Eric Roberts
CS91

Handout #12
February 9, 2000

Cryptography

On Friday, I will offer an overview of modern cryptography, focusing on the development of public-key cryptosystems. The attached material provides some background reading about cryptography and consists of two independent sources:

1. A draft chapter from my forthcoming book on *The Intellectual Excitement of Computer Science.* This book is intended for a broad audience and provides some background on codes and codebreaking without going deeply into the mathematics of the process.
2. Excerpts from a report on cryptography prepared by three students in my Sophomore College class at Stanford in 1997. This section describes the RSA coding system, which is one of the principal public-key encryption systems in use today.

Chapter 32
Cryptography

Cryptography, derived from the Greek word κρυπτοσ meaning *hidden,* is the science of creating and decoding secret messages whose meaning cannot be understood by those who intercept the message. In the language of cryptography, the message you are trying to send is called the **plaintext**; the message that you actually send is called the **ciphertext**. Unless your adversaries know the secret of the encoding system, which is usually embodied in some privileged piece of information called a **key**, intercepting the ciphertext should not make it possible for them to discover the original plaintext version of the message. On the other hand, the recipient, who is presumably in possession of the key, can easily translate the ciphertext back into its plaintext counterpart.

History of cryptography

Cryptography has been around in some form or another for most of recorded history. There is evidence to suggest that coded messages were used in ancient Egypt, China, and India, possibly as early as the third millennium B.C., although few details of the cryptographic systems have survived. In Book 6 of the *Iliad,* Homer suggests the existence of a coded message when King Proitos, seeking to have the young Bellerophontes killed, has

> . . . sent him to Lykia, and handed him murderous symbols,
> which he inscribed on a folding tablet, enough to destroy life

Hamlet, of course, leaves Rosencrantz and Guildenstern carrying a similarly dangerous missive, but Hamlet's message is secured by a royal seal. In the *Iliad,* there is nothing to suggest that Bellerophontes cannot see the "murderous symbols," which implies that their meaning must somehow be disguised.

One of the first encryption systems whose details survive was developed by the Greek historian Polybius in the second century B.C. In this system, the letters of the alphabet are arranged to form a 5 x 5 grid called a **Polybius square**, like this.

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

In the Polybius square, each letter is represented by its row and column number. Thus, Pheidippides' message to Sparta, as reported by Herodotus,

THE ATHENIANS BESEECH YOU TO HASTEN TO THEIR AID

can be transmitted as a series of numeric pairs, as follows:

44 23 15 11 44 23 15 33 24 11 33 43 12 15 43 15 15 13 23 54
34 45 44 34 23 11 43 44 15 33 44 34 44 23 15 24 42 11 24 14

The major advantage of the Polybius square is not so much that it allows for secret messages, but that it simplifies the problem of transmission. Each letter in the message can be represented by holding between one and five torches in each hand, which would allow a message to be passed quickly over great distances. By reducing the alphabet to an easily transmittable code, the Polybius square anticipates such later developments as Morse code and semaphore, not to mention modern digital encodings such as ASCII.

In *De Vita Caesarum,* written sometime around 110 A.D., the Roman historian Suetonius describes an encryption system used by Julius Caesar, as follows:

> If he had anything confidential to say, he wrote it in cipher, that is, by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher these, and get at their meaning, he must substitute the fourth letter of the alphabet, namely D, for A, and so with the others.

Even today, the technique of encoding a message by shifting letters a certain distance in the alphabet is called a **Caesar cipher**. According to the passage from Suetonius, each letter is shifted three letters ahead in the alphabet. For example, if Caesar had had time to translate his final words according to his coding system, **ET TU BRUTE** would have come out as **HW WX EUXWH**, because **E** gets moved three letters ahead to **H**, **T** gets moved three to **W**, and so on. Letters that get advanced past the end of the alphabet wrap around back to the beginning, so that **X** would become **A**, **Y** would become **B**, and **Z** would become **C**.

Caesar ciphers have been used into modern times. The "secret decoder rings" that used to be given away as premiums in cereal boxes were often based on the Caesar cipher principle. On the system of electronic bulletin boards called USENET, users could disguise the content of postings that might offend some readers by employing a mode called **ROT13**, in which all letters were cycled forward 13 positions in the alphabet. And the fact that the name of the **HAL** computer in Arthur C. Clarke's *2001* is a one-step Caesar cipher of **IBM** has caused a certain amount of speculation over the years.

