8.3 Computational and Programming Exercises

Computations and Explorations

Using a computation program such as Maple or *Mathematica*, or programs you have written, carry out the following computations and explorations.

1. Encrypt some messages for your classmates to decrypt using exponentiation ciphers.
2. Decrypt messages encrypted by your classmates using exponentiation ciphers, given the encryption key and prime modulus.

Programming Projects

Write computer programs using Maple, *Mathematica*, or a language of your choice to do the following.

1. Encrypt some messages for your classmates to decrypt using exponentiation ciphers.
2. Decrypt messages encrypted by your classmates using exponentiation ciphers, given the encrypting key and prime modules.

8.4 Public Key Cryptography

The cryptosystems we have discussed so far are all examples of *private key*, or *symmetric* cryptosystems, where the encryption and decryption keys are either the same or can be easily found from each other. For example, in a shift cipher, the encrypting key is an integer k and the corresponding decrypting key is the integer $-k$. In an affine cipher, the encrypting key is a pair (a, b) and the corresponding decrypting key is the pair $(\overline{a}, -\overline{a}b)$, where $\overline{a}$ is an inverse of a modulo 26. In a Hill cipher, the encrypting key is an $n \times n$ matrix $\mathbf{A}$ and the corresponding decrypting key is the $n \times n$ matrix $\overline{\mathbf{A}}$, where $\overline{\mathbf{A}}$ is an inverse of the matrix $\mathbf{A}$ modulo 26. In the Pohlig-Hellman exponentiation cipher, the encrypting key is (e, p), where p is a prime, and the corresponding decrypting key is (d, p), where d is an inverse of e modulo $p - 1$. For the DEA, the encrypting and decrypting keys are exactly the same.

For that reason, if one of the cryptosystems discussed so far is used to establish secure communications within a network, then each pair of communicants must employ an encryption key that is kept secret from the other individuals in the network, because once the encryption key in such a cryptosystem is known, the decryption key can be found using a small amount of computer time. Consequently, to maintain secrecy, the encryption keys must themselves be transmitted over a channel of secure communications.

To avoid assigning a key to each pair of individuals, which must be kept secret from the rest of the network, a new type of cryptosystem, called a *public key* cryptosystem, was invented in the 1970s. In this type of cryptosystem, encrypting keys can be made public, because an unrealistically large amount of computer time is required to find a decrypting transformation from an encrypting transformation. To use a public key cryptosystem to establish secret communications in a network of n individuals, each individual produces a key of the type specified by the cryptosystem, retaining certain private information that went into the construction of the encrypting transformation $E(k)$, obtained from the key

k according to a specified rule. Then a directory of the n keys $k_1, k_2, \ldots, k_n$ is published. When individual i wishes to send a message to individual j, the letters of the message are translated into their numerical equivalents and combined into blocks of specified size. Then, for each plaintext block P a corresponding ciphertext block $C = E_{k_j}(P)$ is computed using the encrypting transformation E_{k_j}. To decrypt the message, individual j applies the decrypting transformation D_{k_j} to each ciphertext block C to find P; that is,

$$D_{k_j}(C) = D_{k_j}(E_{k_j}(P)) = P.$$

Because the decrypting transformation D_{k_j} cannot be found in a realistic amount of time by anyone other than individual j, no unauthorized individuals can decrypt the message, even though they know the key k_j. Furthermore, cryptanalysis of the ciphertext message, even with knowledge of k_j, is extremely infeasible due to the large amount of computer time needed.

