Nội dung Ôn tập
Các hệ mật Khóa công khai

1. Giới thiệu mở đầu.

1.1. Sự ra đời của mật mã khoá công khai.

Trong chương I ta đã giới thiệu qua định nghĩa của các khái niệm hệ mật mã khoá đối xứng và hệ mật mã khoá công khai. Sự ra đời của khái niệm hệ mật mã khoá công khai là một tiến bộ có tính chất bước ngoặt trong lịch sử mật mã nói chung, gắn liền với sự phát triển của khoa học tính toán hiện đại. Người ta có thể xem thời điểm khởi đầu của bước ngoặt đó là sự xuất hiện ý tưởng của W. Diffie và M.E. Hellman được trình bày vào tháng sáu năm 1976 tại Hội nghị quốc gia hàng năm của AFIPS (Hoa kỳ) trong bài *Multiuser cryptographic techniques*. Trong bài đó, cùng với ý tưởng chung, hai tác giả cũng đã đưa ra những thí dụ cụ thể để thực hiện ý tưởng đó, và mặc dù các thí dụ chưa có ý nghĩa thuyết phục ngay đối với tác giả, thì ý tưởng về các hệ mật mã khoá công khai cũng đã rất rõ ràng và có sức hấp dẫn đối với nhiều người. Và ngay sau đó, công việc tìm kiếm những thể hiện cụ thể có khả năng ứng dụng trong thực tế đã bắt đầu thu hút sự quan tâm của nhiều chuyên gia. Một năm sau, năm 1977, R.L. Rivest, A. Shamir và L.M. Adleman đề xuất một hệ cụ thể về mật mã khoá công khai mà độ an toàn của hệ dựa vào bài toán khó "phân tích số nguyên thành thừa số nguyên tố", hệ này về sau trở thành một hệ nổi tiếng và mang tên là hệ RSA, được sử dụng rộng rãi trong thực tiễn bảo mật và an toàn thông tin. Cũng vào thời gian đó, M.O. Rabin cũng đề xuất một hệ mật mã khoá công khai dựa vào cùng bài toán số học khó nói trên. Liên tiếp sau đó, nhiều hệ mật mã khóa công khai được đề xuất, mà khá nổi tiếng và được quan tâm nhiều là các hệ: hệ McEliece được đưa ra năm 1978 dựa trên độ *NP*-khó của bài toán giải mã đối với các hệ mã cyclic tuyến tính, hệ Merkle- Hellman dựa trên tính *NP*-đầy đủ của bài toán xếp ba lô(knapsack problem), hệ mật mã nổi tiếng ElGamal dựa trên độ khó của bài toán lôgarit rời rạc, hệ này về sau được mở rộng để phát triển nhiều hệ tương tự dựa trên độ khó của các bài toán tương tự lôgarit rời rạc trên các cấu trúc nhóm cyclic hữu hạn, nhóm các điểm nguyên trên đường cong eliptic, v.v... Để tăng độ bảo mật, hệ mật mã ElGamal còn dùng với tư cách đầu vào cho thuật toán lập mật mã của mình, ngoài khoá công khai và bản rõ, một yếu tố ngẫu nhiên được chọn tuỳ ý, điều đó làm cho hệ mật mã trở thành một hệ mật mã xác suất khoá công khai. Một số hệ mật mã xác suất khoá công khai cũng được phát triển sau đó bởi Goldwasser-Micali và Blum-Goldwasser. Tất cả các hệ mật mã khoá công khai kể trên sẽ được trình bày trong chương này cùng với một số tính chất liên quan của chúng.

1.2. Một số bài toán cơ bản.

Sau đây ta sẽ nhắc lại một số bài toán số học được sử dụng đến khi xây dựng các hệ mật mã khoá công khai như nói ở trên. Các bài toán này phần lớn đã được trình bày trong chương II, một số được phát triển thêm cho các ứng dụng trực tiếp khi xây dựng các hệ mã cụ thể, ta liệt kê dưới đây một lần để thuận tiện cho các chỉ dẫn về sau.

Bài toán phân tích số nguyên (thành thừa số nguyên tố):

Cho số nguyên dương n , tìm tất cả các ước số nguyên tố của nó, hay là tìm dạng phân tích chính tắc của $n = p_1^{\alpha_1}.p_2^{\alpha_2}...p_k^{\alpha_k}$, trong đó p_i là các số nguyên tố từng cặp khác nhau và các $\alpha_i \geq 1$.

Bài toán này có liên hệ mật thiết với các bài toán thử tính nguyên tố hay thử tính hợp số của một số nguyên, nhưng với những gì mà ta biết đến nay, nó dường như khó hơn nhiều so với hai bài toán thử tính nguyên tố và tính hợp số.

Trong lý thuyết mật mã, bài toán này thường được sử dụng với các dữ liệu n là số nguyên Blum, tức các số nguyên dương có dạng tích của hai số nguyên tố lớn nào đó.

Bài toán RSA (Rivest-Shamir-Adleman) :

Cho số nguyên dương n là tích của hai số nguyên tố lẻ khác nhau, một số nguyên dương e sao cho gcd($e,\phi(n)$) =1, và một số nguyên c ; tìm một số nguyên m sao cho $m^e \equiv c \pmod{n}$.

Điều kiện gcd($e,\phi(n)$) =1 bảo đảm cho việc với mỗi số nguyên $c \in \{0,1,...,n-1\}$ có đúng một số $m \in \{0,1,...,n-1\}$ sao cho $m^e \equiv c \pmod{n}$.

Dễ thấy rằng nếu biết hai thừa số nguyên tố của n, tức là biết $n = p.q$ thì sẽ biết $\phi(n) = (p-1)(q-1)$, và từ đó, do gcd($e,\phi(n)$) =1 sẽ tìm được $d = e^{-1} \bmod \phi(n)$, và do đó sẽ tìm được $m = c^d \bmod n$. Như vậy, bài toán RSA có thể qui dẫn trong thời gian đa thức về bài toán phân tích số nguyên. Tuy rằng cho đến nay chưa có một chứng minh nào cho việc qui dẫn ngược lại nhưng nhiều người vẫn tin rằng hai bài toán đó là tương đương với nhau về độ phức tạp tính toán.

Bài toán thặng dư bậc hai :

Cho một số nguyên lẻ n là hợp số, và một số nguyên $a \in J_n$, tập tất cả các số a có ký hiệu Jacobi $\left(\frac{a}{n}\right)$=1. Hãy quyết định xem a có là thặng dư bậc hai theo mod$n$ hay không?

Trong lý thuyết mật mã, bài toán này cũng thường được xét với trường hợp n là số nguyên Blum, tức n là tích của hai số nguyên tố p và q , $n = p.q$. Ta chú ý rằng trong trường hợp này, nếu $a \in J_n$, thì a là thặng dư bậc hai theo mod$n$ khi và chỉ khi $\left(\frac{a}{p}\right)$=1, điều kiện này có thể thử được dễ dàng vì nó tương đương với điều kiện $a^{(p-1)/2} \equiv 1 \pmod{p}$. Như vậy, trong trường hợp này, bài toán thặng dư bậc hai có thể qui dẫn trong thời gian đa thức về bài toán phân tích số nguyên. Mặt khác, nếu không biết cách phân tích n thành thừa số nguyên tố thì cho đến nay, không có cách nào giải được bài toán thặng dư bậc hai trong thời gian đa thức. Điều đó củng cố thêm niềm tin rằng bài toán thặng dư bậc hai và bài toán phân tích số nguyên là có độ khó tương đương nhau.

Bài toán tìm căn bậc hai modn :

Cho một số nguyên lẻ n là hợp số Blum, và một số $a \in Q_n$, tức a là một thặng dư bậc hai theo mod$n$. Hãy tìm một căn bậc hai của a theo mod$n$, tức tìm x sao cho $x^2 \equiv a \pmod{n}$.

Nếu biết phân tích n thành thừa số nguyên tố, $n = p.q$, thì bằng cách giải các phương trình $x^2 \equiv a$ theo các mod$p$ và mod$q$, rồi sau đó kết hợp các nghiệm của chúng lại theo định lý số dư Trung quốc ta sẽ được nghiệm theo mod$n$, tức là căn bậc hai của a theo mod$n$ cần tìm. Vì mỗi phương trình $x^2 \equiv a$ theo mod$p$ và mod$q$ có hai nghiệm (tương ứng theo mod$p$ và mod$q$), nên kết hợp lại ta được bốn nghiệm, tức bốn căn bậc hai của a theo mod$n$. Người ta đã tìm được một số thuật toán tương đối đơn giản (trong thời gian đa thức) giải phương trình $x^2 \equiv a \pmod{p}$ với p là số nguyên tố. Như vậy, bài toán tìm căn bậc hai mod$n$ có thể qui dẫn trong thời gian đa thức về bài

toán phân tích số nguyên. Ngược lại, nếu có thuật toán $\triangleq$ giải bài toán tìm căn bậc hai mod$n$ thì cũng có thể xây dựng một thuật toán giải bài toán phân tích số nguyên như sau: Chọn ngẫu nhiên một số x với gcd(x,n) =1, và tính $a = x^2 \bmod n$. Dùng thuật toán $\triangleq$ cho a để tìm một căn bậc hai mod$n$ của a. Gọi căn bậc hai tìm được đó là y. Nếu $y \equiv \pm x \pmod n$, thì phép thử coi như thất bại, và ta phải chọn tiếp một số x khác. còn nếu $y \not\equiv \pm x \pmod n$, thì gcd($x$-$y$, n) chắc chắn là một ước số không tầm thường của n, cụ thể là p hay là q. Vì n có 4 căn bậc hai mod$n$ nên xác suất của thành công ở mỗi lần thử là 1/2, và do đó số trung bình (kỳ vọng toán học) các phép thử để thu được một thừa số p hay$q$ của n là 2, từ đó ta thu được một thuật toán giải bài toán phân tích số nguyên (Blum) với thời gian trung bình đa thức. Tóm lại, theo một nghĩa không chặt chẽ lắm, ta có thể xem hai bài toán phân tích số nguyên và tìm căn bậc hai mod$n$ là khó tương đương nhau.

Bài toán lôgarit rời rạc :

Cho số nguyên tố p, một phần tử nguyên thuỷ α theo mod$p$ (hay α là phần tử nguyên thuỷ của Z_p^*), và một phần tử $\beta \in Z_p^*$. Tìm số nguyên x ($0 \le x \le p - 2$) sao cho $\alpha^x \equiv \beta \pmod p$.

Trong mục 2.4.3 ta đã giới thiệu qua bài toán này, và biết rằng trong trường hợp chung, cho đến nay chưa có một thuật toán nào giải bài toán này trong thời gian đa thức.

Bài toán này cũng được suy rộng cho các nhóm cyclic hữu hạn như sau:

Bài toán lôgarit rời rạc suy rộng :

Cho một nhóm cyclic hữu hạn G cấp n, một phần tử sinh (nguyên thuỷ) α của G, và một phần tử $\beta \in G$. Tìm số nguyên x ($0 \le x \le n - 1$) sao cho $\alpha^x = \beta$.

Các nhóm được quan tâm nhiều nhất trong lý thuyết mật mã là: nhóm nhân của trường hữu hạn $GF(p)$ - đẳng cấu với nhóm Z_p^* của trường Z_p ,nhóm nhân $\mathbb{F}_{2^m}^*$ của trường hữu hạn $GF(2^m)$, nhóm nhân $Z_n^* = \{a : 0 \le a \le n-1, \gcd(a,n) = 1\}$ của trường Z_n với n là hợp số, nhóm gồm các điểm trên một đường cong elliptic xác định trên một trường hữu hạn, v.v...

Bài toán Diffie-Hellman :

Cho số nguyên tố p, một phần tử nguyên thuỷ α theo mod$p$ (tức phần tử sinh của Z_p^*), và các phần tử $\alpha^a \bmod p$ và $\alpha^b \bmod p$.

Hãy tìm giá trị $\alpha^{ab} \bmod p$.

Có thể chứng minh được rằng bài toán Diffie-Hellman qui dẫn được về bài toán lôgarit rời rạc trong thời gian đa thức. Thực vậy, giả sử có thuật toán $\triangleq$ giải bài toán lôgarit rời rạc. Khi đó, cho một bộ dữ liệu vào của bài toán Diffie-Hellman gồm p, α, $\alpha^a \bmod p$ và $\alpha^b \bmod p$; trước hết dùng thuật toán $\triangleq$ cho (p, α, $\alpha^a \bmod p$) ta tìm được a, và sau đó tính được $\alpha^{ab} \bmod p = (\alpha^b)^a \bmod p$. Người ta cũng chứng minh được hai bài toán lôgarit rời rạc và Diffie-Hellman là tương đương về mặt tính toán trong một số trường hợp, ví dụ p -1 là B-mịn với $B = O((\ln p)^c)$,c là hằng số.

Tương tự như với bài toán lôgarit rời rạc, ta cũng có thể định nghĩa các bài toán Diffie-Hellman suy rộng cho các nhóm cyclic hữu hạn khác.

Bài toán tổng tập con (hay bài toán KNAPSACK) :

Cho một tập các số nguyên dương $\{a_1, a_2, ..., a_n\}$ và một số nguyên dương s. Hãy xác định xem có hay không một tập con các a_j mà tổng của chúng bằng s. Một cách tương đương, hãy xác định xem có hay không các $x_i \in \{0,1\}$ $(1 \le i \le n)$ sao cho $\sum_{i=1}^{n} a_i x_i = s$.

Bài toán này là một bài toán $\mathcal{NP}$- đầy đủ, tức là thuộc lớp những bài toán khó mà cho đến nay chưa tìm được thuật toán giải chúng trong thời gian đa thức !