Although they are certainly simple, Caesar ciphers are also extremely easy to break. There are, after all, only 25 possible Caesar ciphers for English text. If you wanted to try to break it, all you would have to do is try each of the 25 possibilities and see which one translated the ciphertext message into something readable. A somewhat better scheme is to allow each letter in the plaintext message to be represented by some other letter, but not one that is simply a fixed distance from the original. In this case, the key for the encoding operation is a letter translation table that shows what each of the possible plaintext characters becomes in the ciphertext. Such a coding scheme is called a **letter-substitution cipher.**

Letter-substitution ciphers have been used for many, many years. Examples of such ciphers appear in several works from both classical and medieval times. In the 15th century, an Arabic encyclopedia included a section on cryptography describing various methods for creating ciphers as well as techniques for breaking them. More importantly, this same manuscript includes the first instance of a cipher in which several different coded symbols can stand for the same plaintext character. Codes in which each plaintext letter maps into a single ciphertext equivalent are called **monoalphabetic ciphers**; codes in which each character can have more than one coded representation are called **polyalphabetic ciphers**.

Cryptograms

Today, monoalphabetic ciphers survive primarily in the form of letter-substitution puzzles called **cryptograms.** Edgar Allen Poe was a great fan of cryptograms and included a cryptographic puzzle in the excerpt from *The Gold Bug* shown in Figure 1.

Figure 1. Excerpt from *The Gold Bug* by Edgar Allen Poe (1843)

Here Legrand, having re-heated the parchment, submitted it to my inspection. The following characters were rudely traced, in a red tint, between the death's head and the goat:

53‡‡†305))6*;4826)4‡•)4‡);806*;48†8¶
60))85;1‡(;:‡*8†83(88)5*†;46(;88*96*?
;8)*‡(;485);5*†2:*‡(;4956*2(5*—4)8¶8*
;4069285);)6†8)4‡‡;1(‡9;48081;8:8‡1;
48†85;4)485†528806*81(‡9;48;(88;4(‡
?34;48)4‡;161;:188;‡?;

"But," said I, returning him the slip, "I am as much in the dark as ever. Were all the jewels of Golconda awaiting me upon my solution of this enigma, I am quite sure that I should be unable to earn them."

"And yet, " said Legrand, "the solution is by no means so difficult as you might be led to imagine from the first hasty inspection of the characters. These characters, as any one might readily guess, form a cipher—that is to say, they convey a meaning; but then from what is known of Kidd, I could not suppose him capable of constructing any of the more abstruse cryptographs. I made up my mind, at once, that this was of a simple species—such, however, as would appear to the crude intellect of the sailor, absolutely insoluble without the key."

"And you really solved it?"

"Readily; I have solved others of an abstruseness ten thousand times greater. Circumstances, and a certain bias of mind, have led me to take interest in such riddles, and it may well be doubted whether human ingenuity can construct an enigma of the kind which human ingenuity may not, by proper application, resolve. In fact, having once established connected and legible characters, I scarcely gave a thought to the mere difficulty of developing their import. . . .

"You observe there are no divisions between the words. Had there been divisions the task would have been comparatively easy. In such cases I should have commenced with a collation and analysis of the shorter words, and, had a word of a single letter occurred, as is most likely (**a** or **I**, for example), I should have considered the solution as assured. But, there being no division, my first step was to ascertain the predominant letters, as well as the least frequent. Counting all, I constructed a table thus:

Of the character	**8**	there are	33
	;	"	26
	4	"	19
	‡,)	"	16
	*****	"	13
	5	"	12
	6	"	11
	(	"	10
	†, 1	"	8
	0	"	6
	9, 2	"	5
	:, 3	"	4
	?	"	3
	¶	"	2
	—, •	"	1

"Now, in English, the letter which most frequently occurs is **e**. Afterward, the succession runs thus: **a o i d h n r s t u y c f g l m w b k p q x z**. **E** predominates so remarkably, that an individual sentence of any length is rarely seen, in which it is not the prevailing character.

"Here, then, we have, in the very beginning, the groundwork for something more than a mere guess. The general use which may be made of the table is obvious—but, in this particular cipher, we shall only very partially require its aid. As our predominant character is **8**, we will commence by assuming it as the **e** of the natural alphabet. To verify the supposition, let us observe it the **8** be seen often in couples—for **e** is doubled with great frequency in English—in such words, for example, as **meet, fleet, speed, seen, been, agree,** etc. In the present instance we see it doubled no less than five times, although the cryptograph is brief.