Many cryptosystems have been proposed as public key cryptosystems. All but a few have been shown to be unsuitable, by demonstrating that ciphertext messages can be decrypted using a feasible amount of computer time. In this section, we will introduce the most widely used public key cryptosystem, the RSA cryptosystem. In addition, we will introduce several other public key cryptosystems, including the Rabin public key cryptosystem, which we will discuss at the end of this section, and the ElGamal public key cryptosystem, which we will discuss in Chapter 10. The security of these systems rests on the difficulty of two computationally intensive mathematical problems, factoring integers (discussed in Chapter 3) and finding discrete logarithms (to be discussed in Chapter 9). In Section 8.5, we will describe a proposed public key cryptosystem, the knapsack cryptosystem, that turned out not to be suitable as a basis for a public key cryptosystem. (See [MevaVa97] for a comprehensive look at most of the important public key cryptosystems.)

Although public key cryptosystems have many advantages, they are not extensively used for general-purpose encryption. The reason is that encrypting and decrypting in these cryptosystems require too much time and memory on most computers, generally several orders of magnitude more than required for symmetric cryptosystems currently in use. However, public key cryptosystems are used extensively to encrypt keys for symmetric cryptosystems such as DES, so that these keys can be transmitted securely. They are also used in a wide variety of cryptographic protocols, such as in digital signatures (discussed in Section 8.6). They are also particularly useful for applications involving smart cards and electronic commerce.

Also note that in modern cryptography, the cryptosystem used to encrypt messages is publicly known. Consequently, the secrecy of encrypted messages does not depend on the secrecy of the encryption algorithm in use. For symmetric key cryptosystems, the secrecy of messages depends on the secrecy of the encryption key in use and the computational difficulty of finding this key from other information (such as plaintext–ciphertext pairs). For public key cryptosystems, secrecy rests on the secrecy of the decryption key and the computational difficulty of finding this key from the encryption key and other public information (such as plaintext–ciphertext pairs).

The RSA Cryptosystem

The *RSA cryptosystem*, invented by *Ronald Rivest, Adi Shamir, and Leonard Adleman* [RiShAd78] in the 1970s (and patented by them [RiShAd83] in 1983) is a public key cryptosystem based on modular exponentiation, where the keys are pairs (e, n) consisting of an exponent e and a modulus n that is the product of two large primes; that is, $n = pq$, where p and q are large primes, so that $(e, \phi(n)) = 1$. To encrypt a message, we first translate the letters into their numerical equivalents and then form blocks of the largest possible size (with an even number of digits). To encrypt a plaintext block P, we form a ciphertext block C by

$$E(P) = C \equiv P^e \pmod{n}, \quad 0 \leq C < n.$$

The decrypting procedure requires knowledge of an inverse d of e modulo $\phi(n)$, which exists because $(e, \phi(n)) = 1$. To decrypt the ciphertext block C, we find

RONALD RIVEST (b. 1948) received his B.A. from Yale University in 1969 and his Ph.D. in computer science from Stanford University in 1974. He is a professor of computer science at M.I.T., and a cofounder of RSA Data Security, Inc. (now a subsidiary of Security Dynamics), the company that holds the patents on the RSA cryptosystem. Rivest has worked in the areas of machine learning, computer algorithms, and VLSI design. He is one of the authors of a popular textbook on algorithms ([ColeRi01]).

ADI SHAMIR (b. 1952) was born in Tel Aviv, Israel. He received his undergraduate degree from Tel Aviv University in 1972, and his Ph.D. in computer science from the Weizmann Institute of Science in 1977. He held a research assistantship at the University of Warwick for one year, and in 1978 he became an assistant professor at M.I.T. He is now a professor in the Applied Mathematics Department at the Weizmann Institute in Israel, where he formed a group to study computer security. Shamir has made many contributions to cryptography besides coinventing the RSA cryptosystem, including cracking the knapsack cryptosystem proposed as a public cryptosystem by Merkle and Hellman, developing numerous cryptographic protocols, and creative cryptanalysis of DES.

