

Chú ý rằng số nguyên tố p sinh được từ hàm *PrimeP-1Generator*(k) với p-1 có ước nguyên tố là r với k bit và nó được chọn là số đầu tiên thoả mãn điều kiện này.

### *3.2.2. Dùng thuật toán Gordon xây dựng hàm sinh các số RSA-mạnh*

Thuật toán Gordon mà chúng tôi áp dụng ở đây được xây dựng là một hàm ký hiệu là *StrongPrimeGenerator*(n,E) với.

Input : n, E là các số nguyên dương.

Output: p nguyên tố n bit với φ(p-1) và φ(p+1) đều có ước nguyên tố không dưới E bit (số RSA-mạnh).

1. $k_0 \leftarrow$*Random*(E;n); $k_1 \leftarrow$*Random*(E;n);
2. $p_0 \leftarrow$*PrimeP-1Generator*($k_0$); $p_1 \leftarrow$ *PrimeP-1Generator*($k_1$); (Chú ý: $p_0 \neq p_1$)
3. $F \leftarrow p_0 p_1$;
4. $a \leftarrow ( p_1^{p_0-1} - p_0^{p_1-1} ) \pmod F$
5. Xmax←*max*{x: xF+a với n bit}; Xmin←*min*{x: xF+a với n bit};
6. do
    6.1. x←*Random*(Xmin;Xmax);
    6.1. p←x*F+a;
7. until (*LucasPocklingtonPrimeTest*(p,$p_0$,$p_1$)==TRUE);
8. return p.

### 3.3. Đánh giá lực lượng

Trong phần 2, công thức (2-3) đã cho ta một ước lượng về tỷ lệ giữa số các số các số nguyên tố n bit có thể sinh được bởi phương pháp dãn dần độ dài vtrên tổng số các số nguyên tố n bit. Tại đây sự quan tâm của chúng ta là hướng vào đánh giá tương tự  nhưng cho các số nguyên tố RSA-mạnh. Sự khác biệt giữa các số nguyên tố nói chung và các số nguyên tố RSA-mạnh là sự không cho phép tính $2^{2E}$-trơn của cả p-1 và p+1. Chính điều kiện trên đã tạo ra cho việc sinh số nguyên tố RSA-mạnh phù hợp hơn với phương pháp dãn dần độ dài. Trong giả  thiết chúng ta có thể sinh được toàn bộ các số nguyên tố thì hàm StrongPrimeGenerator chúng ta thiết kế ở đây có thể sinh được số nguyên tố RSA-mạnh cho bởi các kết quả sau.

***Định lý 2.6.*** *Mật độ số RSA-mạnh trong các số nguyên tố n bit được cho bởi công thức sau.*

$$\xi_m = 1 - 2\left(\frac{m}{2E+1}\right)^{-\left(\frac{m}{2E+1}\right)} \qquad (2\text{-}7)$$

Chứng minh.

Ta biết RSA-mạnh là số nguyên tố với λ(p-1) và λ(p+1) có ước nguyên tố không dưới 2E bit như vậy p-1 và p+1 phải có ước nguyên tố không dưới 2E+1 bit. Điều kiện sau suy ra p-1 và p+1 đồng thời không là số $2^{2E+1}$-trơn. Theo công thức (1-11) thì xác suất để số m bit là $2^{2E+1}$-trơn bằng $\left(\frac{m}{2E+1}\right)^{-\left(\frac{m}{2E+1}\right)}$ vậy để có p-1 hoặc p+1 là $2^{2E+1}$-trơn là $2\left(\frac{m}{2E+1}\right)^{-\left(\frac{m}{2E+1}\right)}$ hay xác suất để p là RSA-mạnh bằng $\xi_m=1-2\left(\frac{m}{2E+1}\right)^{-\left(\frac{m}{2E+1}\right)}$ và đây là công thức cần chứng minh.□

***Định lý 2.7.*** *Mật độ số RSA-mạnh có thể sinh được từ hàm StrongPrimeGenerator trong các số nguyên tố n bit ký hiệu là* $\zeta_m$ *sẽ.*

*(i).* $\zeta_m \geq \frac{127}{128}$ *nếu 8E<m.*

*(ii).* $\zeta_m=1-2\left(\frac{m}{2E+1}\right)^{-\left(\frac{m}{2E+1}\right)}$ *trong trường hợp ngược lại.*

Chứng minh.