"Let us assume **8**, then, as **e**. Now, of all words in the language, **the** is most usual; let us see, therefore, whether there are not repetitions of any three characters, in the same order of collocation, the last of them being **8**. If we discover a repetition of such letters, so arranged, they will most probably represent the word **the**. Upon inspection, we find no less than seven such arrangements, the characters being **;48**. We may, therefore, assume that **;** represents **t, 4** represents **h,** and **8** represents **e**—the last being now well confirmed. Thus a great step has been taken. . . .

"But, having established a single word, we are enabled to establish a vastly important point; that is to say, several commencements and terminations of other words. Let us refer, for example, to the last instance but one, in which the combination **;48** occurs—not far from the end of the cipher. We know that the **;** immediately ensuing is the commencement of a word, and, of the six characters succeeding this **the**, we are cognizant of no less than five. Let us set these characters down, thus, by the letters we know them to represent, leaving a space for the unknown—**t eeth**.

"Here we are enabled, at once, to discard the **th** as forming no portion of the word commencing with the first **t**; since, by experiment of the entire alphabet for a letter adapted to the vacancy, we perceive that no word can be formed of which this **th** can be a part. We are thus narrowed into **t ee**, and, going through the alphabet, if necessary, as before, we arrive at the word **tree** as the sole possible reading. We thus gain another letter, **r** . . .

"I have said enough to convince you that ciphers of this nature are readily soluble, and to give you some insight into the rationale of their development. But be assured that the specimen before us appertains to the very simplest species of cryptograph. It now only remains to give you the full translation of the characters upon the parchment, as unriddled. Here it is:

A good glass in the bishop's hostel in the devil's seat forty-one degrees and thirteen minutes northeast and by north main branch seventh limb east side shoot from the left eye of the death's-head a bee-line from the tree through the shot fifty feet out.

In this excerpt, Poe not only provides the solution to the cryptogram but provides a general technique for solving monoalphabetic ciphers: calculate the frequency of the letters used in the ciphertext and correlate the appearance of coded sequences with the frequency of letters in English. By guessing that the letters appearing most often in the ciphertext correspond to the most common letters in English, you can usually make a good start toward solving such puzzles.

You might try to apply Poe's strategy to one of the most famous literary cryptograms, which appears in the Sherlock Holmes mystery, *The Dancing Men*, by Sir Arthur Conan Doyle. In this adventure, Holmes receives several messages written in what appears to be "a number of absurd little figures dancing across the page upon which they are drawn." These messages appear in Figure 2 and, of course, did not confound Holmes for very long. If you do attempt to solve this problem, it may help you to know that Poe's list of the most common letters is not in fact correct. The most common letters in English, as verified by computerized analysis, are, in order of decreasing frequency

E T A O I N S H R D L U

Given that statistical studies of English text were by no means as well developed in Poe's day, Poe can perhaps be excused for making a few mistakes.

What Poe did realize is that solving a monoalphabetic cipher does require a strategy. The Caesar cipher, for example, requires one to check only 25 possibilities before the correct plaintext must appear. In the general case of a letter-substitution cipher, there are 26 possible letters to choose as the coded representation for **A**, 25 remaining possible

Figure 2. Messages from Sherlock Holmes *The Dancing Men* by Sir Arthur Conan Doyle

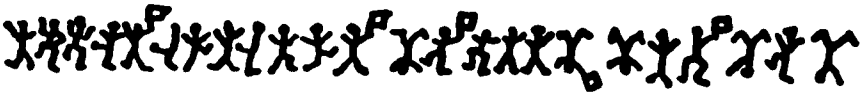
Message 1:

Message 2:

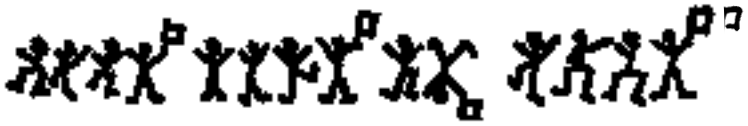
Message 3:

Message 4:

Message 5:



Message 6:



letters to choose as the coded representation for **B**, 24 possibilities for **C**, and so on, for a total of 26x25x24x . . . x3x2x1 possible encodings. This number, which mathematicians call the **factorial** of 26 and write as 26!, is an extremely large number, equal in decimal notation to

403,291,461,126,605,635,584,000,000

This number is far too large to allow brute-force solutions to work for any letter-substitution cipher. Solution strategies for such codes must take account of the specific characteristics of the language used for the plaintext, building up deductions as you go.