Bài toán giải mã đối với mã tuyến tính :

Mã tuyến tính là một lớp mã truyền tin có tính chất tự sửa sai được sử dụng trong kỹ thuật truyền tin số hoá. Không đi vào chi tiết của lớp mã này, ta có thể phát biểu trực tiếp bài toán giải mã đối với mã tuyến tính như sau:

Cho một ma trận cấp n xm A=(a_{ij}) gồm các thành phần là 0 hoặc 1, một vectơ $\boldsymbol{y}$ =(y_1,y_2,...,y_m) các giá trị 0 và 1, và một số nguyên dương K. Hỏi: có hay không một vectơ $\boldsymbol{x}$ =(x_1,x_2,...,x_n) gồm các số 0 hoặc 1 và có không nhiều hơn K số 1 sao cho với mọi j $(1 \le j \le m)$:

$$\sum_{i=1}^{n} x_i . a_{ij} \equiv y_j \pmod 2 \ ?$$

Chú ý rằng ở đây, $\boldsymbol{x}$ là vectơ thông tin, và $\boldsymbol{y}$ là vectơ mã, phép giải mã là tìm lại $\boldsymbol{x}$ khi nhận được $\boldsymbol{y}$, bài toán này tiếc thay lại là một bài toán khó; Berlekamp, McEliece và Tilborg năm 1978 đã chứng minh nó thuộc lớp các bài toán $\mathcal{NP}$- đầy đủ !

2. Hệ mật mã khoá công khai RSA.

2.1. Mô tả hệ mật mã RSA.

Sơ đồ chung của hệ mật mã khoá công khai được cho bởi

$$S = (P, C, K, E, D) \qquad (1)$$

trong đó P là tập ký tự bản rõ, C là tập ký tự bản mã, K là tập các khoá K , mỗi khoá K gồm có hai phần K =(K',K''), K' là khoá công khai dành cho việc lập mật mã, còn K'' là khoá bí mật dành cho việc giải mã. Với mỗi ký tự bản rõ $x \in P$, thuật toán lập mã E cho ta ký tự mã tương ứng y =E (K', x) $\in C$, và với ký tự mã y thuật toán giải mã D sẽ cho ta lại ký tự bản rõ x : D (K'', y) = D (K'', E (K', x)) =x.

Để xây dựng một hệ mật mã khoá công khai RSA, ta chọn trước một số nguyên n =$p.q$ là tích của hai số nguyên tố lớn, chọn một số e sao cho gcd(e, $\phi(n)$) =1, và tính số d sao cho

$$e.d \equiv 1 (\mathrm{mod}\, \phi(n)).$$

Mỗi cặp K =(K',K''), với K' =(n,e) và K'' = d sẽ là một cặp khoá của một hệ mật mã RSA cụ thể cho một người tham gia.

Như vậy, sơ đồ chung của hệ mật mã RSA được định nghĩa bởi danh sách (1), trong đó:

$P = C = Z_n$, trong đó n là một số nguyên Blum, tức là tích của hai số nguyên tố;

K = {K =(K',K''): K' =(n,e) và K'' = d, gcd(e, $\phi(n)$) =1, $e.d \equiv 1 (\mathrm{mod}\, \phi(n))$};

E và D được xác định bởi:

$$E(K', x) = x^e \bmod n, \text{ với mọi } x \in P,$$
$$D(K'', y) = y^d \bmod n, \text{ với mọi } y \in C.$$

Để chứng tỏ định nghĩa trên là hợp thức, ta phải chứng minh rằng với mọi cặp khoá $K=(K', K'')$, và mọi $x \in P$, ta đều có

$$D(K'', E(K', x)) = x.$$

Thực vậy, do $e.d \equiv 1 (\bmod \phi(n))$ ta có thể viết $e.d = t.\phi(n) + 1$. Nếu x nguyên tố với n, thì dùng định lý Euler (xem 2.1.3) ta có

$$D(K'', E(K', x)) = x^{ed} \equiv x^{t\phi(n)+1} \equiv x^{t\phi(n)}.x \pmod n = x.$$

Nếu x không nguyên tố với n, thì do $n = p.q$, hoặc x chia hết cho p và nguyên tố với q, hoặc x chia hết cho q và nguyên tố với p, và

$\phi(n) = (p-1).(q-1)$, trong cả hai trường hợp ta đều có

$$x^{t\phi(n)+1} \equiv x \pmod p,$$
$$x^{t\phi(n)+1} \equiv x \pmod q;$$

từ đó suy ra $x^{t\phi(n)+1} \equiv x \pmod n$, tức $D(K'', E(K', x)) = x$.

Thí dụ: Giả sử chọn $n = p.q = 2357.2551 = 6012707$, ta sẽ có $\phi(n) = (p-1).(q-1) = 2356.2550 = 6007800$. Chọn $e = 3674911$, và tính được $d = 422191$ sao cho $e.d \equiv 1 (\bmod \phi(n))$. Một người dùng A có thể chọn khoá công khai là $K' = (n = 6012707, e = 3674911)$ và giữ khoá bí mật $K'' = d = 422191$. Một đối tác B muốn gửi cho A một thông báo $x = 5234673$, sẽ dùng khoá công khai để tạo bản mật mã $y = x^e = 5234673^{3674911} \bmod 6012707 = 3650502$. A nhận được y, giải mã sẽ được bản rõ $x = 3650502^{422191} \bmod 6012707 = 5234673$.

2.2. Thực hiện hệ mật mã RSA.

Để thực hiện hệ mật mã RSA cho một mạng truyền tin bảo mật, ngoài việc xây dựng các chương trình tính toán hàm E (với tham biến đầu vào là n, e và x) và hàm D (với tham biến đầu vào là n, d và y), ta còn phải chọn cho mỗi người tham gia một bộ (n,e,d) để tạo các khoá công khai K' và khoá bí mật K''. Hệ mã của mỗi người tham gia chỉ có khả năng bảo mật khi $n = p.q$ là số nguyên rất lớn (và do đó p,q cũng phải là những số nguyên tố rất lớn); rất lớn có nghĩa là p,q phải có biểu diễn thập phân cỡ hơn 100 chữ số, do đó n có cỡ hơn 200 chữ số thập phân, hay $n \geq 10^{200}$!

Tính toán các số e,d, hay thực hiện các hàm E, D, đều chủ yếu là thực hiện các phép tính số học trên các số nguyên rất lớn; về vấn đề này trong mấy chục năm qua, khoa lập trình máy tính đã đề xuất nhiều chương trình máy tính làm việc rất có hiệu quả, ta có thể tham khảo để sử dụng khi thực thi các hệ mật mã RSA cũng như nhiều hệ mật mã khác.

2.3. Tính bảo mật của mật mã RSA.

Bài toán thám mã (khi chỉ biết bản mã) đối với mật mã RSA là: biết khoá công khai $K' = (n,e)$, biết bản mã $y = x^e \bmod n$, tìm x. Bài toán này chính là bài toán RSA được trình bày trong mục 4.1.2. Trong mục đó ta đã chứng tỏ rằng nếu biết hai thừa số p,q của n thì dễ tìm được x từ y, và nói chung có bằng chứng để coi rằng bài toán RSA (hay bài toán thám mã RSA) là có độ khó tương đương với bài toán phân tích số nguyên (Blum) thành thừa số nguyên tố. Do đó, giữ tuyệt mật khoá bí mật d, hay giữ tuyệt mật các thừa số p,q, là có ý nghĩa rất quyết định đến việc bảo vệ tính an toàn của hệ mật mã RSA.

Một mạng truyền tin bảo mật sử dụng sơ đồ các hệ mật mã RSA được xem là an toàn, nếu tuân thủ các điều kiện cơ bản: mỗi người tham gia phải độc lập lựa chọn các tham số n, e,d của riêng mình, chọn n cũng có nghĩa là chọn các thừa số p,q của n ($n = p.q$), và do có p,q nên tính được $\phi(n) = (p-1).(q-1)$, và từ đó tìm được e,d tương đối dễ dàng; nhưng cũng chính vì vậy mà sau khi đã chọn thì mỗi người tham gia phải giữ tuyệt đối bí mật các giá trị p,q,d , chỉ công bố khoá công khai (n,e) mà thôi.

Tuy nhiên, đó là điều kiện chung, còn trong thực tế vẫn có thể còn nhiều sơ hở mà người thám mã có thể lợi dụng để tấn công vào tính bảo mật của các hệ mã RSA khó mà lường trước hết được; sau đây là một số trường hợp đơn giản đã biết mà ta cần chú ý:

1.*Dùng môđuyn n chung*. Giả sử có hai người tham gia A và B cùng sử dụng một môđuyn chung n trong khoá công khai của mình, chẳng hạn A chọn khoá công khai (n,e) và giữ khoá bí mật d, B chọn khoá công khai (n,a) và giữ khoá bí mật b. Một người tham gia thứ ba C gửi một văn bản cần bảo mật x đến cả A và B thì dùng các khoá công khai nói trên để gửi đến A bản mật mã $y = x^e \bmod n$ và gửi đến B bản mật mã $z = x^a \bmod n$. Ta sẽ chứng tỏ rằng một người thám mã O có thể dựa vào những thông tin n,e,a,y,z trên đường công khai mà phát hiện ra bản rõ x như sau:

a. Tính $c = e^{-1} \bmod a$,
b. Sau đó tính $h = (ce-1)/a$,
c. Và ta được $x = y^c (z^h)^{-1} \bmod n$.

Thực vậy, theo định nghĩa trên, ce -1 chia hết cho a, và tiếp theo ta có: $y^c (z^h)^{-1} \bmod n = x^{ec}$. $(x^{a(ce-1)/a})^{-1} \bmod n = x^{ce}.(x^{ce-1})^{-1} \bmod n = x$. Như vậy, trong trường hợp này việc truyền tin bảo mật không còn an toàn nữa. Vì vậy, ta cần nhớ khi dùng các hệ RSA để tổ chức mạng truyền tin bảo mật, cần tránh dùng môđuyn n chung cho các người tham gia khác nhau!

2. *Dùng số mũ lập mã e bé.* Để cho việc tính toán hàm lập mã được hiệu quả, ta dễ có xu hướng chọn số mũ e của hàm lập mã là một số nguyên bé, chẳng hạn e =3. Tuy nhiên, nếu trong một mạng truyền tin bảo mật dùng các hệ mật mã RSA, nếu có nhiều người cùng chọn số mũ lập mã e bé giống nhau thì sẽ có nguy cơ bị tấn công bởi việc thám mã như sau : Giả sử có ba người tham gia chọn ba khoá công khai là (n_1, e), (n_2, e), (n_3, e) với cùng số mũ e =3. Một người tham gia A muốn gửi một thông báo x cho cả ba người đó, và để bảo mật, gửi bản mã $c_i = x^3 \bmod n_i$ cho người thứ i. Ba môđuyn n_i là khác nhau, và có phần chắc là từng cặp nguyên tố với nhau. Một người thám mã có thể dùng định lý số dư Trung quốc để tìm một số m ($0 \le m \le n_1 n_2 n_3$) thoả mãn

$$\begin{cases} m \equiv c_1 \bmod n_1 \\ m \equiv c_2 \bmod n_2 \\ m \equiv c_3 \bmod n_3 \end{cases}$$

Vì $x \le n_i$, nên $x^3 \le n_1 n_2 n_3$, do đó ắt có $m = x^3$. Vậy là ta đã đưa được bài toán tìm căn bậc ba theo nghĩa đồng dư modn_i về bài toán tìm căn bậc ba theo nghĩa số học thông thường: tìm căn bậc ba của m ta được x, tức được bản rõ!

Với những lý do khác, người ta đã có những bằng chứng để chứng tỏ rằng hệ RSA cũng không bảo đảm an toàn nếu ta dùng các khoá có số mũ giải mã d là số nguyên bé, dù rằng khi đó thuật toán giải mã có làm việc hiệu quả hơn. Vì thế, khi sử dụng các hệ mật mã RSA, để bảo đảm an toàn ta nên chọn các số mũ e và d là những số nguyên lớn, có kích cỡ lớn gần như bản thân số n.

3. *Lợi dụng tính nhân của hàm lập mã*. Ta chú ý rằng hàm lập mã $f(x) = x^e \bmod n$ có tính nhân (multiplicative property), nghĩa là $f(x.y) = f(x).f(y)$. Dựa vào tính chất đó, ta thấy rằng nếu

c là mật mã của bản rõ x, thì $\overline{c} = c.u^e \bmod n$ sẽ là mật mã của bản rõ xu. Do đó, khi lấy được bản mật mã c , để phát hiện bản rõ x người thám mã có thể chọn ngẫu nhiên một số u rồi tạo ra bản mã $\overline{c}$,và nếu người thám mã có khả năng thám mã theo kiểu ô có bản mã được chọn ầ (xem 1.5.1), tức có khả năng với $\overline{c}$ được chọn tìm ra bản rõ tương ứng là $\overline{x} = xu$,thì bản rõ gốc cần phát hiện sẽ là $x = \overline{x}.u^{-1} \bmod n$. Tất nhiên, khả năng người thám mã có năng lực giải quyết bài toán thám mã theo kiểu có bản mã được chọn là rất hiếm, nhưng dầu sao đấy cũng là một trường hợp mà vấn đề bảo mật dễ bị tấn công, ta không thể không tính đến để tìm cách tránh!