Trước hết ta thấy rằng hiệu lực của LucasPocklingtonPrimeTest(x,$p_0$,$p_1$) như nêu trong điều kiện Lucas-Pocklington 2.4 là số bit của tích $F=p_0p_1$ là không dưới 0.5m, hiển nhiên điều kiện trên sẽ được kéo theo khi số bit của $p_0$ và $p_1$ đều không dưới 0.25m. Với lập luận đã sử dụng trong chứng minh định lý 2.6 ta có ngay nếu 8E≥m thì thuật toán sinh được toàn bộ các số RSA-mạnh và điều này theo định lý 2.6 ta chứng minh được (ii). Ngược lại ta chỉ sinh được các số RSA-mạnh với điều kiện số bit của tích $F=p_0p_1$ là không dưới 0.5m như vậy các số này sẽ không ít hơn các số có cả p-1 và p+1 đều không $2^{0.25m}$-trơn suy ra $\zeta_m \geq 1-2*4^{-4}=\frac{127}{128}$. Tóm lại định lý đã được chứng minh.□

# 4. SINH CẶP SỐ NGUYÊN TỐ CÓ QUAN HỆ MẠNH

## 4.1. Mở đầu

Mặc dù rằng trong [Silverman] có đưa ra tiêu chuẩn p-q có ước nguyên tố lớn nhưng tác giả của bài viết này cũng chỉ nhận định rằng *"điều kiện xem ra không thực hiện được"* với lý do chính đáng là để kiểm tra được nó ta phải đối đầu với bài toán phân tích số, một bài toán vốn được coi là khó!?. Cũng trong tài liệu này, tác giả đưa ra một giải pháp là chỉ kiểm tra phân tích theo ECM nhằm chỉ ra được rằng không còn nhân tử nguyên tố nhỏ.

Trong mục này chúng tôi phỏng theo ý tưởng của Gordon và đã đưa ra một thuật toán nhằm sinh được cặp số nguyên tố RSA-mạnh p và q đồng thời thoả mãn điều kiện gcd(p-1;q-1) có ước nguyên tố lớn. Thành công lớn nhất mà chúng tôi đã đạt được là đã đưa ra một thuật toán sinh được cặp số nguyên tố p, q đều là RSA-mạnh đồng thời thoả mãn điều kiện có quan hệ mạnh mà không cần đến sự tham gia của một thuật toán phân tích số nguyên.

## 4.2. Thuật toán sinh cặp số RSA-mạnh p và q với gcd(p-1;q-1) có ước nguyên tố không dưới E bit

Chúng ta đã biết, với E là số mũ an toàn định nghĩa trong chương 1 thì số nguyên tố RSA-mạnh là số thoả mãn điều kiện φ(p-1) và φ(p+1) có ước nguyên tố không dưới 2E bit. Mặt khác tiêu chuẩn về cặp số p, q có quan hệ mạnh trước hết là gcd(p-1;q-1) có ước nguyên tố không dưới E bit. Thuật toán dưới đây cho phép ta sinh được các số nguyên tố thoả mãn các điều kiện trên.

Thuật toán được thiết kế thành hàm với 2 biến đầu vào là các số nguyên dương n và E và có 2 biến đầu ra là các số nguyên tố p và q thoả mãn các tiêu chuẩn trong hệ tiêu chuẩn đã đưa ra. Hàm sẽ được ký hiệu là *RSA-Generator*(.,.).

### *4.2.1. Hàm GordonGenerator(.,.,.)*

Để phục vụ việc xây dựng hàm *RSA-Generator*(.,.,.) chúng ta cần đến hàm sinh một cặp số nguyên tố p và q là RSA-mạnh thoả mãn điều kiện gcd(p-1;q-1) có ước nguyên tố lớn.

Để có được điều kiện gcd(p-1;q-1) có ước nguyên tố lớn thì chúng ta chỉ cần thay đổi một chút thuật toán của Gordon mỗi khi sinh một cặp số RSA-mạnh dùng cho mỗi modulo, thuật toán được thiết kế thành hàm ký hiệu là *GordonGenerator*(n, $p_0$, $p_1$, r) và thuật toán thực hiện như sau.

Input : n là số nguyên dương;

r là số nguyên tố không dưới E bit.

$p_0$, $p_1$, là các số nguyên tố với $p_0$-1 và $p_1$-1 có ước nguyên tố không dưới 2E bit.