LEONARD ADLEMAN (b. 1945) was born in San Francisco, California. He received his B.S. in mathematics and his Ph.D. in computer science from the University of California, Berkeley, in 1968 and 1976, respectively. He was a member of the mathematics faculty at M.I.T. from 1976 until 1980; during his stay at M.I.T., he helped invent the RSA cryptosystem. In 1980 he was appointed to a position in the computer science department of the University of Southern California, and to a chaired professorship in 1985. Adleman has worked in the areas of computational complexity, computer security, immunology, and molecular biology, in addition to his work in cryptography. He coined the term "computer virus." His recent work on computing using DNA has attracted great interest. Adleman served as the technical adviser for the movie *Sneakers*, in which computer security figured prominently.

$$D(C) \equiv C^d = (P^e)^d = P^{ed} = P^{k\phi(n)+1}$$
$$\equiv (P^{\phi(n)})^k P \equiv P \pmod{n},$$

where $ed = k\phi(n) + 1$ for some integer k, because $ed \equiv 1 \pmod{\phi(n)}$, and by Euler's theorem, we have $P^{\phi(n)} \equiv 1 \pmod{n}$, when $(P, n) = 1$ the (probability that P and n are not relatively prime is extremely small; see Exercise 4 at the end of this section). The pair (d, n) is a decrypting key.

Example 8.16. To illustrate how the RSA cryptosystem works, suppose that the encrypting modulus is the product of the two primes 43 and 59 (which are smaller than the large primes that would actually be used); thus, we have $n = 43 \cdot 59 = 2537$ as the modulus. We take $e = 13$ as the exponent; note that we have $(e, \phi(n)) = (13, 42 \cdot 58) = 1$. To encrypt the message

PUBLIC KEY CRYPTOGRAPHY,

we first translate the letters into their numerical equivalents, and then group these numbers together into blocks of four. We obtain

1520 0111 0802 1004
2402 1724 1519 1406
1700 1507 2423,

where we have added the dummy letter $X = 23$ at the end of the passage to fill out the final block.

We encrypt each plaintext block into a ciphertext block, using the relationship

$$C \equiv P^{13} \pmod{2537}.$$

For instance, when we encrypt the first plaintext block 1520, we obtain the ciphertext block

$$C \equiv (1520)^{13} \equiv 95 \pmod{2537}.$$

Encrypting all the plaintext blocks, we obtain the ciphertext message

0095 1648 1410 1299
0811 2333 2132 0370
1185 1957 1084.

To decrypt messages that have been encrypted using this RSA cipher, we must find an inverse of $e = 13$ modulo $\phi(2537) = \phi(43 \cdot 59) = 42 \cdot 58 = 2436$. A short computation using the Euclidean algorithm, as done in Section 4.2, shows that $d = 937$ is an inverse of 13 modulo 2436. Consequently, to decrypt the ciphertext block C, we use the relationship

$$P \equiv C^{937} \pmod{2537}, \quad 0 \le P < 2537,$$

which is valid because

$$C^{937} \equiv (P^{13})^{937} \equiv (P^{2436})^5 P \equiv P \pmod{2537}.$$

Note that we have used Euler's theorem to see that

$$P^{\phi(2537)} = P^{2436} \equiv 1 \pmod{2537},$$

when $(P, 2537) = 1$ (which is true for all of the plaintext blocks in this example). ◀

The Security of the RSA Cryptosystem To understand how the RSA cryptosystem fulfills the requirements of a public key cryptosystem, first note that each individual can find two large primes p and q, each with 100 decimal digits, in just a few minutes of computer time. These primes can be found by picking odd integers with 100 digits at random; by the prime number theorem, the probability that such an integer is prime is approximately $2/\log 10^{100}$. Hence, we expect to find a prime after examining an average of $1/(2/\log 10^{100})$, or approximately 115, such integers. To test these randomly chosen odd integers for primality, we use Rabin's probabilistic primality test (discussed in Section 6.2). For each of these 100-digit odd integers we perform Miller's test for 100 bases less than the integer; the probability that a composite integer passes all these tests is less than 10^{-60}. The procedure we have just outlined requires only a few minutes of computer time to find a 100-digit prime, and each individual need do so only twice.