4. *Tấn công bằng cách lặp phép mã*. Ta cũng chú ý rằng hàm lập mã $f(x) = x^e \bmod n$ là một phép hoán vị trên tập $Z_n = \{0,1,...,n-1\}$, do đó với mọi $c \in Z_n$ nếu ta thực hiện lặp phép lập mã để được

$$c_0 = c, c_1 = c^e \bmod n, c_2 = c^{e^2} \bmod n, ..., c_i = c^{e^i} \bmod n, ...$$

ắt sẽ tìm được số $k \geq 1$ sao cho $c_k = c^{e^k} \bmod n = c$. Nếu c là bản mã của một bản rõ x nào đó, $c = x^e \bmod n$, thì người thám mã có thể xuất phát từ c thực hiện lặp phép lập mã như trên sẽ tìm được số $k \geq 1$ bé nhất sao cho $c_k = c$. Và khi đó ta sẽ có số hạng trước đó $c_{k-1} = x$, là bản rõ cần phát hiện. Thuật toán về hình thức là khá đơn giản, nhưng hiệu quả thực hiện không đáng hy vọng lắm, vì số phép lặp cần thực hiện nói chung có thể là rất lớn, cỡ bằng số các phép hoán vị trên Z_n , tức là bằng n !, với số n có khoảng 200 chữ số thập phân. Trên thực tế, phỏng theo thuật toán nói trên ta có thể dễ dàng có một thuật toán phân tích n thành thừa số nguyên tố, mà một thuật toán như vậy làm việc có hiệu quả thiết thực, như đã trình bày trong một phần trên, là chưa có! Vì vậy, nguy cơ bị thám mã bằng thuật toán đơn giản nói trên đối với tính an toàn của hệ mật mã RSA là không đáng ngại lắm.

5. *Về khả năng che giấu của bản mật mã*. Mật mã, sở dĩ nó giữ được bí mật, là do khả năng che giấu thông tin của nó, tức là biết bản mã y khó lòng tìm được thông tin nào để phát hiện ra bản rõ x. Một cách thô thiển, ta nói bản rõ x là *không che giấu được* qua phép lập mật mã RSA $e_K(x) = x^e \bmod n$, nếu $e_K(x) = x$. Nói cách khác, x là không che giấu được nếu bản mã của x cũng chính là x. Tiếc rằng với bất kỳ hệ mật mã RSA nào cũng có những bản rõ không che giấu được, đó là những bản rõ $x = -1, 0, 1 \bmod n$ (vì số mũ e luôn luôn là số lẻ). Người ta chứng minh được rằng nếu $n = p.q$, thì số các bản rõ $x \in Z_n$ không che giấu được là bằng

$$(1+\gcd(e-1, p-1)).(1+\gcd(e-1, q-1)).$$

Vì $e-1$, $p-1$, $q-1$ là các số chẵn, nên số đó ít nhất là 9, nên mỗi hệ RSA có ít nhất 9 bản rõ không che giấu được. Tuy nhiên, thường n, và do đó cả p và q, đều rất lớn, nên tỷ lệ các bản rõ không che giấu được nói chung là bé không đáng kể, và do đó khả năng gặp các bản rõ không che giấu được không tạo nên một nguy cơ đáng kể nào đối với việc dùng các hệ mật mã RSA.

3. Hệ mật mã khoá công khai Rabin.

3.1. Mô tả hệ mật mã Rabin.

Sơ đồ hệ mật mã khoá công khai Rabin được cho bởi
$S = (P, C, K, E, D)$,

trong đó: P =C = Z_n , trong đó n là một số nguyên Blum, n =$p.q$, với p và q là hai số nguyên tố có tính chất $p \equiv 3(\text{mod}4)$, $q \equiv 3(\text{mod}4)$,

K = {K = (K', K'') : K' =(n,B), K'' =(p,q), $0 \le B \le n-1$},

các thuật toán E và D được xác định bởi

$$E\,(K',x) = x\,(x+B) \bmod n\,,$$

$$D\,(K'',y) = \sqrt{\frac{B^2}{4}+y} - \frac{B}{2} \bmod n.$$

(ký hiệu căn bậc hai sẽ được giải thích sau).

Trong một mạng truyền tin bảo mật với sơ đồ mật mã Rabin, mỗi người tham gia chọn cho mình các yếu tố n,B,p,q để lập nên khoá công khai và khoá bí mật của mình.

Ta chú ý rằng với mỗi bộ khoá K, các thuật toán $e_{K'} = E\,(K',.)$ và $d_{K''} = D\,(K'',.)$ không lập thành một cặp song ánh, cụ thể là $e_{K'}$ không phải là một đơn ánh, vì nếu w là một căn bậc hai của 1 theo mod$n$ thì $e_{K'}(w(x+\frac{B}{2}) - \frac{B}{2}) = e_{K'}(x)$, mà ta có đến 4 căn bậc hai của 1 theo mod$n$,tức là ta có 4 giá trị khác nhau của đối số x cho cùng một giá trị $e_{K'}(x)$.

Bây giờ nói đến thuật toán giải mã $d_{K''} = D\,(K'',.)$. Đặt C = $B^2/4 + y$, ta có $d_{K''}(y) = \sqrt{C} - B/2 \bmod n$, do đó để có $d_{K''}(y)$, ta cần tính $\sqrt{C}$ mod$n$, tức cần giải phương trình $z^2 \equiv C \bmod n$. Phương trình đó tương đương với hệ thống gồm hai phương trình sau đây:

$$\begin{cases} z^2 \equiv C \bmod p, \\ z^2 \equiv C \bmod q. \end{cases} \qquad (2)$$

Vì p và q là các số nguyên tố nên ta có $C^{\frac{p-1}{2}} \equiv 1 \bmod p$, $C^{\frac{q-1}{2}} \equiv 1 \bmod q$. Theo giả thiết, $p \equiv 3(\text{mod}4)$ và $q \equiv 3(\text{mod}4)$, nên $\frac{p+1}{4}$ va` $\frac{q+1}{4}$ là các số nguyên; và ta có

$$(\pm C^{\frac{p+1}{4}})^2 \equiv C\,(\text{mod}\,p),\ (\pm C^{\frac{q+1}{4}})^2 \equiv C\,(\text{mod}\,q).$$

Do đó,phương trình $z^2 \equiv C$ mod$n$, hay hệ phương trình (2), có 4 nghiệm theo mod$n$, tương ứng với 4 hệ phương trình sau đây :

$$\begin{cases} z \equiv C^{(p+1)/4} (\text{mod}\,p) \\ z \equiv C^{(q+1)/4} \ (\text{mod}\,q) \end{cases} \quad \begin{cases} z \equiv C^{(p+1)/4} (\text{mod}\,p) \\ z \equiv -C^{(q+1)/4} (\text{mod}\,q) \end{cases}$$

$$\begin{cases} z \equiv -C^{(p+1)/4} (\text{mod}\,p) \\ z \equiv C^{(q+1)/4} \ (\text{mod}\,q) \end{cases} \quad \begin{cases} z \equiv -C^{(p+1)/4} (\text{mod}\,p) \\ z \equiv -C^{(q+1)/4} (\text{mod}\,q) \end{cases}$$

Cả 4 nghiệm của 4 hệ phương trình đó theo mod$n$ đều được viết chung dưới một ký hiệu là $\sqrt{C}$ mod$n$, và vì vậy thuật toán giải mã $d_{K''}(y)$ thực tế sẽ cho ta 4 giá trị khác nhau theo mod$n$ mà bản

rõ là một trong 4 giá trị đó. Việc chọn giá trị nào trong 4 giá trị tìm được làm bản rõ là tuỳ thuộc vào những đặc trưng khác của bản rõ mà người giải mã nhận biết (thí dụ bản rõ dưới dạng số phải có biểu diễn nhị phân là mã của một văn bản tiếng Anh thông thường).

Thí dụ : Giả sử n =77 = 7.11, B =9 (ở đây p =7, q =11). Ta có

$$e_{K'}(x) = x^2 + 9x \bmod 77,$$

$$d_{K''}(y) = \sqrt{1+y} - 43 \bmod 77,$$

vì 2^{-1}=39mod77, 9.2^{-1} =9.39 =43mod77, B^2=4mod77, $B^2/4$ =1mod 77.

Với x =44 ta có $e_{K'}(x) = 44^2+9.44$ =2332 =22mod77, bản mã tương ứng với x là y = 22. Bây giờ giải mã với bản mã y =22, bằng thủ tục nói trên ta có thể tìm được 4 giá trị của $\sqrt{1+y} = \sqrt{1+22} = \sqrt{23}$ theo mod77 là 10,67,32,45, từ đó 4 giá trị có thể có của $d_{K''}(y)$ là

$$d_{K''}(y) = 44, 24, 66, 2.$$

Bản rõ nằm trong 4 giá trị đó, trong trường hợp này là 44.

3.2. Tính an toàn của hệ mật mã Rabin.

Trong định nghĩa của hệ mật mã Rabin, khoá công khai là (n,B), khoá bí mật là (p,q) tức là cặp thừa số nguyên tố của n . Như vậy, tính an toàn của hệ mật mã nằm ở việc giữ bí mật các thừa số p và q. Định nghĩa của phép giải mã cũng cho ta thấy rằng yếu tố có ý nghĩa quyết định trong phép giải mã là việc tính căn bậc hai của một số theo mod$n$. Trong mục 4.1.2 bài toán tìm căn bậc hai theo mod$n$ (với n là hợp số Blum) đã được chứng tỏ là có độ khó tương đương với bài toán phâ n tích n thành thừa số nguyên tố. Vì vậy, bài toán giải mã đối với hệ mật mã Rabin, cũng là bài toán giữ bí mật khoá bí mật (p,q), và bài toán phân tích số nguyên thành thừa số nguyên tố là có độ khó tương đương nhau. Và đó cũng là yếu tố bảo đảm tính an toàn của hệ mật mã Rabin !

4. Hệ mật mã khoá công khai ElGamal.

4.1. Mô tả hệ mật mã ElGamal.

Hệ mật mã ElGamal được T. ElGamal đề xuất năm 1985, dựa vào độ phức tạp của bài toán tính lôgarit rời rạc, và sau đó đã nhanh chóng được sử dụng rộng rãi không những trong vấn đề bảo mật truyền tin mà còn trong các vấn đề xác nhận và chữ ký điện tử.

Sơ đồ hệ mật mã khoá công khai ElGamal được cho bởi

$$S = (P, C, K, E, D),$$

trong đó: $P = Z_p^*$, $C = Z_p^* \times Z_p^*$, với p là một số nguyên tố;

$K = \{K = (K', K'') : K' = (p, \alpha, \beta), K'' = a, \beta \equiv \alpha^a \bmod p\}$,

ở đây α là một phần tử nguyên thuỷ theo mod$p$, tức của Z_p^*.

Các thuật toán lập mã $e_{K'} = E(K',.)$ và giải mã $d_{K''} = D(K'',.)$ được xác định như sau: Với mỗi $x \in P = Z_p^*$, để lập mật mã cho x trước hết ta chọn thêm một số ngẫu nhiên $k \in Z_{p-1}$ rồi tính:

$$e_{K'}(x,k) = (y_1, y_2), \quad \text{với} \quad \begin{cases} y_1 = \alpha^k \bmod p, \\ y_2 = x.\beta^k \bmod p. \end{cases}$$

Với mọi số ngẫu nhiên k bất kỳ, ta đều xem $e_{K'}(x,k)$ là mật mã của x. Và thuật toán giải mã được xác định bởi

$$d_{K''}(y_1, y_2) = y_2.(y_1^a)^{-1} \bmod p.$$

Các phép lập mật mã và giải mã được xác định như vậy là hợp thức, vì ta có với mọi $x \in P = Z_p^*$ và mọi $k \in Z_{p-1}$:

$$d_{K''}(e_{K'}(x,k)) = x.\beta^k.(\alpha^{k.a})^{-1} \bmod p = x.\beta^k.\beta^{-k} \bmod p = x.$$

Ta chú ý rằng trong một mạng truyền thông bảo mật với việc dùng sơ đồ mật mã ElGamal, mỗi người tham gia tự chọn cho mình các tham số p, α, a, rồi tính β, sau đó lập và công bố khoá công khai $K' = (p, \alpha, \beta)$, nhưng phải giữ tuyệt mật khoá bí mật $K'' = a$. Bài toán biết khoá công khai tìm ra khoá bí mật chính là bài toán tính lôgarit rời rạc được kể đến trong mục 4.1.2, một bài toán khó cho đến nay chưa có một thuật toán nào làm việc trong thời gian đa thức giải được nó.

Thí dụ : Chọn $p = 2579$, $\alpha = 2$, $a = 765$, ta tính được $\beta = 2^{765} = 949 \bmod 2579$. Ta có khoá công khai (2579, 2, 949) và khoá bí mật 765. Giả sử để lập mật mã cho $x = 1299$, ta chọn ngẫu nhiên $k = 853$, sẽ có

$$e_{K'}(1299, 853) = (2^{853}, 1299.\ 949^{853}) \bmod 2579$$
$$= (453, 2396).$$

Và giải mã ta được lại

$$d_{K''}(453, 2396) = 2396.\ (453^{765})^{-1} \bmod 2579 = 1299.$$

4.2. Tính an toàn của hệ mật mã ElGamal.

Như đã trình bày ở trên, nếu ta xem tính an toàn của hệ mật mã ElGamal là ở việc giữ tuyệt mật khoá bí mật K'', thì ta có thể yên tâm vì bài toán phát hiện khoá bí mật có độ khó tương đương với bài toán tính lôgarit rời rạc, mà bài toán này thì như ở các mục 4.1.2 và 2.4.3 đã chứng tỏ, cho đến nay chưa có một thuật toán nào làm việc trong thời gian đa thức giải được nó. Có một điều cảnh báo là nên chú ý chọn môđuyn p là số nguyên tố sao cho $p - 1$ có ít nhất một ước số nguyên tố lớn (xem 2.4.3). Điều đó là thực hiện được nếu số nguuyên tố p được chọn là số nguyên tố Sophie Germain (tức có dạng $2q + 1$, với q cũng là số nguyên tố lớn).

Ngoài ra, còn có khả năng khoá bí mật $K'' = a$ bị lộ do cẩu thả trong việc sử dụng số ngẫu nhiên k, đặc biệt là khi *để lộ số k được dùng*. Thực vậy, nếu để lộ số k, thì khoá bí mật a được tính ra ngay theo công thức sau đây:

$$a = (x - ky_2)y_1^{-1} \bmod (p-1).$$

Như vậy,một người thám mã có khả năng tấn công theo kiểu "biết cả bản rõ" (xem 1.5.1) có thể phát hiện ra khoá a nếu biết k.

