FROM
"THE CODE BOOK"
BY
SIMON SINGH

6 Alice and Bob Go Public

During the Second World War, British codebreakers had the upper hand over German codemakers, mainly because the men and women at Bletchley Park, following the lead of the Poles, developed some of the earliest codebreaking technology. In addition to Turing's bombes, which were used to crack the Enigma cipher, the British also invented another codebreaking device, Colossus, to combat an even stronger form of encryption, namely the German Lorenz cipher. Of the two types of codebreaking machine, it was Colossus that would determine the development of cryptography during the latter half of the twentieth century.

The Lorenz cipher was used to encrypt communications between Hitler and his generals. The encryption was performed by the Lorenz SZ40 machine, which operated in a similar way to the Enigma machine, but the Lorenz was far more complicated, and it provided the Bletchley codebreakers with an even greater challenge. However, two of Bletchley's codebreakers, John Tiltman and Bill Tutte, discovered a weakness in the way that the Lorenz cipher was used, a flaw that Bletchley could exploit and thereby read Hitler's messages.

Breaking the Lorenz cipher required a mixture of searching, matching, statistical analysis and careful judgment, all of which was beyond the technical abilities of the bombes. The bombes were able to carry out a specific task at high speed, but they were not flexible enough to deal with the subtleties of Lorenz. Lorenz-encrypted messages had to be broken by hand, which took weeks of painstaking effort, by which time the messages were largely out of date. Eventually, Max Newman, a Bletchley mathematician, came up with a way to mechanize the cryptanalysis of the Lorenz cipher. Drawing heavily on Alan Turing's concept of the universal machine,

Newman designed a machine that was capable of adapting itself to different problems, what we today would call a programmable computer.

Implementing Newman's design was deemed technically impossible, so Bletchley's senior officials shelved the project. Fortunately, Tommy Flowers, an engineer who had taken part in discussions about Newman's design, decided to ignore Bletchley's skepticism, and went ahead with building the machine. At the Post Office's research center at Dollis Hill, North London, Flowers took Newman's blueprint and spent ten months turning it into the Colossus machine, which he delivered to Bletchley Park on December 8, 1943. It consisted of 1,500 electronic valves, which were considerably faster than the sluggish electromechanical relay switches used in the bombes. But more important than Colossus's speed was the fact that it was programmable. It was this fact that made Colossus the precursor to the modern digital computer.

Colossus, as with everything else at Bletchley Park, was destroyed after the war, and those who worked on it were forbidden to talk about it. When Tommy Flowers was ordered to dispose of the Colossus blueprints, he obediently took them down to the boiler room and burned them. The plans for the world's first computer were lost forever. This secrecy meant that other scientists gained the credit for the invention of the computer. In 1945, J. Presper Eckert and John W. Mauchly of the University of Pennsylvania completed ENIAC (Electronic Numerical Integrator And Calculator), consisting of 18,000 electronic valves, capable of performing 5,000 calculations per second. For decades, ENIAC, not Colossus, was considered the mother of all computers.

Having contributed to the birth of the modern computer, cryptanalysts continued after the war to develop and employ computer technology in order to break all sorts of ciphers. They could now exploit the speed and flexibility of programmable computers to search through all possible keys until the correct one was found. In due course, the cryptographers began to fight back, exploiting the power of computers to create increasingly complex ciphers. In short, the computer played a crucial role in the postwar battle between codemakers and codebreakers.

Using a computer to encipher a message is, to a large extent, very similar to traditional forms of encryption. Indeed, there are only three significant differences between computer encryption and the sort of

mechanical encryption that was the basis for ciphers like Enigma. The first difference is that a mechanical cipher machine is limited by what can be practically built, whereas a computer can mimic a hypothetical cipher machine of immense complexity. For example, a computer could be programmed to mimic the action of a hundred scramblers, some spinning clockwise, some anticlockwise, some vanishing after every tenth letter, others rotating faster and faster as encryption progresses. Such a mechanical machine would be practically impossible to build, but its "virtual" computerized equivalent would deliver a highly secure cipher.

The second difference is simply a matter of speed. Electronics can operate far more quickly than mechanical scramblers: a computer programmed to mimic the Enigma cipher could encipher a lengthy message in an instant. Alternatively, a computer programmed to perform a vastly more complex form of encryption could still accomplish the task within a reasonable time.

The third, and perhaps most significant, difference is that a computer scrambles numbers rather than letters of the alphabet. Computers deal only in binary numbers–sequences of ones and zeros known as *binary digits*, or *bits* for short. Before encryption, any message must therefore be converted into binary digits. This conversion can be performed according to various protocols, such as the American Standard Code for Information Interchange, known familiarly by the acronym ASCII, pronounced "ass-key." ASCII assigns a 7-digit binary number to each letter of the alphabet. For the time being, it is sufficient to think of a binary number as merely a pattern of ones and zeros that uniquely identifies each letter (Table 24), just as Morse code identifies each letter with a unique series of dots and dashes. There are 128 (2^7) ways to arrange a combination of 7 binary digits, so ASCII can identify up to 128 distinct characters. This allows plenty of room to define all the lowercase letters (e.g., a = 1100001), all necessary punctuation (e.g., ! = 0100001), as well as other symbols (e.g., & = 0100110). Once the message has been converted into binary, encryption can begin.