Once the primes p and q have been found, an encrypting exponent e must be chosen such that $(e, \phi(pq)) = 1$. One suggestion for choosing e is to take any prime greater than both p and q. No matter how e is found, it should be true that $2^e > n = pq$, so that it is impossible to recover the plaintext block P, $P \neq 0$ or 1, just by taking the eth root of the integer C with $C \equiv P^e \pmod{n}$, $0 \leq C < n$. As long as $2^e > n$, every message, other than $P = 0$ and 1, is encrypted by exponentiation followed by a reduction modulo n.

We note that the modular exponentiation needed for encrypting messages using the RSA cryptosystem can be done using only a few seconds of computer time when the modulus, exponent, and base in the modular exponentiation have as many as 200 decimal digits. Also, using the Euclidean algorithm, we can rapidly find an inverse d of the encryption exponent e modulo $\phi(n)$ when the primes p and q are known, so that $\phi(n) = \phi(pq) = (p-1)(q-1)$ is known.

To see why knowledge of the encrypting key (e, n) does not easily lead to the decrypting key (d, n), note that to find d, an inverse of e modulo $\phi(n)$, requires that we first find $\phi(n) = \phi(pq) = (p-1)(q-1)$. Note that finding $\phi(n)$ is not easier than factoring the integer n. To see why, note that $p + q = n - \phi(n) + 1$ and $p - q = \sqrt{(p+q)^2 - 4pq} = \sqrt{(p+q)^2 - 4n}$ and that $p = \frac{1}{2}[(p+q) + (p-q)]$ and $q = \frac{1}{2}[(p+q) - (p-q)]$. Consequently, p and q can easily be found when $n = pq$ and $\phi(n) = (p-1)(q-1)$ are known. Note that when p and q both have approximately 100 decimal digits, $n = pq$ has approximately 200 decimal digits. Using the fastest factorization algorithm known, millions of years of computer time are required to factor an integer of this size. Also, if the integer d is known, but $\phi(n)$ is not, then n may also be factored easily, since $ed - 1$ is a multiple of $\phi(n)$ and there are special algorithms for factoring an integer n using any multiple of $\phi(n)$ (see [Mi76]).

It has not been proven that it is impossible to decrypt messages encrypted using the RSA cryptosystem without factoring n, but so far no such method has been discovered.

As yet, all decrypting methods that work in general are equivalent to factoring n and, as we have remarked, factoring large integers seems to be an intractable problem, requiring tremendous amounts of computer time. If no method of decrypting RSA messages without factoring the modulus n is found, the security of the RSA system can be maintained as factoring methods and computational power improve, by increasing the size of the modulus. Unfortunately, messages encrypted using the RSA will become vulnerable to attack when factoring the modulus n becomes feasible. This means that extra care should be taken—for example, by using primes p and q each with several hundred digits—to protect the secrecy of messages that must be kept secret for tens, or hundreds, of years.

Note that a few extra precautions should be taken in choosing the primes p and q to be used in the RSA cryptosystem, to prevent the use of special rapid techniques to factor $n = pq$. For example, both $p - 1$ and $q - 1$ should have large prime factors, $(p - 1, q - 1)$ should be small, and p and q should have decimal expansions differing in length by a few digits.

As we have remarked, the security of the RSA cryptosystem depends on the difficulty of factoring large integers. In particular, for the RSA cryptosystem, once the modulus n has been factored it is easy to find the decrypting transformation from the encrypting transformation. Note, however, that it may be possible to somehow find the decrypting transformation from the encrypting transformation without factoring n, although this seems unlikely at present.

Attacks on Implementations of the RSA Cryptosystem

After 20 years of scrutiny, a variety of attacks on particular implementations of the RSA cryptosystem have been devised. These attacks show that care must be taken when implementing RSA to avoid particular vulnerabilities. Note that no fundamental vulnerability has been found that would make RSA unsuitable for use as a public key cryptosystem. We will describe a variety of these attacks. The interested reader should consult [Bo99].