Một trường hợp khác làm mất tính an toàn của hệ mật mã ElGamal là việc *dùng cùng một số k cho nhiều lần lập mật mã*. Thực vậy, giả sử dùng cùng một số ngẫu nhiên k cho hai lần lập mã, một lần cho x_1, một lần cho x_2, và được các bản mã tương ứng (y_1, y_2) và (z_1, z_2). Vì cùng dùng một số k nên $y_1 = z_1$. Và do đó theo công thức lập mã ta có $z_2/y_2 = x_2/x_1$, tức là $x_2 = x_1.z_2/y_2$. Như vậy, một người thám mã, một lần "biết cả bản rõ" dễ dàng phát hiện được bản rõ trong các lần sau.

4.3. Các hệ mật mã tương tự ElGamal.

Hệ mật mã ElGamal được xây dựng dựa trên các yếu tố : một nhóm hữu hạn cyclic (Z_p^*), một phần tử nguyên thuỷ ($\alpha \in Z_p^*$) sao cho bài toán tính lôgarit rời rạc (tính $a = \log_\alpha \beta$, tức cho β tìm a sao cho $\beta = \alpha^a \bmod p$) là rất khó thực hiện. Vì vậy, nếu có đủ các yếu tố đó thì ta có thể xây dựng các hệ mật mã tương tự ElGamal. Như vậy, sơ đồ của một hệ mật mã tương tự ElGamal được cho bởi

$$S = (P, C, K, E, D),$$

trong đó: $P = G$, $C = G \times G$, với G là một nhóm cyclic hữu hạn;

$K = \{K = (K', K'') : K' = (G, \alpha, \beta), K'' = a, \beta = \alpha^a\}$,

ở đây α là một phần tử nguyên thuỷ của nhóm G.

Các thuật toán lập mã $e_{K'} = E(K',.)$ và giải mã $d_{K''} = D(K'',.)$ được xác định như sau: Với mỗi $x \in P = G$, để lập mật mã cho x trước hết ta chọn thêm một số ngẫu nhiên k $(0 \le k \le |G|)$ rồi tính:

$$e_{K'}(x,k) = (y_1, y_2), \quad \text{với} \quad \begin{cases} y_1 = \alpha^k \\ y_2 = x.\beta^k \end{cases}$$

Với mọi số ngẫu nhiên k bất kỳ, ta đều xem $e_{K'}(x,k)$ là mật mã của x. Và thuật toán giải mã được xác định bởi

$$d_{K''}(y_1, y_2) = y_2.(y_1^a)^{-1} \bmod p.$$

Phép nhân trong các biểu thức nói trên đều là phép nhân của G.

Có hai lớp nhóm thường được sử dụng để xây dựng các hệ mật mã tương tự ElGamal là nhóm nhân của trường Galois GF(p^n) và nhóm cộng của một đường cong elliptic xác định trên một trường hữu hạn.

1. <u>*Nhóm nhân của trường Galois GF(p^n)*</u> : Trường Galois GF(p^n) là trường của các đa thức với hệ số trong Z_p lấy theo môđuyn là một đa thức bậc n bất khả qui; với phép cộng và phép nhân là phép cộng và phép nhân đa thức theo môđuyn đó. Trường có p^n phần tử, có thể xem mỗi phần tử là một đa thức bậc n -1 với hệ số thuộc $Z_p = \{0,1,2,...,p-1\}$, thậm chí là một vectơ n chiều mà các thành phần là các hệ số của đa thức đó. Tập tất cả các đa thức khác 0 lập thành nhóm nhân của trường $GF(p^n)$,và người ta chứng minh được rằng nhóm nhân đó là cyclic.

Như vậy, nhóm $G = GF(p^n) \setminus \{0\}$ là nhóm cyclic cấp p^n-1. ta có thể chọn một phần tử nguyên thuỷ của nhóm đó, và thiết lập bài toán lôgarit rời rạc tương ứng, từ đó xây dựng được hệ mật mã tương tự ElGamal.

2. <u>*Nhóm cộng của đường cong elliptic*</u> : Giả sử p là một số nguyên tố > 3. *Đường cong elliptic* $y^2 = x^3 + a.x + b$ *trên* Z_p **,** trong đó $a,b \in Z_p$ là các hằng số thoả mãn $4a^3 + 27b^2 \not\equiv 0 \pmod p$, được định nghĩa là tập hợp tất cả các điểm $(x,y) \in Z_p \times Z_p$ thoả mãn phương trình

$$y^2 \equiv x^3 + a.x + b \pmod p,$$

cùng với một phần tử đặc biệt mà ta ký hiệu là $\mathcal{O}$. Tập hợp đó được ký hiệu là E. Trên tập E ta xác định một phép cộng như sau : Giả sử $P = (x_1, y_1)$ và $Q = (x_2, y_2)$ là hai điểm của E. Nếu $x_1 = x_2$ và $y_1 = -y_2$ thì ta định nghĩa $P + Q = \mathcal{O}$; nếu không thì $P + Q = (x_3, y_3)$, trong đó

$$x_3 = \lambda^2 - x_1 - x_2 , \quad y_3 = \lambda (x_1 - x_3) - y_1 ,$$

với

$$\lambda = \begin{cases} (y_2 - y_1)/(x_2 - x_1), khi\ P \neq Q; \\ (3x_1^2 + a)/2y_1, \qquad khi\ P = Q. \end{cases}$$

Ngoài ra, ta định nghĩa thêm : $P + \mathcal{O} = \mathcal{O} + P = P$.

Tập E với phép toán cộng đó lập thành một nhóm. Nếu $|E| = q$ là số nguyên tố thì nhóm cộng đó là nhóm cyclic, và mọi phần tử khác không ($\neq \mathcal{O}$) đều là phần tử nguyên thuỷ. Ta nhớ rằng trong trường hợp này, phần tử nghịch đảo là phần tử đối, phép nâng lên luỹ thừa n là phép nhân với số n, phép lôgarit tương ứng với một kiểu phép chia. Ta có thể xuất phát từ nhóm E này để xây dựng hệ mật mã tương tự ElGamal.

5. Các hệ mật mã dựa trên các bài toán $\mathcal{NP}$-đầy đủ.

5.1. Nguyên tắc chung.

Như đã giới thiệu trong chương II, các bài toán $\mathcal{NP}$-đầy đủ là các bài toán mà cho đến nay chưa tìm được một thuật toán với độ phức tạp tính toán đa thức nào để giải chúng. Và tính ô khóả của các bài toán đó lại được bảo đảm bằng sự kiện là chỉ cần có một thuật toán với độ phức tạp đa thức giải một bài toán $\mathcal{NP}$-đầy đủ nào đó thì lập tức mọi bài toán $\mathcal{NP}$-đầy đủ đều giải được trong thời gian đa thức.

Đối với một số bài toán $\mathcal{NP}$-đầy đủ, tuy không có thuật toán với độ phức tạp đa thức để giải đối với mọi dữ liệu của bài toán, nhưng có thể có một lớp các dữ liệu mà đối với chúng có thuật toán để giải với thời gian chấp nhận được. Với những bài toán như vậy ta có thể sử dụng để xây dựng các hệ mật mã khoá công khai với nguyên tắc chung như sau : Hệ mật mã sẽ có phép giải mã tương đương với việc tìm lời giải cho bài toán $\mathcal{NP}$-đầy đủ đó; tuy nhiên có một thủ tục để biến một dữ liệu nói chung của bài toán $\mathcal{NP}$-đầy đủ đó thành một dữ liệu thuộc lớp đặc biệt mà đối với nó có thể giải được bởi một thuật toán với độ phức tạp thời gian chấp nhận được. Như vậy, ta đã biến được phép lập mã thành một hàm ô cửa sập một phía ằ, và đó là cơ sở để xây dựng hệ mật mã khoá công khai tương ứng.

Ta sẽ xét sau đây hai trường hợp xây dựng được các hệ mật mã khoá công khai theo cách như vậy : một là hệ mật mã Merkle-Hellman dựa trên bài toán sắp ba lô (hay bài toán tổng tập con), và hai là hệ mật mã Mc-Eliece dựa trên bài toán giải mã tuyến tính tự sửa sai.

5.2. Hệ mật mã Merkle-Hellman.

Bài toán sắp ba lô (tức bài toán KNAPSACK, cũng được gọi là bài toán tổng tập con) được đặt ra như sau: Cho một tập các số nguyên dương $\{a_1, a_2, ..., a_n\}$ và một số nguyên dương s. Hãy xác định xem có hay không một tập con các a_j mà tổng của chúng bằng s. Một cách tương đương, hãy xác định xem có hay không các $x_i \in \{0,1\}$ $(1 \leq i \leq n)$ sao cho $\sum_{i=1}^{n} a_i x_i = s$.

Bài toán này là $\mathcal{NP}$-đầy đủ, tuy nhiên nếu ta hạn chế bài toán trên các dữ liệu I =($\{a_1, a_2, ..., a_n\}, T$), trong đó $\{a_1, a_2, ..., a_n\}$ là dãy *siêu tăng*, tức là dãy thoả mãn điều kiện

$$\forall j = 2,3,...,n:\ a_j > \sum_{i=1}^{j-1} a_i,$$

thì việc tìm trả lời là khá dễ dàng, chẳng hạn có thể bằng thuật toán đơn giản dưới đây:

1. **for** $i = n$ **downto** 1 **do**
   if $T > a_i$ **then** $T = T - a_i$, $x_i = 1$, **else** $x_i = 0$
2. **if** $\sum_{i=1}^{n} x_i.a_i = T$ **then** $X = (x_1,...,x_n)$ is the solution of problem,
   else there is no solution.

Bây giờ, để chuẩn bị xây dựng một sơ đồ mật mã Merkle-Hellman, ta chọn trước một số nguyên dương n và một số nguyên tố p đủ lớn. Với mỗi người tham gia sẽ được chọn một bộ khoá $K = (K', K'')$,

trong đó khoá bí mật $K'' = (A, p, a)$ gồm một dãy siêu tăng $A = \{a_1, a_2, ..., a_n\}$ thoả mãn $\sum_{i=1}^{n} a_i < p$,

và một số a, $1 < a < p$; khoá công khai $K' = \{b_1,...,b_n\}$ với $b_i = a.a_i \bmod p$.

Sơ đồ hệ mật mã Merkle-Hellman được định nghĩa bởi

$$S = (P, C, K, E, D),$$

trong đó $P = \{0,1\}^n$, $C = \{0,1,...,n(p-1)\}$, K là tập các bộ khoá $K = (K', K'')$ như được xây dựng ở trên. Các thuật toán lập mật mã và giải mã được xác định bởi:

Với mọi $x = (x_1,...,x_n) \in P$ thuật toán lập mã cho ta

$$E(K', x) = \sum_{i=1}^{n} x_i.b_i \ ;$$

và với mọi $y \in C$, ta tính $z = a^{-1}.y \bmod p$, rồi sau đó giải bài toán sắp balô đối với dữ liệu $I = (\{a_1, a_2, ..., a_n\}, z)$ ta sẽ được lời giải $(x_1,...,x_n)$, lời giải đó là giá trị của $D(K'', y)$.

Thí dụ: Chọn $n = 6$, khoá bí mật có $p = 737$, $A = \{12, 17, 33, 74, 157, 316\}$, $a = 635$. Tính được khoá công khai là {250, 477, 319, 559, 200, 196}. Với bản rõ $x = 101101$ ta có bản mã tương ứng là $y = 1324$. Để giải mã, trước hết tính $z = a^{-1}.y \bmod p = 635^{-1}.1324 \bmod 737 = 435$, sau đó giải bài toán sắp balô với dãy siêu tăng A và z ta được

$$435 = 12 + 33 + 74 + 316,$$

tức được lời giải $x = (1,0,1,1,0,1)$.

Hệ mật mã Merkle-Hellman được đề xuất khá sớm, từ năm 1978, đến năm 1985 Shamir tìm được một phương pháp thám mã trong thời gian đa thức dựa vào một thuật toán của Lenstra giải bài toán qui hoạch động. Tuy nhiên, sau đó, vào năm 1988, Chor và Rivest có đưa ra một cách khác xây dựng hệ mật mã cũng dựa vào bài toán sắp balô, cho đến nay vẫn giữ được an toàn.

5.3. Hệ mật mã McEliece.

Hệ mật mã McEliece được xây dựng dựa vào tính $\mathcal{NP}$-đầy đủ của bài toán giải mã tuyến tính tự sửa sai (trong lý thuyết truyền tin). Bài toán được đặt ra như sau: giả sử nguồn tin là tập các từ k bit nhị phân, tức tập hợp $\{0,1\}^k$, được truyền đi trên một kênh có nhiễu, tức là nếu truyền trực tiếp các dãy từ k bit thì thông tin mà ta nhận được có thể bị sai lệch và ta không nhận được đúng thông tin được truyền đi. Để khắc phục những sai lệch đó người ta tìm cách mã hoá nguồn tin gốc bằng cách thêm cho mỗi từ k bit mang thông tin một số bit dùng để tự hiệu chỉnh, tức là thực hiện

một phép mã hoá biến mỗi từ k bit ban đầu thành một từ n bit, với $n > k$, được gọi là từ mã. Phép mã hoá tuyến tính là phép mã hoá được thực hiện bằng cách nhân từ k bit ban đầu x với một ma trận G cấp $k \times n$ để được từ mã n bit y, $y = x.G$ (các phép toán cộng và nhân được thực hiện theo mod2). Ta định nghĩa khoảng cách Hamming giữa hai từ mã n bit là số các vị trí mà tại đó hai từ mã có giá trị khác nhau; khoảng cách d của hệ mã là khoảng cách Hamming bé nhất giữa hai từ mã bất kỳ. Như vậy, một hệ mã tuyến tính được xác định bởi một ma trận G (gọi là ma trận sinh), và được đặc trưng bởi ba số $[n,k,d]$. Nếu $d = 2t + 1$, thì hệ mã có khả năng tự sửa sai đến t sai ngẫu nhiên nhiễm phải do nhiễu của kênh truyền. Tuy nhiên, việc tự sửa sai (tức là khi nhận được từ mã có thể có đến t sai ta tìm lại được đúng từ k bit thông tin ban đầu) của các hệ mã tuyến tính như vậy nói chung khá phức tạp, và bài toán giải mã tuyến tính tự sửa sai đã được chứng minh là một bài toán $\mathcal{NP}$-khó, tức cho đến nay chưa biết có thuật toán nào làm việc trong thời gian đa thức giải được nó. Mặc dầu vậy, người ta đã tìm được một số lớp riêng các hệ mã tuyến tính mà đối với chúng có thể xây dựng được những thuật toán giải mã tự sửa sai làm việc có hiệu quả, các hệ mã Goppa là một lớp như vậy. Hệ mã Goppa là một loại hệ mã tuyến tính có các đặc trưng $n = 2^m$, $d = 2t + 1$, $k = n - mt$, có ma trận sinh G cấp $k \times n$ được xây dựng dựa trên một số tính chất đại số của trường $GF(2^n)$-mà ở đây ta không đi vào các chi tiết.