Even though we are dealing with computers and numbers, and not machines and letters, the encryption still proceeds by the age-old principles of substitution and transposition, in which elements of the message are substituted for other elements, or their positions are switched, or both. Every encipherment, no matter how complex, can be broken down into

combinations of these simple operations. The following two examples demonstrate the essential simplicity of computer encipherment by showing how a computer might perform an elementary substitution cipher and an elementary transposition cipher.

First, imagine that we wish to encrypt the message **HELLO**, employing a simple computer version of a transposition cipher. Before encryption can begin, we must translate the message into ASCII according to Table 24:

Plaintext = **HELLO** = 1001000 1000101 1001100 1001100 1001111

One of the simplest forms of transposition cipher would be to swap the first and second digits, the third and fourth digits, and so on. In this case the final digit would remain unchanged because there are an odd number of digits. In order to see the operation more clearly, I have removed the spaces between the ASCII blocks in the original plaintext to generate a single string, and then lined it up against the resulting ciphertext for comparison:

Plaintext = 1001000100010110011001001100100111
Ciphertext = 0110001000101001100110001100011 0111

An interesting aspect of transposition at the level of binary digits is that the transposing can happen within the letter. Furthermore, bits of one letter can swap places with bits of the neighboring letter. For example, by swapping

Table 24 ASCII binary numbers for the capital letters.

A	1000001	N	1001110
B	1000010	O	1001111
C	1000011	P	1010000
D	1000100	Q	1010001
E	1000101	R	1010010
F	1000110	S	1010011
G	1000111	T	1010100
H	1001000	U	1010101
I	1001001	V	1010110
J	1001010	W	1010111
K	1001011	X	1011000
L	1001100	Y	1011001
M	1001101	Z	1011010

the seventh and eighth numbers, the final 0 of H is swapped with the initial 1 of E. The encrypted message is a single string of 35 binary digits, which can be transmitted to the receiver, who then reverses the transposition to re-create the original string of binary digits. Finally, the receiver reinterprets the binary digits via ASCII to regenerate the message **HELLO**.

Next, imagine that we wish to encrypt the same message, **HELLO**, this time employing a simple computer version of a substitution cipher. Once again, we begin by converting the message into ASCII before encryption. As usual, substitution relies on a key that has been agreed between sender and receiver. In this case the key is the word **DAVID** translated into ASCII, and it is used in the following way. Each element of the plaintext is "added" to the corresponding element of the key. Adding binary digits can be thought of in terms of two simple rules. If the elements in the plaintext and the key are the same, the element in the plaintext is substituted for 0 in the ciphertext. But, if the elements in the message and key are different, the element in the plaintext is substituted for 1 in the ciphertext:

Message	**HELLO**
Message in ASCII	1001000100010110011001001100100111
Key = DAVID	1000100100000110101101001001100010 0
Ciphertext	0001100000010000110100000101000101 1

The resulting encrypted message is a single string of 35 binary digits which can be transmitted to the receiver, who uses the same key to reverse the substitution, thus recreating the original string of binary digits. Finally, the receiver reinterprets the binary digits via ASCII to regenerate the message **HELLO**.

Computer encryption was restricted to those who had computers, which in the early days meant the government and the military. However, a series of scientific, technological and engineering breakthroughs made computers, and computer encryption, far more widely available. In 1947, AT&T Bell Laboratories invented the transistor, a cheap alternative to the electronic valve. Commercial computing became a reality in 1951 when companies such as Ferranti began to make computers to order. In 1953 IBM launched its first computer, and four years later it introduced Fortran, a programming language that allowed "ordinary" people to write

computer programs. Then, in 1959, the invention of the integrated circuit heralded a new era of computing.

During the 1960s, computers became more powerful, and at the same time they became cheaper. Businesses were increasingly able to afford computers, and could use them to encrypt important communications such as money transfers or delicate trade negotiations. However, as more and more businesses bought computers, and as encryption between businesses spread, cryptographers were confronted with new problems, difficulties that had not existed when cryptography was the preserve of governments and the military. One of the primary concerns was the issue of standardization. A company might use a particular encryption system to ensure secure internal communication, but it could not send a secret message to an outside organization unless the receiver used the same system of encryption. Eventually, on May 15, 1973, America's National Bureau of Standards planned to solve the problem, and formally requested proposals for a standard encryption system that would allow business to speak secretly unto business.