Encrypting the same plaintext message with different keys can lead to a successful *Hastad broadcast attack*. For example, when the encryption exponent 3 is used by three different people with different encryption moduli to encrypt the same plaintext message, someone who has the three ciphertext messages produced can recover the original plaintext. In general, it is possible to recover a plaintext message from ciphertext produced by encrypting the message using different RSA encryption keys when sufficiently many copies of the message have been encrypted. This type of attach can even succeed if the original message is altered for each recipient in a way that produces linearly related plaintext. To avoid this vulnerability, different random paddings of the message should be encrypted.

We now describe a vulnerability of RSA found by M. Wiener [Wi90]. He showed that the decrypting exponent d of an RSA cryptosystem with encrypting key (e, n) can be efficiently determined if $n = pq$, p and q are primes with $q < p < 2q$, and the decrypting exponent d is less than $n^{1/4}/3$. (In Chapter 12 we will use the theory of continued

fractions to develop this attack.) This result shows that primes p and q that are not too close together should be used to produce the encrypting modulus and a decrypting exponent d that is relatively large should be used. Although it is customary to first select the encryption key in an RSA cipher, we can make the decrypting exponent large by selecting it first, and then using it to compute the encrypting exponent e.

Disclosing partial information about one of the primes that make up the encrypting modulus n leads to another weakness of the RSA cryptosystem. Suppose that $n = pq$ has m digits. Then knowing the initial $m/4$ or the final $m/4$ digits of p allows n to be efficiently factored. For example, when both p and q have 100 decimal digits, if we know the first 50 or the last 50 digits of p, we will be able to factor n. Details of this partial key disclosure attack can be found in [Co97]. A similar result shows that if we know the last $m/4$ digits of the decrypting exponent d, then we can efficiently find d using $O(e \log e)$ operations. This shows that if the encryption exponent e is small, the decryption exponent d can be found if we know the last 1/4 of its digits.

The final type of attack we mention was discovered by Paul Kocher in 1995 when he was an undergraduate at Stanford University. He demonstrated that the decryption exponent in the RSA cryptosystem can be determined by carefully measuring the time required for the system to perform a series of decryptions. This provides information that can be used to determine the decryption key d. Fortunately, it is easy to devise methods to thwart this attack. For a description of this attack, see [TrWa02] and the article by Kocher [Ko96a].

The widespread acceptance and use of the RSA cryptosystem makes in an inviting target for attack. That only minor vulnerabilities have been found has given people confidence in the practical use of this cryptosystem. This fuels the search for vulnerabilities in this popular cryptosystem.

The Rabin Cryptosystem

Michael Rabin [Ra79] discovered a variant of the RSA cryptosystem for which factorization of the modulus n has almost the same computational complexity as obtaining the decrypting transformation from the encrypting transformation. To describe Rabin's cryptosystem let $n = pq$, where p and q are odd primes, and let b be an integer with $0 \leq b < n$. To encrypt the plaintext message P, we form

$$C \equiv P(P + b) \pmod{n}.$$

We will not discuss the decrypting procedure for Rabin ciphers here, because it relies on some concepts that we have not yet developed (see Exercise 49 in Section 11.1). However, we remark that there are four possible values of P for each ciphertext C such that $C \equiv P(P + b) \pmod{n}$, an ambiguity that complicates the decrypting process. When p and q are known, the decrypting procedure for a Rabin cipher can be carried out rapidly because $O(\log\ n)$ bit operations are needed.

Rabin has shown that if there is an algorithm for decrypting in this cryptosystem, without knowledge of the primes p and q, that requires $f(n)$ bit operations, then there is an algorithm for the factorization of n requiring only $2(f(n) + \log n)$ bit