Để có một hệ mật mã McEliece, trước hết ta chọn một hệ mã Goppa với ma trận sinh G và các đặc trưng trên, sau đó dùng một ma trận S khả nghịch cấp $k \times k$ trên Z_2 và một ma trận hoán vị P cấp $n \times n$ (cũng có các phần tử trong Z_2) để biến hệ mã Goppa với ma trận sinh G thành một hệ mã tuyến tính "phổ biến" với ma trận sinh $G^* = SGP$; vậy là đã biến hệ mã Goppa có thuật toán giải mã hiệu quả thành một hệ mã tuyến tính nói chung mà ta chỉ biết việc giải mã tự sửa sai đối với nó là $\mathcal{NP}$-khó. Hệ mật mã mà ta xây dựng sẽ có thuật toán giải mã là "dễ" đối với người trong cuộc như giải mã Goppa, và là "khó" đối với người ngoài như giải mã tuyến tính nói chung!

Như vậy, một *hệ mật mã khoá công khai McEliece* được xác định bởi

$$S = (P, C, K, E, D),$$

trong đó $P = \{0,1\}^k$, $C = \{0,1\}^n$, K là tập hợp các bộ khoá $K = (K', K'')$, với khoá bí mật $K'' = (G,S,P)$ gồm một ma trận sinh G của một hệ mã Goppa, một ma trận khả nghịch S cấp $k \times k$ trên Z_2 và một ma trận hoán vị P cấp $n \times n$; khoá công khai $K' = G^*$ là ma trận "đã được biến đổi" nói trên.

Thuật toán lập mật mã $E(K',.): P \to C$ được xác định bởi

$$E(K', x) = x.G^* + \boldsymbol{e},$$

trong đó $\boldsymbol{e} \in \{0,1\}^n$ là một vectơ ngẫu nhiên có trọng số t, tức có t thành phần là 1. Thuật toán giải mã $D(K'',.)$ được thực hiện theo ba bước như sau với mọi $y \in C = \{0,1\}^n$:

1. Tính $y_1 = y.P^{-1}$,
2. Giải mã Goppa đối với y_1, giả sử được x_1.
3. Tính $D(K'', y) = x_1.S^{-1}$.

Dễ thử lại rằng các thuật toán lập mật mã và giải mã xác định như trên là hợp thức, vì với mọi $x \in P = \{0,1\}^k$, ta đều có

$$D(K'', E(K', x)) = x,$$

Đẳng thức đó đúng với mọi vectơ $\boldsymbol{e}$ bất kỳ có trọng số $\le t$. Hệ mật mã này cũng tương tự như hệ mật mã ElGamal ở chỗ khi lập mật mã ta có thể chọn thêm cho dữ liệu vào một yếu tố ngẫu nhiên; về sau ta sẽ gọi những hệ mật mã như vậy là hệ mật mã xác suất.

Yếu tố chủ yếu bảo đảm tính an toàn của các hệ mật mã McEliece là ở chỗ từ khoá công khai G^* khó phát hiện ra khoá bí mật (G,S,P) và ở tính $\mathcal{NP}$-khó của bài toán giải mã tuyến tính tự sửa sai nói chung. Cũng cần nhớ rằng độ an toàn còn phụ thuộc vào việc chọn các tham số k,n,t đủ lớn; theo gợi ý của các nghiên cứu thực nghiệm thì đủ lớn có nghĩa là $n \approx 1024$, $k \approx 644$, $t \approx 38$. Với những đòi hỏi đó thì kích cỡ của các ma trận G, S, P và G^* sẽ quá lớn, khá bất tiện cho việc thực thi trong thực tế, vì vậy mà các hệ mật mã McEliece chưa được sử dụng phổ biến lắm.

6. Các hệ mật mã xác suất khoá công khai.

6.1. Đặt vấn đề và định nghĩa.

Mật mã xác suất là một ý tưởng được đề xuất bởi Goldwasser và Micali từ năm 1984, xuất phát từ yêu cầu giải quyết một vấn đề sau đây: Giả thiết ta có một hệ mật mã khoá công khai, và ta muốn lập mật mã cho bản rõ chỉ gồm một bit. Điều đó thường gặp khi ta muốn bí mật truyền đi một thông tin chỉ có nội dung là *có* hoặc *không*, tức là một thông tin đặc biệt quan trọng nhưng chỉ gồm một bit. Nếu ta dùng một hệ mật mã khoá công khai thông thường, thì bản mật mã được truyền đi sẽ là $e_{K'}(0)$ hoặc $e_{K'}(1)$, một người thám mã có thể không biết cách giải mã, nhưng lại hoàn toàn có thể tính trước các giá trị $e_{K'}(0)$ và $e_{K'}(1)$, và khi lấy được bản mã truyền đi trên kênh truyền tin công cộng, chỉ cần so sánh bản mã nhận được đó với hai bản $e_{K'}(0)$ và $e_{K'}(1)$ đã được tính sẵn là đủ biết được thông tin mật được truyền đi là 0 hay là 1. Các hệ mật mã khoá công khai sở dĩ có được tính bảo mật là vì từ thông tin về bản mã khó lòng khai thác được thông tin gì về bản rõ, nhưng rõ ràng điều đó không còn được bảo đảm nếu số các bản rõ là rất ít, chẳng hạn như khi các bản rõ có độ dài cực ngắn, hay như trường hợp trên, số các bản rõ chỉ là hai, cụ thể là 0 và 1.

Mục đích của việc xây dựng mật mã xác suất là để bảo đảm không một thông tin nào về bản rõ có thể khai thác được (trong thời gian đa thức) từ bản mã; điều này, đối với các hệ mật mã khoá công khai, có thể được thực hiện bằng cách tạo cho một bản rõ nhiều bản mã khác nhau thu được một cách ngẫu nhiên với việc sử dụng các số ngẫu nhiên trong tiến trình lập mã. Sau đây là định nghĩa về một hệ mật mã xác suất khoá công khai:

Định nghĩa. Một *hệ mật mã xác suất khoá công khai* được xác định bởi một bộ

$$S = (P, C, K, E, D, R),$$

trong đó P, C, K được hiểu như đối với các hệ mật mã khoá công khai thông thường, R là một tập các phần tử ngẫu nhiên, và với mỗi $K = (K', K'') \in K$, thuật toán lập mật mã $e_{K'} = E(K',.): P \times R \to C$ và giải mã $d_{K''} = D(K'',.): C \to P$ thoả mãn đẳng thức:

với mọi $x \in P$, $r \in R$, $d_{K''}(e_{K'}(x,r)) = x$.

Ngoài ra, ta mong muốn một *điều kiện an toàn* như trong định nghĩa sau đây được thoả mãn: ta ký hiệu $p_{K,x}$ là phân bố xác suất trên tập C, trong đó $p_{K,x}(y)$ là xác suất của việc y là bản mã khi biết K là khoá và x là bản rõ (xác suất được tính cho tất cả $r \in R$). Ta nói hai phân bố xác suất p_1 và p_2 trên C là ε-*phân biệt được* nếu có một thuật toán ε-phân biệt hai phân bố xác suất đó, tức là một thuật toán $A: C \to \{0,1\}$ thoả mãn tính chất

$$|E_A(p_1) - E_A(p_2)| \geq \varepsilon,$$

trong đó

$$E_A(p_i) = \sum_{y \in \mathcal{C}} p_i(y).p(A(y) = 1).$$

Bây giờ *điều kiện an toàn* được phát biểu như sau: Hệ mật mã xác suất khoá công khai S là an toàn nếu có ε>0 sao cho với mọi $K \in K$ và mọi $x \neq x'$, các phân bố xác suất $p_{K,x}$ và $p_{K,x'}$ là không ε-phân biệt được.

6.2. Hệ mật mã xác suất Goldwasser-Micali.

Sau đây là mô tả sơ đồ của hệ mật mã xác suất khoá công khai trên tập văn bản một bit do Goldwasser và Micali đề xuất năm 1984. Một hệ như vậy được cho bởi một danh sách

$$S = (P, C, K, E, D, R),$$

trong đó $P = \{0,1\}$, $C = R = Z_n^*$, $n = p.q$ là tích của hai số nguyên tố lớn, K là tập hợp các bộ khoá $K = (K', K'')$, trong đó khoá công khai $K' = (n, m)$ với $m \in Q_n = J_n - Q_n$ là một giả thặng dư bậc hai mod$n$, và khoá bí mật $K'' = (p,q)$. Các thuật toán lập mật mã và giải mã được xác định bởi

$$e_{K'}(x,r) = m^x.r^2 \bmod n,$$

$$d_{K''}(y) = \begin{cases} 0, \text{ khi } y \in Q_n \\ 1, \text{ khi } y \in \bar{Q}_n \end{cases}$$

với mọi $x \in P$, $r \in R$, $y \in C$.

Hệ mật mã Goldwasser-Micali lập mật mã cho bản rõ một bit: mật mã của bit 0 luôn luôn là một thặng dư bậc hai mod$n$, và mật mã của bit 1 là một giả thặng dư bậc hai mod$n$. Việc giải mã là khá dễ dàng khi ta biết khoá bí mật $K'' = (p,q)$. Thực vậy, với mọi $y \in Q_n \cup \bar{Q}_n$ ta có $\left(\frac{y}{n}\right) = 1$. Vì biết $K'' = (p,q)$, nên ta tính được

$$\left(\frac{y}{p}\right) = y^{\frac{p-1}{2}} \bmod p,$$

và do đó dễ thử được $y \in Q_n \Leftrightarrow \left(\frac{y}{p}\right) = 1$, và tính được $d_{K''}(y)$.

6.3.Hệ mật mã xác suất Blum-Goldwasser.

Hệ mật mã xác suất khoá công khai Blum-Goldwasser được xây dựng trên nền của các hệ mật mã theo dòng với dòng khoá là dãy số giả ngẫu nhiên Blum-Blum-Shub (xem 3.3.3), yếu tố ngẫu nhiên $r \in R$ ở đây sẽ được sử dụng như mầm sinh ra dãy số giả ngẫu nhiên của dòng khoá đó. Sơ đồ của *hệ mật mã xác suất khoá công khai Blum-Goldwasser* được cho bởi danh sách

$$S = (P, C, K, E, D, R),$$

trong đó $P = Z_2^*$, $C = Z_2^* \times Z_n$, $R = Q_n$, $n = p.q$ là tích của hai số nguyên tố lớn với $p \equiv q \equiv 3 \bmod 4$; K là tập hợp các bộ khoá $K = (K', K'')$, trong đó khoá công khai $K' = n$, và khoá bí mật $K'' = (p,q)$.

Thuật toán lập mã $e_{K'} = E(K',.) : P \times R \to C$ được tính theo các bước sau:

1. Cho $x = (x_1,...,x_l) \in P$ và $r \in R$. Từ mầm r theo thuật toán Blum-Blum-Shub tính dãy số $(s_0, s_1,..., s_{l+1})$ theo công thức

$$\begin{cases} s_0 = r, \\ s_{i+1} = s_i^2 \bmod n, \end{cases}$$

sau đó tính dãy số giả ngẫu nhiên $(z_1,...,z_l)$ bởi $z_i = s_i \bmod 2$.

2.Tính $y = (y_1,...,y_l)$ với $y_i = x_i + z_i \bmod 2$ $(1 \le i \le l)$.

3. Bản mã là $e_{K'}(x, r) = (y, s_{l+1}) = (y_1,...,y_l; s_{l+1})$.

Thuật toán giải mã $d_{K''} = D(K'',.): C \to P$ được thực hiện theo các bước sau đây sau khi nhận được bản mã $(y_1,...,y_l; s_{l+1})$:

1. Tính

$$a_1 = ((p+1)/4)^{l+1} \bmod (p-1),$$

$$a_2 = ((q+1)/4)^{l+1} \bmod (q-1).$$

2. Tính $b_1 = s_{l+1}^{a_1} \bmod p,\ b_2 = s_{l+1}^{a_2} \bmod q.$

3. Tìm $s_0 = r$ bằng cách giải hệ phương trình

$$\begin{cases} s_0 \equiv b_1 \bmod p \\ s_0 \equiv b_2 \bmod q \end{cases}$$

4. Với s_0 theo thuật toán BBS ta tìm lại được dãy bit $(z_1,...,z_l)$.

5. Cuối cùng ta được

$d_{K''}(y_1,...,y_l; s_{l+1}) = (x_1,...,x_l)$, với $x_i = y_i + z_i \bmod 2$ $(1 \le i \le l)$.

Như vậy là hệ mật mã Blum-Goldwasser đã được định nghĩa đầy đủ. Ta chú ý rằng nếu bản rõ x gồm l bit thì trong bản mã tương ứng, ngoài các bit mã $y_1,...,y_l$ ta phải gửi thêm số s_{l+1}, số đó được sử dụng trong các bước 1-3 của thuật toán giải mã để tìm lại mầm s_0 cần thiết cho việc tìm dòng khoá ngẫu nhiên $(z_1,...,z_l)$.