Alan Turing and the Enigma machine

One of the most important codebreaking efforts of modern times took place in World War II, when Great Britain—its shipping lanes under attack by German U-boats—undertook a secret project to break the code used by the Germans to communicate with their military force. The British decryption effort was headquartered at the Government Code and Cipher School in Bletchley Park. The leading mathematician on the project was Alan Turing, one of the founders of modern computer science.

The physical Enigma machine, which appears in the photograph on the right, consisted of a keyboard, a panel of indicator lights, and a set of three rotors that were responsible for the encryption. The structure of the machine—prior to several improvements made by the Germans during the war—appears in Figure 3. Each rotor contained 26 contacts on each side connected by wires so that current would be interchanged between pairs of contacts as it flowed through the series of rotors. At the end of the rotor sequence, the contacts from the final rotor were connected to a fixed circuit element called the **reflector**, in which pairs of contacts would again be interchanged.

The operation of the Enigma machine circuits is illustrated in Figure 4. If the user typed an **A** on the keyboard, current would begin to flow on the wire labeled **A** at the right edge of the diagram. The first rotor in the chain would map the **A** to a different contact on the output side of the rotor. Current would then flow on to the next two rotors, through the reflector, and back through all three rotors, ending up on the wire labeled **M**, which would cause the corresponding lamp to light.

If this mechanism were all there were to the Enigma machine, it would implement a simple letter-substitution cipher, which could easily be broken using the techniques for solving cryptograms discussed in the preceding section. What makes the Enigma code challenging is that the encoding changes on every character. After the operator presses a key, the fast rotor advances one notch, thereby changing the interconnections. After the fast rotor completes a full rotation, a notch in the rotor advances the medium rotor, which in turn carries over once per revolution to the slow rotor, in much the same fashion as an odometer on a car. The ever-changing encodings make it difficult to apply standard cryptographic techniques.