Output: p nguyên tố n bit với r và $p_0$ là ước của p-1, $p_1$ là ước của p+1.

1. a←*CRT*(1,-1,1,$p_0$,$p_1$,r); F←$p_0p_1$r.
2. Xmax←*max*{x: xF+a với n bit}; Xmin←*min*{x: xF+a với n bit};
3. do
   3.1. x←*Random*(Xmin;Xmax);
   3.2. p←x*F+a;
8. until (*LucasPocklingtonPrimeTest*(p,r*$p_0$,$p_1$)==TRUE);
9. return p.

Chú ý, ở trên hàm *CRT*($a_0$,$a_1$,$a_2$,$m_0$,$m_1$,$m_2$) với 6 biến đầu vào và một biến đầu ra đều là các số nguyên dương được thực hiện theo kết quả của định lý phần dư Trung hoa (CRT) như sau.

input: các số nguyên $a_0$, $a_1$, $a_2$, $m_0$, $m_1$, $m_2$, với $m_0$, $m_1$, $m_2$ nguyên tố cùng nhau,

output: số nguyên a thoả mãn $0 \le a < F = m_0m_1m_2$ và $a \equiv a_i \pmod{m_i}$.

### *4.2.2. Hàm RSA-Generator(.,.)*

Hiển nhiên các số nguyên tố được sinh từ hàm *GordonGenerator* đều là các số RSA-mạnh, ngoài ra chúng còn có thêm tính chất là p-1 có ước là r. Nhờ đặc tính trên của hàm *GordonGenerator* nên để sinh được cặp số nguyên tố RSA-mạnh đồng thời có quan hệ mạnh thì chúng ta chỉ cần thực hiện sinh chúng từ hàm *GordonGenerator* với cùng tham số đầu vào r. Tóm lại hàm *RSA_Generator* được thiết kế như sau.

Input: các số nguyên dương n, E.

Output:hai số nguyên tố RSA-mạnh p, q có quan hệ mạnh.

1. k←*Random*(E;n);

   $k_0$←*Random*(2*E;n); $k_1$←*Random*(2*E;n);

   $h_0$←*Random*(2*E;n); $h_1$←*Random*(2*E;n);

2. r←*PrimeGenerator*(k);

   $p_0$←*PrimeP-1Generator*($k_0$); $p_1$← *PrimeP-1Generator*($k_1$);

   $q_0$←*PrimeP-1Generator*($k_0$); $q_1$← *PrimeP-1Generator*($k_1$);

   (Chú ý: r, $p_0$, $p_1$, $q_0$, $q_1$ là các số khác nhau)

3.do

   3.1. p←*GordonGenerator*(n, $p_0$, $p_1$, r);

   q←*GordonGenerator*(n, $q_0$, $q_1$, r);

   3.2. m←max{*SoBit*(*gcd*(p±1;q±1)};

4. until (m<$\frac{n-2E}{4}$);

5. return p, q;

ở trên hàm *gcd*(x,y) trả về giá trị ước chung lớn nhất của x và y còn hàm *SoBit*(x) tra về số bit tối thiểu cần đến để biểu diễn số nguyên x (dạng nhị phân).

lý.□

## TÀI LIỆU THAM KHẢO.


[Lều Tân]. Lều Đức Tân. Một số thuật toán kiểm tra tính nguyên tố đối với một số lớp số. Luận án phó tiến sỹ khoa học toán lý, Hà nội 1994.

[Blanke-Seroussi-Smart] Ian Blanke, Gadiel Seroussi & Nigel Smart. Elliptic Curves in Cryptography. Cambridge Universty press 1999.

[Gordon] D. M. Gordon, Strong Primes Are Ease to Find, Advances in Cryptology-Proceedings of EUROCRYPT 84 (LNCS 209), 216-223, 1985.

[Riesel]. Hans Riesel, Prime Number and Computer Methods for Factorization, Progress in Mathematics, 57, 1985.

[RivestSilverman] R. L. Rivest and R. D. Silverman. Are Strong Primes Needed for RSA? To apear.

[Silverman] Robert D. Silverman. Fast Generation of Random, Strong RSA Primes. The Technical Newsletter of RSA Laborastories. Spring 1997.

[Stephens] N.M.Stephens. Lenstra's Factorisation Based On Elliptic Curves. Springer-Verlag 1998, pp 409-416.