Ta chứng minh rằng số s_0 tính được theo thuật toán giải mã đúng là mầm s_0 mà ta cần tìm. Thực vậy, theo định nghĩa, ta có với mọi $i = 0,1,...,l+1$, s_i đều là thặng dư bậc hai, và với mọi $i = 0,...,l$, s_i đều là căn bậc hai của s_{i+1} theo mod$n$; điều đó cũng đúng đối với mod$p$ và mod$q$. Vì $p \equiv 3$ mod4, nên mỗi thặng dư bậc hai x theo mod$p$ đều có duy nhất một căn bậc hai mod$p$ cũng là thặng dư bậc hai mod$p$, đó là $x^{(p+1)/4} \bmod p$. Thực vậy, vì $x^{(p+1)/2} \equiv x \bmod p$, nên $\pm x^{(p+1)/4} \bmod p$ là căn bậc hai theo mod$p$ của x ; mặt khác ta lại có $\left(\frac{x^{(p+1)/4}}{p}\right) = \left(\frac{x}{p}\right)^{(p+1)/4} = 1$, nên $x^{(p+1)/4} \bmod p$ cũng là một thặng dư bậc hai mod$p$. Từ nhận xét đó ta suy ra với mọi i $(i = 0,1,..,l)$:

$$s_i \equiv s_{i+1}^{(p+1)/4} \pmod p,$$

do đó,

$$s_0 = s_{l+1}^{((p+1)/4)^{l+1}} \bmod p = s_{l+1}^{a_1} \bmod p.$$

Xét tương tự đối với q, ta cũng được

$$s_0 = s_{l+1}^{a_2} \bmod q.$$

Vậy số s_0 tính theo các bước 1-3 của thuật toán giải mã đúng là mầm $s_0 = r$ mà ta cần tìm. Các thuật toán lập mật mã và giải mã như được định nghĩa ở trên là hợp thức.

Thí dụ : Chọn $n = 192649 = 383.503$.

Cho bản rõ $x = 11010011010011101101$. $(l = 20)$

Giả sử chọn ngẫu nhiên $s_0 = r = 20749$. Ta tính được dãy z :

z = 11001110000100111010.

Ta tính thêm được s_{21}=94739, và bản mã được gửi đi là

$$e_{K'}(x, r) = (y, s_{l+1}) = (y, 94739),$$

trong đó y = 00011101010111010111.

Để giải mã, trước hết ta tìm s_0 từ s_{21} = 94739. Ta có

$(p+1)/4 = 96$, $(q+1)/4 = 126$.

Theo thuật toán giải mã:

$a_1 = 96^{21} \bmod 382 = 266$,

$a_2 = 126^{21} \bmod 502 = 486$.

Từ đó tính được

$b_1 = 94739^{266} \bmod 383 = 67$,

$b_2 = 94739^{486} \bmod 503 = 126$.

Giải hệ phương trình đồng dư:

$$\begin{cases} s_0 \equiv 67 \pmod{383} \\ s_0 \equiv 126 \pmod{503} \end{cases}$$

ta được s_0=20749, từ đó tính lại được dãy z, cộng mod2 từng bit với y ta lại thu được bản rõ x .

Bài tập

5.1 In Algorithm 5.1, prove that

$$\gcd(r_0, r_1) = \gcd(r_1, r_2) = \cdots = \gcd(r_{m-1}, r_m) = r_m$$

and, hence, $r_m = \gcd(a, b)$.

5.2 Suppose that $a > b$ in Algorithm 5.1.

(a) Prove that $r_i \geq 2r_{i+2}$ for all i such that $0 \leq i \leq m-2$.

(b) Prove that m is $O(\log a)$.

(c) Prove that m is $O(\log b)$.

5.3 Use the EXTENDED EUCLIDEAN ALGORITHM to compute the following multiplicative inverses:

(a) $17^{-1} \bmod 101$

(b) $357^{-1} \bmod 1234$

(c) $3125^{-1} \bmod 9987$.

5.4 Compute $\gcd(57, 93)$, and find integers s and t such that $57s + 93t = \gcd(57, 93)$.

5.4 Compute $\gcd(91, 56)$, and find integers s and t such that $91s + 56t = \gcd(91, 56)$.

5.5 Suppose $\chi : \mathbb{Z}_{105} \to \mathbb{Z}_3 \times \mathbb{Z}_5 \times \mathbb{Z}_7$ is defined as

$$\chi(x) = (x \bmod 3, x \bmod 5, x \bmod 7).$$

Give an explicit formula for the function χ^{-1}, and use it to compute $\chi^{-1}(2, 2, 3)$.

5.6 Solve the following system of congruences:

$$x \equiv 12 \pmod{25}$$

$$x \equiv 9 \pmod{26}$$

$$x \equiv 23 \pmod{27}.$$

5.7 Solve the following system of congruences:

$$13x \equiv 4 \pmod{99}$$

$$15x \equiv 56 \pmod{101}.$$

HINT First use the EXTENDED EUCLIDEAN ALGORITHM, and then apply the Chinese remainder theorem.

5.8 Use Theorem 5.8 to find the smallest primitive element modulo 97.

5.9 Suppose that $p = 2q + 1$, where p and q are odd primes. Suppose further that $\alpha \in \mathbb{Z}_p^*$, $\alpha \not\equiv \pm 1 \pmod{p}$. Prove that α is a primitive element modulo p if and only if $\alpha^q \equiv -1 \pmod{p}$.

5.10 Suppose that $n = pq$, where p and q are distinct odd primes and $ab \equiv 1 \pmod{(p-1)(q-1)}$. The **RSA** encryption operation is $e(x) = x^b \bmod n$ and the decryption operation is $d(y) = y^a \bmod n$. We proved that $d(e(x)) = x$ if $x \in \mathbb{Z}_n^*$. Prove that the same statement is true for any $x \in \mathbb{Z}_n$.

HINT Use the fact that $x_1 \equiv x_2 \pmod{pq}$ if and only if $x_1 \equiv x_2 \pmod{p}$ and $x_1 \equiv x_2 \pmod{q}$. This follows from the Chinese remainder theorem.

5.11 For $n = pq$, where p and q are distinct odd primes, define

$$\lambda(n) = \frac{(p-1)(q-1)}{\gcd(p-1, q-1)}.$$

Suppose that we modify the *RSA Cryptosystem* by requiring that $ab \equiv 1 \pmod{\lambda(n)}$.

(a) Prove that encryption and decryption are still inverse operations in this modified cryptosystem.

(b) If $p = 37$, $q = 79$, and $b = 7$, compute a in this modified cryptosystem, as well as in the original *RSA Cryptosystem*.

5.12 Two samples of RSA ciphertext are presented in Tables 5.1 and 5.2. Your task is to decrypt them. The public parameters of the system are $n = 18923$ and $b = 1261$ (for Table 5.1) and $n = 31313$ and $b = 4913$ (for Table 5.2). This can be accomplished as follows. First, factor n (which is easy because it is so small). Then compute the exponent a from $\phi(n)$, and, finally, decrypt the ciphertext. Use the SQUARE-AND-MULTIPLY ALGORITHM to exponentiate modulo n.

In order to translate the plaintext back into ordinary English text, you need to know how alphabetic characters are "encoded" as elements in $\mathbb{Z}_n$. Each element of $\mathbb{Z}_n$ represents three alphabetic characters as in the following examples:

$$\begin{array}{lcll} DOG & \rightarrow & 3 \times 26^2 + 14 \times 26 + 6 & = & 2398 \\ CAT & \rightarrow & 2 \times 26^2 + 0 \times 26 + 19 & = & 1371 \\ ZZZ & \rightarrow & 25 \times 26^2 + 25 \times 26 + 25 & = & 17575. \end{array}$$

You will have to invert this process as the final step in your program.

The first plaintext was taken from "The Diary of Samuel Marchbanks," by Robertson Davies, 1947, and the second was taken from "Lake Wobegon Days," by Garrison Keillor, 1985.

TABLE 5.1
RSA ciphertext

12423	11524	7243	7459	14303	6127	10964	16399
9792	13629	14407	18817	18830	13556	3159	16647
5300	13951	81	8986	8007	13167	10022	17213
2264	961	17459	4101	2999	14569	17183	15827
12693	9553	18194	3830	2664	13998	12501	18873
12161	13071	16900	7233	8270	17086	9792	14266
13236	5300	13951	8850	12129	6091	18110	3332
15061	12347	7817	7946	11675	13924	13892	18031
2620	6276	8500	201	8850	11178	16477	10161
3533	13842	7537	12259	18110	44	2364	15570
3460	9886	8687	4481	11231	7547	11383	17910
12867	13203	5102	4742	5053	15407	2976	9330
12192	56	2471	15334	841	13995	17592	13297
2430	9741	11675	424	6686	738	13874	8168
7913	6246	14301	1144	9056	15967	7328	13203
796	195	9872	16979	15404	14130	9105	2001
9792	14251	1498	11296	1105	4502	16979	1105
56	4118	11302	5988	3363	15827	6928	4191
4277	10617	874	13211	11821	3090	18110	44
2364	15570	3460	9886	9988	3798	1158	9872
16979	15404	6127	9872	3652	14838	7437	2540
1367	2512	14407	5053	1521	297	10935	17137
2186	9433	13293	7555	13618	13000	6490	5310
18676	4782	11374	446	4165	11634	3846	14611
2364	6789	11634	4493	4063	4576	17955	7965
11748	14616	11453	17666	925	56	4118	18031
9522	14838	7437	3880	11476	8305	5102	2999
18628	14326	9175	9061	650	18110	8720	15404
2951	722	15334	841	15610	2443	11056	2186

TABLE 5.2
RSA ciphertext

6340	8309	14010	8936	27358	25023	16481	25809
23614	7135	24996	30590	27570	26486	30388	9395
27584	14999	4517	12146	29421	26439	1606	17881
25774	7647	23901	7372	25774	18436	12056	13547
7908	8635	2149	1908	22076	7372	8686	1304
4082	11803	5314	107	7359	22470	7372	22827
15698	30317	4685	14696	30388	8671	29956	15705
1417	26905	25809	28347	26277	7897	20240	21519
12437	1108	27106	18743	24144	10685	25234	30155
23005	8267	9917	7994	9694	2149	10042	27705
15930	29748	8635	23645	11738	24591	20240	27212
27486	9741	2149	29329	2149	5501	14015	30155
18154	22319	27705	20321	23254	13624	3249	5443
2149	16975	16087	14600	27705	19386	7325	26277
19554	23614	7553	4734	8091	23973	14015	107
3183	17347	25234	4595	21498	6360	19837	8463
6000	31280	29413	2066	369	23204	8425	7792
25973	4477	30989					

5.13 A common way to speed up RSA decryption incorporates the Chinese remainder theorem, as follows. Suppose that $d_K(y) = y^d \bmod n$ and $n = pq$. Define $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$; and let $M_p = q^{-1} \bmod p$ and $M_q = p^{-1} \bmod q$. Then consider the following algorithm:

Algorithm 5.15: CRT-OPTIMIZED RSA DECRYPTION$(n, d_p, d_q, M_p, M_q, y)$

$x_p \leftarrow y^{d_p} \bmod p$
$x_q \leftarrow y^{d_q} \bmod q$
$x \leftarrow M_p q x_p + M_q p x_q \bmod n$
return (x)

Algorithm 5.15 replaces an exponentiation modulo n by modular exponentiations modulo p and q. If p and q are ℓ-bit integers and exponentiation modulo an ℓ-bit integer takes time $c\ell^3$, then the time to perform the required exponentiation(s) is reduced from $c(2\ell)^3$ to $2c\ell^3$, a savings of 75%. The final step, involving the Chinese remainder theorem, requires time $O(\ell^2)$ if d_p, d_q, M_p and M_q have been pre-computed.

(a) Prove that the value x returned by Algorithm 5.15 is, in fact, $y^d \bmod n$.

(b) Given that $p = 1511$, $q = 2003$ and $d = 1234577$, compute d_p, d_q, M_p and M_q.

(c) Given the above values of p, q and d, decrypt the ciphertext $y = 152702$ using Algorithm 5.15.

5.14 Prove that the *RSA Cryptosystem* is insecure against a chosen ciphertext attack. In particular, given a ciphertext y, describe how to choose a ciphertext $\hat{y} \neq y$, such that knowledge of the plaintext $\hat{x} = d_K(\hat{y})$ allows $x = d_K(y)$ to be computed.

HINT Use the multiplicative property of the *RSA Cryptosystem*, i.e., that

$$e_K(x_1)e_K(x_2) \bmod n = e_K(x_1x_2 \bmod n).$$

5.15 This exercise exhibits what is called a *protocol failure*. It provides an example where ciphertext can be decrypted by an opponent, without determining the key, if a cryptosystem is used in a careless way. The moral is that it is not sufficient to use a "secure" cryptosystem in order to guarantee "secure" communication.

Suppose Bob has an *RSA Cryptosystem* with a large modulus n for which the factorization cannot be found in a reasonable amount of time. Suppose Alice sends a message to Bob by representing each alphabetic character as an integer between 0 and 25 (i.e., $A \leftrightarrow 0$, $B \leftrightarrow 1$, etc.), and then encrypting each residue modulo 26 as a separate plaintext character.

(a) Describe how Oscar can easily decrypt a message which is encrypted in this way.