Figure 3. Structure of the Enigma machine

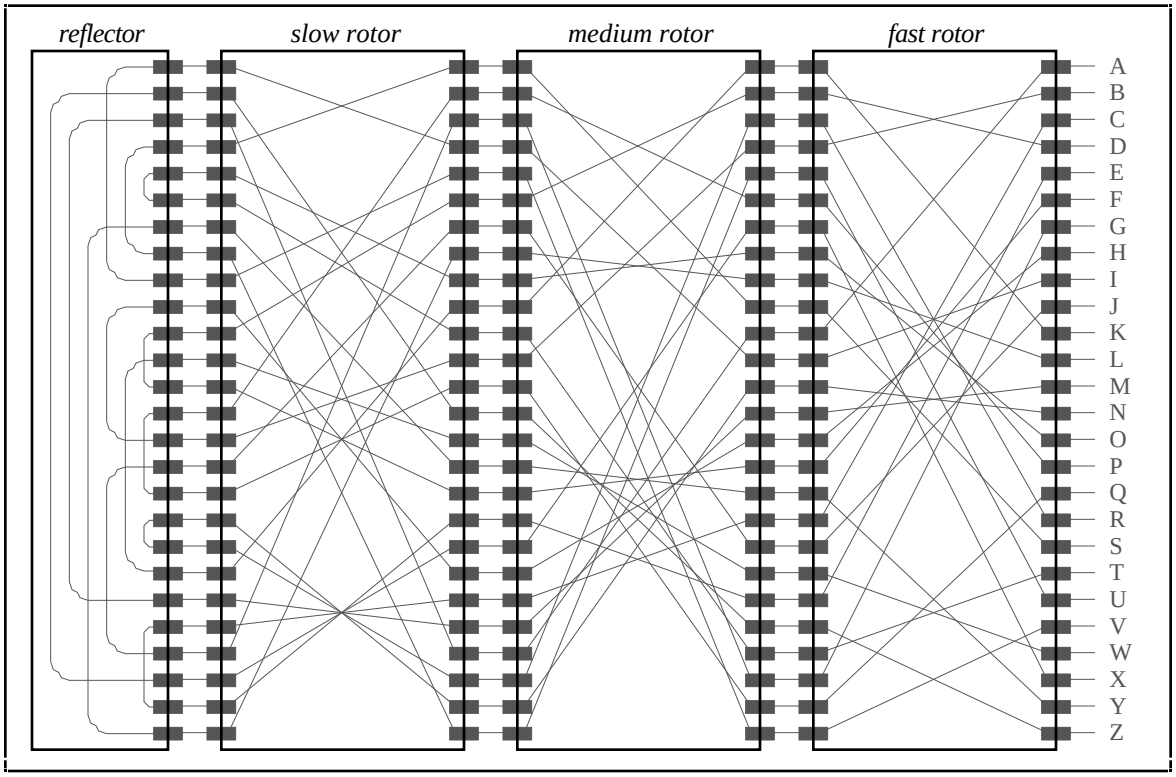
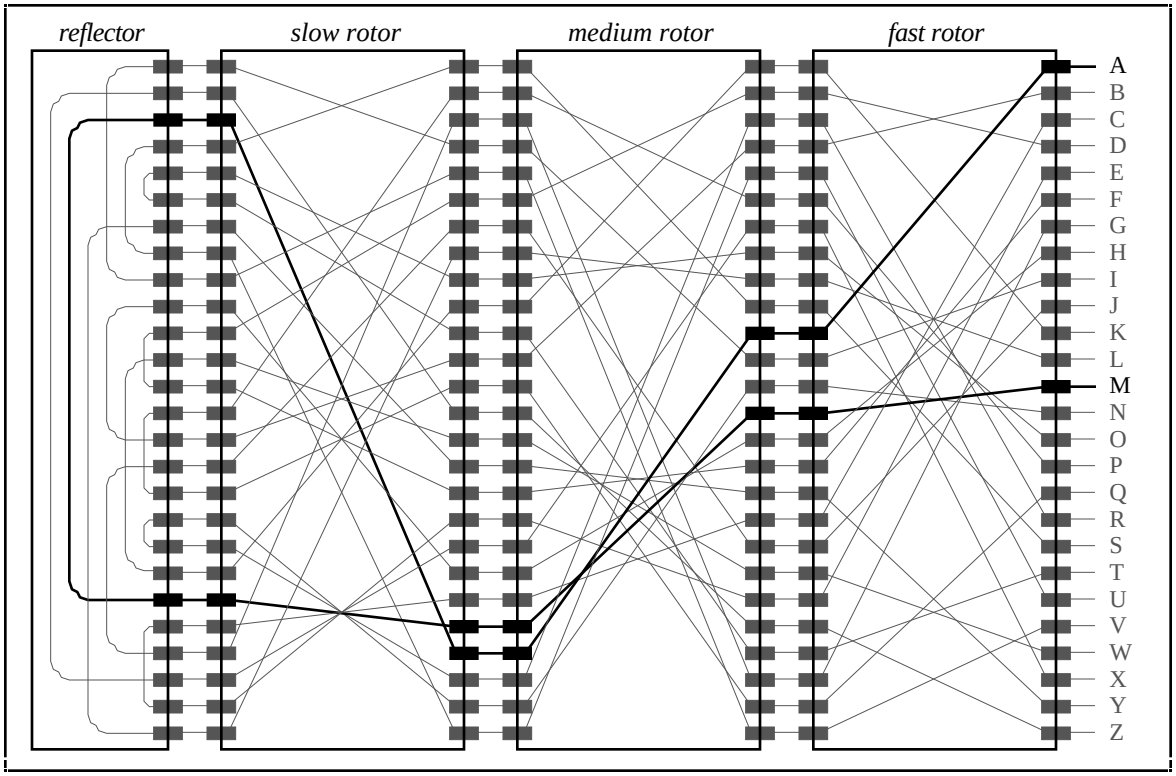


Figure 4. Illustration of current flow after typing A on the Enigma machine



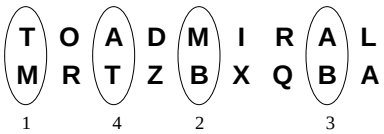
In solving the problem, the British were fortunate in several respects.

1. The Polish underground had managed to capture an Enigma machine and smuggle it to England. The team at Bletchley Park knew exactly how the machine worked, although they did not know the initial settings of the rotors.
2. The German military was quite rigid in its communication style, which allowed the British to anticipate what certain parts of the message looked like. In particular, messages tended to start with a salutation to the receiving general, admiral, or captain in a way that included the full name and title. This fact made it possible to guess, for example, that the first words in an intercepted message might be the German equivalent of **TO ADMIRAL —**.
3. The Germans were convinced that their code was unbreakable, which in turn made them careless. In addition to maintaining a stylized message syntax that allowed the British to guess the plaintext, the Germans periodically reused their code book, which made the code easier to break.