(b) Illustrate this attack by decrypting the following ciphertext (which was encrypted using an *RSA Cryptosystem* with $n = 18721$ and $b = 25$) without factoring the modulus:

$$365, 0, 4845, 14930, 2608, 2608, 0.$$

5.16 This exercise illustrates another example of a protocol failure (due to Simmons) involving the *RSA Cryptosystem*; it is called the "common modulus protocol failure." Suppose Bob has an *RSA Cryptosystem* with modulus n and encryption exponent b_1, and Charlie has an *RSA Cryptosystem* with (the same) modulus n and encryption exponent b_2. Suppose also that $\gcd(b_1, b_2) = 1$. Now, consider the situation that arises if Alice encrypts the same plaintext x to send to both Bob and Charlie. Thus, she computes $y_1 = x^{b_1} \bmod n$ and $y_2 = x^{b_2} \bmod n$, and then she sends y_1 to Bob and y_2 to Charlie. Suppose Oscar intercepts y_1 and y_2, and performs the computations indicated in Algorithm 5.16.

Algorithm 5.16: RSA COMMON MODULUS DECRYPTION(n, b_1, b_2, y_1, y_2)

$c_1 \leftarrow {b_1}^{-1} \bmod b_2$
$c_2 \leftarrow (c_1 b_1 - 1)/b_2$
$x_1 \leftarrow y_1^{c_1} (y_2^{c_2})^{-1} \bmod n$
return (x_1)

(a) Prove that the value x_1 computed in Algorithm 5.16 is in fact Alice's plaintext, x. Thus, Oscar can decrypt the message Alice sent, even though the cryptosystem may be "secure."

(b) Illustrate the attack by computing x by this method if $n = 18721$, $b_1 = 43$, $b_2 = 7717$, $y_1 = 12677$ and $y_2 = 14702$.

5.17 We give yet another protocol failure involving the *RSA Cryptosystem*. Suppose that three users in a network, say Bob, Bart and Bert, all have public encryption exponents $b = 3$. Let their moduli be denoted by n_1, n_2, n_3, and assume that n_1, n_2 and n_3, are pairwise relatively prime. Now suppose Alice encrypts the same plaintext x to send to Bob, Bart and Bert. That is, Alice computes $y_i = x^3 \bmod n_i$, $1 \leq i \leq 3$. Describe how Oscar can compute x, given y_1, y_2 and y_3, without factoring any of the moduli.

5.18 A plaintext x is said to be *fixed* if $e_K(x) = x$. Show that, for the *RSA Cryptosystem*, the number of fixed plaintexts $x \in \mathbb{Z}_n^*$ is equal to

$$\gcd(b-1, p-1) \times \gcd(b-1, q-1).$$

HINT Consider the following system of two congruences:

$$e_K(x) \equiv x \pmod{p},$$
$$e_K(x) \equiv x \pmod{q}.$$

5.19 Suppose **A** is a deterministic algorithm which is given as input an RSA modulus n, an encryption exponent b, and a ciphertext y. **A** will either decrypt y or return no answer. Supposing that there are $\epsilon(n-1)$ nonzero ciphertexts which **A** is able to decrypt, show how to use **A** as an oracle in a Las Vegas decryption algorithm having success probability ϵ.

5.20 Write a program to evaluate Jacobi symbols using the four properties presented in Section 5.4. The program should not do any factoring, other than dividing out powers of two. Test your program by computing the following Jacobi symbols:

$$\left(\frac{610}{987}\right), \left(\frac{20964}{1987}\right), \left(\frac{1234567}{11111111}\right).$$

5.21 For $n = 837, 851$ and 1189, find the number of bases b such that n is an Euler pseudo-prime to the base b.

5.22 The purpose of this question is to prove that the error probability of the Solovay-Strassen primality test is at most $1/2$. Let $\mathbb{Z}_n^*$ denote the group of units modulo n. Define

$$G(n) = \left\{ a : a \in \mathbb{Z}_n^*, \left(\frac{a}{n}\right) \equiv a^{(n-1)/2} \pmod{n} \right\}.$$

(a) Prove that $G(n)$ is a subgroup of $\mathbb{Z}_n^*$. Hence, by Lagrange's theorem, if $G(n) \neq \mathbb{Z}_n^*$, then

$$|G(n)| \leq \frac{|\mathbb{Z}_n^*|}{2} \leq \frac{n-1}{2}.$$

(b) Suppose $n = p^k q$, where p and q are odd, p is prime, $k \geq 2$, and $\gcd(p, q) = 1$. Let $a = 1 + p^{k-1} q$. Prove that

$$\left(\frac{a}{n}\right) \not\equiv a^{(n-1)/2} \pmod{n}.$$

HINT Use the binomial theorem to compute $a^{(n-1)/2}$.

(c) Suppose $n = p_1 \dots p_s$, where the p_i's are distinct odd primes. Suppose $a \equiv u \pmod{p_1}$ and $a \equiv 1 \pmod{p_2 p_3 \dots p_s}$, where u is a quadratic non-residue modulo p_1 (note that such an a exists by the Chinese remainder theorem). Prove that

$$\left(\frac{a}{n}\right) \equiv -1 \pmod{n},$$

but

$$a^{(n-1)/2} \equiv 1 \pmod{p_2 p_3 \dots p_s},$$

so

$$a^{(n-1)/2} \not\equiv -1 \pmod{n}.$$

(d) If n is odd and composite, prove that $|G(n)| \leq (n-1)/2$.

(e) Summarize the above: prove that the error probability of the Solovay-Strassen primality test is at most $1/2$.

5.23 Suppose we have a Las Vegas algorithm with failure probability ϵ.

(a) Prove that the probability of first achieving success on the nth trial is $p_n = \epsilon^{n-1}(1-\epsilon)$.

(b) The average (expected) number of trials to achieve success is

$$\sum_{n=1}^{\infty} (n \times p_n).$$

Show that this average is equal to $1/(1-\epsilon)$.

(c) Let δ be a positive real number less than 1. Show that the number of iterations required in order to reduce the probability of failure to at most δ is

$$\left\lceil \frac{\log_2 \delta}{\log_2 \epsilon} \right\rceil.$$

5.24 Suppose throughout this question that p is an odd prime and $\gcd(a,p) = 1$.

(a) Suppose that $i \geq 2$ and $b^2 \equiv a \pmod{p^{i-1}}$. Prove that there is a unique $x \in \mathbb{Z}_{p^i}$ such that $x^2 \equiv a \pmod{p^i}$ and $x \equiv b \pmod{p^{i-1}}$. Describe how this x can be computed efficiently.

(b) Illustrate your method in the following situation: starting with the congruence $6^2 \equiv 17 \pmod{19}$, find square roots of 17 modulo 19^2 and modulo 19^3.

(c) For all $i \geq 1$, prove that the number of solutions to the congruence $x^2 \equiv a \pmod{p^i}$ is either 0 or 2.

5.25 Using various choices for the bound, B, attempt to factor 262063 and 9420457 using the $p-1$ method. How big does B have to be in each case to be successful?

5.26 Factor 262063, 9420457 and 181937053 using the POLLARD RHO ALGORITHM, if the function f is defined to be $f(x) = x^2 + 1$. How many iterations are needed to factor each of these three integers?

5.27 Suppose we want to factor the integer $n = 256961$ using the RANDOM SQUARES ALGORITHM. Using the factor base

$$\{-1, 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31\},$$

test the integers $z^2 \bmod n$ for $z = 500, 501, \ldots$, until a congruence of the form $x^2 \equiv y^2 \pmod{n}$ is obtained and the factorization of n is found.

5.28 In the RANDOM SQUARES ALGORITHM, we need to test a positive integer $w \leq n-1$ to see if it factors completely over the factor base $\mathcal{B} = \{p_1, \ldots, p_B\}$ consisting of the B smallest prime numbers. Recall that $p_B = m \approx 2^s$ and $n \approx 2^r$.

(a) Prove that this can be done using at most $B + r$ divisions of an integer having at most r bits by an integer having at most s bits.

(b) Assuming that $r < m$, prove that the complexity of this test is $O(rsm)$.

5.29 In this exercise, we show that parameter generation for the *RSA Cryptosystem* should take care to ensure that $q - p$ is not too small, where $n = pq$ and $q > p$.

(a) Suppose that $q - p = 2d > 0$, and $n = pq$. Prove that $n + d^2$ is a perfect square.

(b) Given an integer n which is the product of two odd primes, and given a small positive integer d such that $n + d^2$ is a perfect square, show how this information can be used to factor n.

(c) Use this technique to factor $n = 2189284635403183$.

5.30 Suppose Bob has carelessly revealed his decryption exponent to be $a = 14039$ in an *RSA Cryptosystem* with public key $n = 36581$ and $b = 4679$. Implement the randomized algorithm to factor n given this information. Test your algorithm with the "random" choices $w = 9983$ and $w = 13461$. Show all computations.

5.31 If $q_1, \ldots, q_m$ is the sequence of quotients obtained in the applying the EUCLIDEAN ALGORITHM with input r_0, r_1, prove that the continued fraction $[q_1, \ldots, q_m] = r_0/r_1$.

5.32 Suppose that $n = 317940011$ and $b = 77537081$ in the *RSA Cryptosystem*. Using WIENER'S ALGORITHM, attempt to factor n.

5.33 Consider the modification of the *Rabin Cryptosystem* in which $e_K(x) = x(x + B) \bmod n$, where $B \in \mathbb{Z}_n$ is part of the public key. Supposing that $p = 199$, $q = 211$, $n = pq$ and $B = 1357$, perform the following computations.

(a) Compute the encryption $y = e_K(32767)$.

(b) Determine the four possible decryptions of this given ciphertext y.

5.34 Prove Equations (5.3) and (5.4) relating the functions *half* and *parity*.

5.35 Prove that Cryptosystem 5.3 is not semantically secure against a chosen ciphertext attack. Given x_1, x_2, a ciphertext (y_1, y_2) that is an encryption of x_i ($i = 1$ or 2), and given a decryption oracle DECRYPT for Cryptosystem 5.3, describe an algorithm to determine whether $i = 1$ or $i = 2$. You are allowed to call the algorithm DECRYPT with any input except for the given ciphertext (y_1, y_2), and it will output the corresponding plaintext.

Bài tập hệ mật ElGamal

6.1 Implement SHANKS' ALGORITHM for finding discrete logarithms in $\mathbb{Z}_p^*$, where p is prime and α is a primitive element modulo p. Use your program to find

$\log_{106} 12375$ in $\mathbb{Z}_{24691}^*$ and $\log_6 248388$ in $\mathbb{Z}_{458009}^*$.

6.2 Describe how to modify SHANKS' ALGORITHM to compute the logarithm of β to the base α in a group G if it is specified ahead of time that this logarithm lies in the interval $[s, t]$, where s and t are integers such that $0 \leq s < t < n$, where n is the order of α. Prove that your algorithm is correct, and show that its complexity is $O(\sqrt{t - s})$.

6.3 The integer $p = 458009$ is prime and $\alpha = 2$ has order 57251 in $\mathbb{Z}_p^*$. Use the POLLARD RHO ALGORITHM to compute the discrete logarithm in $\mathbb{Z}_p^*$ of $\beta = 56851$ to the base α. Take the initial value $x_0 = 1$, and define the partition (S_1, S_2, S_3) as in Example 6.3. Find the smallest integer i such that $x_i = x_{2i}$, and then compute the desired discrete logarithm.

6.4 Suppose that p is an odd prime and k is a positive integer. The multiplicative group $\mathbb{Z}_{p^k}^*$ has order $p^{k-1}(p - 1)$, and is known to be cyclic. A generator for this group is called a *primitive element modulo* p^k.

(a) Suppose that α is a primitive element modulo p. Prove that at least one of α or $\alpha + p$ is a primitive element modulo p^2.

(b) Describe how to efficiently verify that 3 is a primitive root modulo 29 and modulo 29^2. Note: It can be shown that if α is a primitive root modulo p and modulo p^2, then it is a primitive root modulo p^k for all positive integers k (you do not have to prove this fact). Therefore, it follows that 3 is a primitive root modulo 29^k for all positive integers k.

(c) Find an integer α that is a primitive root modulo 29 but not a primitive root modulo 29^2.

(d) Use the POHLIG-HELLMAN ALGORITHM to compute the discrete logarithm of 3344 to the base 3 in the multiplicative group $\mathbb{Z}_{24389}^*$.

6.5 Implement the POHLIG-HELLMAN ALGORITHM for finding discrete logarithms in $\mathbb{Z}_p$, where p is prime and α is a primitive element. Use your program to find $\log_5 8563$ in $\mathbb{Z}_{28703}$ and $\log_{10} 12611$ in $\mathbb{Z}_{31153}$.

6.6 Let $p = 227$. The element $\alpha = 2$ is primitive in $\mathbb{Z}_p^*$.

(a) Compute α^{32}, α^{40}, α^{59} and α^{156} modulo p, and factor them over the factor base $\{2, 3, 5, 7, 11\}$.

(b) Using the fact that $\log 2 = 1$, compute $\log 3$, $\log 5$, $\log 7$ and $\log 11$ from the factorizations obtained above (all logarithms are discrete logarithms in $\mathbb{Z}_p^*$ to the base α).

(c) Now suppose we wish to compute $\log 173$. Multiply 173 by the "random" value $2^{177} \bmod p$. Factor the result over the factor base, and proceed to compute $\log 173$ using the previously computed logarithms of the numbers in the factor base.

6.7 Suppose that $n = pq$ is an RSA modulus (i.e., p and q are distinct odd primes), and let $\alpha \in \mathbb{Z}_n^*$. For a positive integer m and for any $\alpha \in \mathbb{Z}_m^*$, define $\mathsf{ord}_m(\alpha)$ to be the order of α in the group $\mathbb{Z}_m^*$.