In order to break the Enigma code, Turing and the rest of the Bletchley Park team had to find ways to reduce the enormous number of possible combinations to a manageable size. The technique that they hit upon made use of the following facts about the Enigma machine:

- *The Enigma encoding is symmetrical.* As you can see from the highlighted connection wire in Figure 4, the circuit that transforms **A** into **M** would also transform **M** into **A**, for that particular rotor setting. This property was useful to the Germans, because it meant that the Enigma machine did not need to have separate modes for encoding and decoding. As long as the sending and receiving machines started in the same state, typing in the ciphertext at the receiving station would invert the direction of current flow through the rotor network and regenerate the plaintext.
- *The Enigma machine can never map a character into itself.* Because of its construction and the fundamental symmetry of the transformation, it is never possible to have the letter **A**, for example, come back as the letter **A**.
- *The medium and slow rotors change position for only a small fraction of typed characters.* The medium rotor advances only once every 26 characters; the slow rotor advances once every 26 x 26 (or 676) rotations. Thus, each coded message will contain small groupings of words in which only the fast rotor advances.

Turing was able to exploit these properties of the Enigma machine by taking advantage of cyclical patterns in the relationship between the ciphered text and a guess at the plaintext equivalent, which is called a **crib**. Suppose, for example, that the initial letters of an encrypted message were **MRTZBXQBA**. Given the highly patterned nature of German messages—these letters might correspond to **TOADMIRAL** in the plaintext. The string **TOADMIRAL** represents the crib. If you write the crib together with its encrypted counterpart, you will discover that there is a sequence of four transformations that forms a loop, as indicated by the circled letter pairs:



The **T** in circle 1 is transformed into an **M**. With a different rotor setting, the **M** in circle 2 becomes a **B**. In circle 3, an **A** is shown transformed into a **B**, but the symmetry of the machine implies that a **B** would have been transformed into an **A** with the same rotor position. Finally, circle 4 shows that, in yet another rotor position, the letter **A** becomes a

T. Turing used the term **loop** to refer to such a cycle of transformations between crib and ciphertext that return to the same letter. Even though the position of the fast rotor changes between each letter in the sequence, there are only a small number of rotor settings that can produce a loop of this sort. By assuming that only the fast rotor advanced during this set of letters—which is reasonably likely over a short sequence of letters—Turing was able to eliminate many impossible combinations, thereby reducing the search space substantially. Moreover, once the setting of the fast rotor had been identified, Turing could use a similar technique to determine the setting of the medium rotor, and so on.

The discovery of these loops gave the Bletchley team what they needed to break the Enigma code, but it was still not possible to carry out the necessary computations by hand. To provide the necessary computational power to check all the necessary configurations, Turing built an electromechanical computing device called the **Bombe**, which simulated the operation of the Enigma machine. The Bombe would be programmed to search for feasible rotor positions given a particular set of loops in the encoding of a suspected plaintext into its encrypted version. At each state of the machine's operation, it would begin by assuming that the current setting of the rotors was correct. If that assumption led to a contradiction in terms of violating the invariants required by the loop, the Bombe would quickly move on to the next cycle. Although the running time depended on the number of loops detected in the crib/ciphertext pairing, the Bombe was able to search through all possible rotor combinations in 10-20 hours.

Cryptography group
Jed Burgess, Murat Goksel, and Erin Pattison

Sophomore College
September 1997

The RSA Algorithm

Before 1974 secure communication was only possible by using private-key cryptography. This method implied the following features:

- Anyone who can encipher a message is also able to decipher it
- Sender and receiver share a common secret key which must be transmitted before encryption procedure can start, usually by some physical means like special couriers, etc.

Such a private-key system is a symmetric system since the same machine (actually private key) can be used to enciphering and deciphering. Yet this is not true for asymmetric systems like public-key cryptography.

What are public-key cryptosystems?

For this kind of secure communications, any participant P has one pair of keys,

- A public key E_P
- A private key D_P for deciphering with the property that it is not feasible to compute D_P from E_P.

All public keys are publicly available; they might be stored in a public file. Other hand, the private keys are kept secret; they are known only to their owners.

A public-key cryptosystem is called a public-key encryption scheme, if for any message *m* we have

$$D(E(m)) = m$$

A public-key cryptosystem, is called a public-key signature scheme, if for any message *m* one can verify using the public key E that *m* and D(*m*) fit.

Let us now make an example of how two parties can communicate securely over electronic networks.

1. If A wants to send the message *m* to B, A
   - looks up the public key E_B of B
   - enciphers the message *m* using E_B
   - sends $E_B(m)$ to B
2. B is able to decipher the ciphertext $E_B(m)$, since he exclusively knows the key D_B:

   $$D_B(E_B(m)) = m$$
3. No other participant can decipher $E_B(m)$, since, by hypothesis, no one can deduce D_B from E_B (and $E_B(m)$).

In order to explain the beauty of public-key crytosystems we will use an everyday analogy. Suppose that each participant of our imaginary communications system has a mailbox with his name on it and his own private key. Applying the public key corresponds to inserting a letter to the mailbox. The individual’s keys correspond to the secret keys.

If someone wants to send a message to Mrs. Johnson, he simply puts the message into Mrs. Johnson's mailbox. Note how the analogy respects the basic property of a public-key system: knowing how to deliver the message does not make the inverse any easier at all. Now anyone, even the sender, may try his key on the mailbox—without any success! Only Mrs. Johnson may open the lock—and she may do so without any difficulty.

What are some advantages?

- *No key exchange among the participants is necessary.* This solves the fundamental problem of private-key systems. The important consequence is that public-key systems offer spontaneous communications that's more secure communication! I am in a position to send secret messages to an agent or offshore bank in the Cayman Islands without any need to have previously agreed upon a secret key.
- *New participants can join the system without any new problems for the old members.* If a new participant joins a symmetric cryptosystem, all three users have to exchange a secret key with him. In this asymmetric system, the old members do not have to update their databases. In short, public-key systems facilitate secure communications and enable us to communicate more.
- *Public-key mechanisms offer an excellent possibility for digital signatures.*

How the RSA algorithm works

In order to grasp how the RSA algorithm works, we need to understand a theorem proposed by Leonard Euler, a Swiss mathematician. First some definitions. For a natural number n, we define $\phi(n)$ to be the number of positive integers smaller than or equal to n that have no factor except 1 in common with n. In other words, we search for all positive integers smaller than n that are relatively prime to n—their greatest common divisor with n equals 1. This sounds more complicated than it actually is. Here are some examples:

$$\phi(1) = 1,\ \phi(2) = 1,\ \phi(3) = 2,\ \phi(4) = 2,\ \phi(6) = 2,\ \phi(10) = 4,\ \phi(15) = 8$$

We can see the last assertions as follows: The positive integers < 15 that are relatively prime to 15 are 1, 2, 4, 7, 8, 11, 13, and 14. Since there are 8 different numbers, we have $\phi(15) = 8$.

There are also some properties of $\phi(n)$:

1. If p denotes a prime number, then $\phi(p) = p - 1$, because all the $p - 1$ integers 1, 2, 3, up to $p - 1$ are relatively prime to p.
2. If p and q are two distinct primes then, $\phi(pq) = (p–1)(q–1)$. This is a particularly important property. There are a total of $pq–1$ positive integers smaller than pq. We count how many of them are not relatively prime to pq. On the one hand, these are the $q–1$ multiples of p, namely

   $$p, 2p, 3p, \ldots\ldots, (q–1)p,$$

   and on the other hand the $p–1$ multiples

   $$q, 2q, 3q, \ldots\ldots, (p–1)q$$

   of q. Since there are the only integers between 1 and $pq–1$ that are not relatively prime to pq, it follows that

   $$\phi(pq) = pq –1 – (q–1) – (p–1) = pq–q–p+1 = (p–1)(q–1)$$

Now we can formulate Euler's Theorem. Denote by m and n two relatively prime positive integers. Then

$$m^{\phi(n)} \bmod n = 1.$$

We shall be particularly interested in the case, when n is the product of two distinct primes. In the view of the second claim, Euler's Theorem reads as follows:

> Let m be an integer that has no common factor except 1 with either of the two distinct primes p and q. Then
>
> $$m^{(p-1)(q-1)} \bmod pq = 1$$

In other words, if the large number $m^{(p-1)(q-1)}$ is divided by pq, it leaves 1 as its remainder.

We won't spend time proving this theorem. However, we will show a couple of examples. If p is 2 and q is 3, the result

$$5^{\phi(6)} \bmod 6 = 5^2 \bmod 6 = 25 \bmod 6 = 1$$

can easily be checked. For larger values, such as $p = 23$ and $q = 37$, verifying that

$$31^{\phi(23x37)} \bmod (23x37) = 31^{\phi(851)} \bmod 851 = 1$$

would require considerably more effort.