(a) Prove that

$$\mathsf{ord}_n(\alpha) = \mathrm{lcm}(\mathsf{ord}_p(\alpha), \mathsf{ord}_q(\alpha)).$$

(b) Suppose that $\gcd(p-1, q-1) = d$. Show that there exists an element $\alpha \in \mathbb{Z}_n^*$ such that

$$\mathsf{ord}_n(\alpha) = \frac{\phi(n)}{d}.$$

(c) Suppose that $\gcd(p-1, q-1) = 2$, and we have an oracle that solves the **Discrete Logarithm** problem in the subgroup $\langle \alpha \rangle$, where $\alpha \in \mathbb{Z}_n^*$ has order $\phi(n)/2$. That is, given any $\beta \in \langle \alpha \rangle$, the oracle will find the discrete logarithm $a = \log_\alpha \beta$, where $0 \leq a \leq \phi(n)/2 - 1$. (The value $\phi(n)/2$ is secret however.) Suppose we compute the value $\beta = \alpha^n \bmod n$ and then we use the oracle to find $a = \log_\alpha \beta$. Assuming that $p > 3$ and $q > 3$, prove that $n - a = \phi(n)$.

(d) Describe how n can easily be factored, given the discrete logarithm $a = \log_\alpha \beta$ from (c).

6.8 In this question, we consider a generic algorithm for the **Discrete Logarithm** problem in $(\mathbb{Z}_{19}, +)$.

(a) Suppose that the set C is defined as follows:

$$C = \{(1 - i^2 \bmod 19, i \bmod 19) : i = 0, 1, 2, 4, 7, 12\}.$$

Compute $\mathsf{Good}(C)$.

(b) Suppose that the output of the group oracle, given the ordered pairs in C, is as follows:

$$(0,1) \mapsto 10111$$
$$(1,0) \mapsto 01100$$
$$(16,2) \mapsto 00110$$
$$(4,4) \mapsto 01010$$
$$(9,7) \mapsto 00100$$
$$(9,12) \mapsto 11001,$$

where group elements are encoded as (random) binary 5-tuples. What can you say about the value of "a"?

6.9 Decrypt the ElGamal ciphertext presented in Table 6.3. The parameters of the system are $p = 31847$, $\alpha = 5$, $a = 7899$ and $\beta = 18074$. Each element of $\mathbb{Z}_n$ represents three alphabetic characters as in Exercise 5.12.

The plaintext was taken from "The English Patient," by Michael Ondaatje, Alfred A. Knopf, Inc., New York, 1992.

TABLE 6.3
ElGamal Ciphertext

(3781, 14409)	(31552, 3930)	(27214, 15442)	(5809, 30274)
(5400, 31486)	(19936, 721)	(27765, 29284)	(29820, 7710)
(31590, 26470)	(3781, 14409)	(15898, 30844)	(19048, 12914)
(16160, 3129)	(301, 17252)	(24689, 7776)	(28856, 15720)
(30555, 24611)	(20501, 2922)	(13659, 5015)	(5740, 31233)
(1616, 14170)	(4294, 2307)	(2320, 29174)	(3036, 20132)
(14130, 22010)	(25910, 19663)	(19557, 10145)	(18899, 27609)
(26004, 25056)	(5400, 31486)	(9526, 3019)	(12962, 15189)
(29538, 5408)	(3149, 7400)	(9396, 3058)	(27149, 20535)
(1777, 8737)	(26117, 14251)	(7129, 18195)	(25302, 10248)
(23258, 3468)	(26052, 20545)	(21958, 5713)	(346, 31194)
(8836, 25898)	(8794, 17358)	(1777, 8737)	(25038, 12483)
(10422, 5552)	(1777, 8737)	(3780, 16360)	(11685, 133)
(25115, 10840)	(14130, 22010)	(16081, 16414)	(28580, 20845)
(23418, 22058)	(24139, 9580)	(173, 17075)	(2016, 18131)
(19886, 22344)	(21600, 25505)	(27119, 19921)	(23312, 16906)
(21563, 7891)	(28250, 21321)	(28327, 19237)	(15313, 28649)
(24271, 8480)	(26592, 25457)	(9660, 7939)	(10267, 20623)
(30499, 14423)	(5839, 24179)	(12846, 6598)	(9284, 27858)
(24875, 17641)	(1777, 8737)	(18825, 19671)	(31306, 11929)
(3576, 4630)	(26664, 27572)	(27011, 29164)	(22763, 8992)
(3149, 7400)	(8951, 29435)	(2059, 3977)	(16258, 30341)
(21541, 19004)	(5865, 29526)	(10536, 6941)	(1777, 8737)
(17561, 11884)	(2209, 6107)	(10422, 5552)	(19371, 21005)
(26521, 5803)	(14884, 14280)	(4328, 8635)	(28250, 21321)
(28327, 19237)	(15313, 28649)		

6.10 Determine which of the following polynomials are irreducible over $\mathbb{Z}_2[x]$: $x^5 + x^4 + 1$, $x^5 + x^3 + 1$, $x^5 + x^4 + x^2 + 1$.

6.11 The field $\mathbb{F}_{2^5}$ can be constructed as $\mathbb{Z}_2[x]/(x^5 + x^2 + 1)$. Perform the following computations in this field.

(a) Compute $(x^4 + x^2) \times (x^3 + x + 1)$.

(b) Using the extended Euclidean algorithm, compute $(x^3 + x^2)^{-1}$.

(c) Using the square-and-multiply algorithm, compute x^{25}.

6.12 We give an example of the *ElGamal Cryptosystem* implemented in $\mathbb{F}_{3^3}$. The polynomial $x^3 + 2x^2 + 1$ is irreducible over $\mathbb{Z}_3[x]$ and hence $\mathbb{Z}_3[x]/(x^3 + 2x^2 + 1)$ is the field $\mathbb{F}_{3^3}$. We can associate the 26 letters of the alphabet with the 26 nonzero field elements, and thus encrypt ordinary text in a convenient way. We will use a lexicographic ordering of the (nonzero) polynomials to set up the correspondence.

This correspondence is as follows:

A	$\leftrightarrow$	1	B	$\leftrightarrow$	2	C	$\leftrightarrow$	x
D	$\leftrightarrow$	$x+1$	E	$\leftrightarrow$	$x+2$	F	$\leftrightarrow$	$2x$
G	$\leftrightarrow$	$2x+1$	H	$\leftrightarrow$	$2x+2$	I	$\leftrightarrow$	x^2
J	$\leftrightarrow$	x^2+1	K	$\leftrightarrow$	x^2+2	L	$\leftrightarrow$	x^2+x
M	$\leftrightarrow$	x^2+x+1	N	$\leftrightarrow$	x^2+x+2	O	$\leftrightarrow$	x^2+2x
P	$\leftrightarrow$	x^2+2x+1	Q	$\leftrightarrow$	x^2+2x+2	R	$\leftrightarrow$	$2x^2$
S	$\leftrightarrow$	$2x^2+1$	T	$\leftrightarrow$	$2x^2+2$	U	$\leftrightarrow$	$2x^2+x$
V	$\leftrightarrow$	$2x^2+x+1$	W	$\leftrightarrow$	$2x^2+x+2$	X	$\leftrightarrow$	$2x^2+2x$
Y	$\leftrightarrow$	$2x^2+2x+1$	Z	$\leftrightarrow$	$2x^2+2x+2$			

Suppose Bob uses $\alpha = x$ and $a = 11$ in an *ElGamal Cryptosystem*; then $\beta = x+2$. Show how Bob will decrypt the following string of ciphertext:

```
(K,H) (P,X) (N,K) (H,R) (T,F) (V,Y) (E,H) (F,A) (T,W) (J,D) (U,J)
```

6.13 Let E be the elliptic curve $y^2 = x^3 + x + 28$ defined over $\mathbb{Z}_{71}$.

(a) Determine the number of points on E.

(b) Show that E is not a cyclic group.

(c) What is the maximum order of an element in E? Find an element having this order.

6.14 Suppose that $p > 3$ is an odd prime, and $a, b \in \mathbb{Z}_p$. Further, suppose that the equation $x^3 + ax + b \equiv 0 \pmod{p}$ has three distinct roots in $\mathbb{Z}_p$. Prove that the corresponding elliptic curve group $(E, +)$ is not cyclic.

HINT Show that the points of order two generate a subgroup of $(E, +)$ that is isomorphic to $\mathbb{Z}_2 \times \mathbb{Z}_2$.

6.15 Consider an elliptic curve E described by the formula $y^2 \equiv x^3 + ax + b \pmod{p}$, where $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$ and $p > 3$ is prime.

(a) It is clear that a point $P = (x_1, y_1) \in E$ has order 3 if and only if $2P = -P$. Use this fact to prove that, if $P = (x_1, y_1) \in E$ has order 3, then

$$3{x_1}^4 + 6a{x_1}^2 + 12x_1 b - a^2 \equiv 0 \pmod{p}. \tag{6.7}$$

(b) Conclude from equation (6.7) that there are at most 8 points of order 3 on the elliptic curve E.

(c) Using equation (6.7), determine all points of order 3 on the elliptic curve $y^2 \equiv x^3 + 34x \pmod{73}$.

6.16 Suppose that E is an elliptic curve defined over $\mathbb{Z}_p$, where $p > 3$ is prime. Suppose that $\#E$ is prime, $P \in E$, and $P \neq \mathcal{O}$.

(a) Prove that the discrete logarithm $\log_P(-P) = \#E - 1$.

(b) Describe how to compute $\#E$ in time $O(p^{1/4})$ by using Hasse's bound on $\#E$, together with a modification of SHANKS' ALGORITHM. Give a pseudocode description of the algorithm.

6.17 Let E be the elliptic curve $y^2 = x^3 + 2x + 7$ defined over $\mathbb{Z}_{31}$. It can be shown that $\#E = 39$ and $P = (2, 9)$ is an element of order 39 in E. The *Simplified ECIES* defined on E has $\mathbb{Z}_{31}^*$ as its plaintext space. Suppose the private key is $m = 8$.

(a) Compute $Q = mP$.

(b) Decrypt the following string of ciphertext:

$$((18, 1), 21), ((3, 1), 18), ((17, 0), 19), ((28, 0), 8).$$

(c) Assuming that each plaintext represents one alphabetic character, convert the plaintext into an English word. (Here we will use the correspondence $A \leftrightarrow 1$, $\ldots$, $Z \leftrightarrow 26$, because 0 is not allowed in a (plaintext) ordered pair.)

6.18 (a) Determine the NAF representation of the integer 87.

(b) Using the NAF representation of 87, use Algorithm 6.5 to compute $87P$, where $P = (2, 6)$ is a point on the elliptic curve $y^2 = x^3 + x + 26$ defined over $\mathbb{Z}_{127}$. Show the partial results during each iteration of the algorithm.

6.19 Let $\mathcal{L}_i$ denote the set of positive integers that have exactly i coefficients in their NAF representation, such that the leading coefficient is 1. Denote $k_i = |\mathcal{L}_i|$.

(a) By means of a suitable decomposition of $\mathcal{L}_i$, prove that the k_i's satisfy the following recurrence relation:

$$k_1 = 1$$

$$k_2 = 1$$

$$k_{i+1} = 2(k_1 + k_2 + \ldots + k_{i-1}) + 1 \quad (\text{for } i \geq 2).$$

(b) Derive a second degree recurrence relation for the k_i's, and obtain an explicit solution of the recurrence relation.

6.20 Find $\log_5 896$ in $\mathbb{Z}_{1103}$ using Algorithm 6.6, given that $L_2(\beta) = 1$ for $\beta = 25, 219$ and 841, and $L_2(\beta) = 0$ for $\beta = 163, 532, 625$ and 656.

6.21 Throughout this question, suppose that $p \equiv 5 \pmod 8$ is prime and suppose that a is a quadratic residue modulo p.

(a) Prove that $a^{(p-1)/4} \equiv \pm 1 \pmod p$.

(b) If $a^{(p-1)/4} \equiv 1 \pmod p$, prove that $a^{(p+3)/8} \bmod p$ is a square root of a modulo p.

(c) If $a^{(p-1)/4} \equiv -1 \pmod p$, prove that $2^{-1}(4a)^{(p+3)/8} \bmod p$ is a square root of a modulo p.

HINT Use the fact that $\left(\frac{2}{p}\right) = -1$ when $p \equiv 5 \pmod 8$ is prime.

(d) Given a primitive element $\alpha \in \mathbb{Z}_p^*$, and given any $\beta \in \mathbb{Z}_p^*$, show that $L_2(\beta)$ can be computed efficiently.

HINT Use the fact that it is possible to compute square roots modulo p, as well as the fact that $L_1(\beta) = L_1(p - \beta)$ for all $\beta \in \mathbb{Z}_p^*$, when $p \equiv 5 \pmod 8$ is prime.

6.22 The *ElGamal Cryptosystem* can be implemented in any subgroup $\langle\alpha\rangle$ of a finite multiplicative group $(G, \cdot)$, as follows: Let $\beta \in \langle\alpha\rangle$ and define (α, β) to be the public key. The plaintext space is $\mathcal{P} = \langle\alpha\rangle$, and the encryption operation is $e_K(x) = (y_1, y_2) = (\alpha^k, x \cdot \beta^k)$, where k is random.

Here we show that distinguishing ElGamal encryptions of two plaintexts can be Turing reduced to **Decision Diffie-Hellman**, and vice versa.

(a) Assume that ORACLEDDH is an oracle that solves **Decision Diffie-Hellman** in $(G, \cdot)$. Prove that ORACLEDDH can be used as a subroutine in an algorithm that distinguishes ElGamal encryptions of two given plaintexts, say x_1 and x_2. (That is, given $x_1, x_2 \in \mathcal{P}$, and given a ciphertext (y_1, y_2) which is an encryption of x_i for some $i \in \{1, 2\}$, the distinguishing algorithm will determine if $i = 1$ or $i = 2$.)

(b) Assume that ORACLE-DISTINGUISH is an oracle that distinguishes ElGamal encryptions of any two given plaintexts x_1 and x_2, for any *ElGamal Cryptosystem* implemented in the group $(G, \cdot)$ as described above. Suppose further that ORACLE-DISTINGUISH will determine if a ciphertext (y_1, y_2) is not a valid encryption of either of x_1 or x_2. Prove that ORACLE-DISTINGUISH can be used as a subroutine in an algorithm that solves **Decision Diffie-Hellman** in $(G, \cdot)$.