The RSA algorithm relies on the following facts as well:

1. It is extremely difficult to factor a large number.
2. It is, however, easy to calculate the greatest common divisor (gcd) of two large numbers using Euclid's algorithm, which can be expressed in C like this:

```
int GCD(int x, int y)
{
    int r;

    while (TRUE) {
        r = x % y;
        if (r == 0) break;
        x = y;
        y = r;
    }
    return (y);
}
```

3. If two integers a and b are relatively prime (that is, their gcd is 1), it is easy to find an integer c satisfying

   $$bc \bmod a = 1$$

   This may be restated as: c is the inverse of b modulo a. The algorithm needed to find c is similar in its operation to Euclid's algorithm.

Key generation

Before we describe how the RSA algorithm works, we must clarify the hypotheses for the system. First, every participant must get a pair of keys. A key center chooses two distinct large primes p and q and multiplies them:

$$n = pq$$

Then the center computes

$$\phi(n) = \phi(pq) = (p–1)(q–1)$$

Finally, the center computes two integers d and e with

$$ed \bmod \phi(n) = 1$$

The participant gets e and n as his public-key and d as his private key.

Using the RSA algorithm to send messages

If somebody wants to send a message to Mrs. Johnson, he first must learn Johnson's public key. This public key consists of the modulus n and the exponent e. Next the message must have the form of one or more nonnegative integers $m < n$. There are many ways to achieve this; one method follows. Suppose that the message consists of letters, numbers and special characters. Each character is represented by its own arrangement of eight bits (zeros and ones); most computers use a standard system called ASCII. So if n has 512 bits, one forms groups of 64 characters each, encodes these characters and gets strings of 64 x 8 = 512 bits. These bit strings are then interpreted as the binary representations of some numbers.

We shall assume that each segment of the message is a positive integer $m < n$. One enciphers such a number m by raising it to the eth power and reducing the result modulo n. In other words, if

$$c = m^e \bmod n$$

then c is the ciphertext corresponding to the clear text m.

How does one decipher c? This has been arranged in such a way that only one person can do it, namely the recipient Mrs. Johnson. She simply applies her private key to the cipher text c. More precisely, the number

$$m' = c^d \bmod n$$

is the message Mrs. Johnson gets by deciphering c. This leaves one obvious question. Mrs. Johnson does not care about an arbitrary message m'; she wants to get the original message m. Is $m' = m$? The following theorem asserts that this is always the case.

> For $c = m^e \bmod n$ and $m' = c^d \bmod n$, provided that $ed \bmod \phi(n) = 1$ and $m < n$, we have $m' = m$.

In other words, the above deciphering algorithm works correctly.

Proof of the RSA algorithm

We know that

$$m' = c^d \bmod n = (m^e)^d \bmod n = m^{de} \bmod n$$

Moreover, since $ed \bmod \phi(n) = 1$, it must be the case that

$$ed = k\phi(n) + 1$$

for some integer k. Thus,

$$m' = m^{de} \bmod n = m^{k\phi(n)+1} \bmod n = m^{k\phi(n)}m \bmod n = (m^{\phi(n)})^k m \bmod n$$

But, by Euler's theorem,

$$m^{\phi(n)} \bmod n = 1$$

So,

$$m' = (m^{\phi(n)})^k \bmod n = 1^k m \bmod n = m \bmod n$$

But, if $m < n$, $m \bmod n$ is simply m, which allows us to conclude that

$$m' = m$$

Are public-key algorithms safe enough?

This question is probably the greatest threat to the RSA algorithm. Is it possible to decipher the message without using the private key?

Suppose we know the public key—that is, the numbers e and n. How could we compute from this private key d? Naturally, if we knew $\phi(n)$, then we could compute d; we could proceed as the key generation center and compute $\phi(n) = (p-1)(q-1)$. Interestingly enough, the converse is also true. If we knew n and $\phi(n)$, then we could factor n. To sum up, if n is the product of two distinct primes, then computing $\phi(n)$ is the same as factoring n.

So everything would be very easy if we could factor n—and there's the rub. Of course, to the casual eye the obstacle is not immediately obvious; it is a popular belief that since there is no difficulty in factoring integers like 72, 123, or 221, it necessarily follows that factoring larger integers remains relatively easy. But the factorization of even rather small integers like 1763 or 8633 (each a product of a prime) imposes considerable difficulties for a human brain. Considering that the n appearing in the RSA algorithm has a suggested length of about 200 decimal digits, one gets a strong feeling that factoring such large integers is an extremely difficult problem.