One of the more established cipher algorithms, and a candidate for the standard, was an IBM product known as Lucifer. It had been developed by Horst Feistel, a German émigré who had arrived in America in 1934. He was on the verge of becoming a U.S. citizen when America entered the war, which meant that he was placed under house arrest until 1944. For some years after, he suppressed his interest in cryptography to avoid arousing the suspicions of the American authorities. When he did eventually begin research into ciphers, at the Air Force's Cambridge Research Center, he soon found himself in trouble with the National Security Agency (NSA), the organization with overall responsibility for maintaining the security of military and governmental communications, and which also attempts to intercept and decipher foreign communications. The NSA employs more mathematicians, buys more computer hardware, and intercepts more messages than any other organization in the world. It is the world leader when it comes to snooping.

The NSA did not object to Feistel's past, they merely wanted to have a monopoly on cryptographic research, and it seems that they arranged for Feistel's research project to be canceled. In the 1960s Feistel moved to the Mitre Corporation, but the NSA continued to apply pressure and forced

him to abandon his work for a second time. Feistel eventually ended up at IBM's Thomas J. Watson Laboratory near New York, where for several years he was able to conduct his research without being harassed. It was there, during the early 1970s, that he developed the Lucifer system.

Lucifer encrypts messages according to the following scrambling operation. First, the message is translated into a long string of binary digits. Second, the string is split into blocks of 64 digits, and encryption is performed separately on each of the blocks. Third, focusing on just one block, the 64 digits are shuffled, and then split into two half-blocks of 32, labeled Left^0 and Right^0. The digits in Right^0 are then put through a "mangler function," which changes the digits according to a complex substitution. The mangled Right^0 is then added to Left^0 to create a new half-block of 32 digits called Right^1. The original Right^0 is relabeled Left^1. This set of operations is called a "round." The whole process is repeated in a second round, but starting with the new half-blocks, Left^1 and Right^1, and ending with Left^2 and Right^2. This process is repeated until there have been 16 rounds in total. The encryption process is a bit like kneading a slab of dough. Imagine a long slab of dough with a message written on it. First, the long slab is divided into blocks that are 64 cm in length. Then, one half of one of the blocks is picked up, mangled, folded over, added to the other half and stretched to make a new block. Then the process is repeated over and over again until the message has been thoroughly mixed up. After 16 rounds of kneading the ciphertext is sent, and is then deciphered at the other end by reversing the process.

The exact details of the mangler function can change, and are determined by a key agreed by sender and receiver. In other words, the same message can be encrypted in a myriad of different ways depending on which key is chosen. The keys used in computer cryptography are simply numbers. Hence, the sender and receiver merely have to agree on a number in order to decide the key. Thereafter, encryption requires the sender to input the key number and the message into Lucifer, which then outputs the ciphertext. Decryption requires the receiver to input the same key number and the ciphertext into Lucifer, which then outputs the original message.

Lucifer was generally held to be one of the strongest commercially available encryption products, and consequently it was used by a variety of organizations. It seemed inevitable that this encryption system would

be adopted as the American standard, but once again the NSA interfered with Feistel's work. Lucifer was so strong that it offered the possibility of an encryption standard that was probably beyond the codebreaking capabilities of the NSA; not surprisingly, the NSA did not want to see an encryption standard that they could not break. Hence, it is rumored that the NSA lobbied to weaken one aspect of Lucifer, the number of possible keys, before allowing it to be adopted as the standard.

The number of possible keys is one of the crucial factors determining the strength of any cipher. A cryptanalyst trying to decipher an encrypted message could attempt to check all possible keys, and the greater the number of possible keys, the longer it will take to find the right one. If there are only 1,000,000 possible keys, a cryptanalyst could use a powerful computer to find the correct one in a matter of minutes, and thereby decipher an intercepted message. However, if the number of possible keys is large enough, finding the correct key becomes impractical. If Lucifer were to become the encryption standard, then the NSA wanted to ensure that it operated with only a restricted number of keys.

The NSA argued in favor of limiting the number of keys to roughly 100,000,000,000,000,000 (technically referred to as 56 bits, because this number consists of 56 digits when written in binary). It seems that the NSA believed that such a key would provide security within the civilian community, because no civilian organization had a computer powerful enough to check every possible key within a reasonable amount of time. However, the NSA itself, with access to the world's greatest computing resource, would just about be able to break into messages. The 56-bit version of Feistel's Lucifer cipher was officially adopted on November 23, 1976, and was called the Data Encryption Standard (DES). A quarter of a century later, DES remains America's official standard for encryption.

The adoption of DES solved the problem of standardization, encouraging businesses to use cryptography for security. Furthermore, DES was strong enough to guarantee security against attacks from commercial rivals. It was effectively impossible for a company with a civilian computer to break into a DES-encrypted message because the number of possible keys was sufficiently large. Unfortunately, despite standardization and despite the strength of DES, businesses still had to deal with one more major issue, a problem known as *key distribution.*

Imagine that a bank wants to send some confidential data to a client via a telephone line, but is worried that there might be somebody tapping the wire. The bank picks a key and uses DES to encrypt the data message. In order to decrypt the message, the client needs not only to have a copy of DES on its computer, but also to know which key was used to encrypt the message. How does the bank inform the client of the key? It cannot send the key via the telephone line, because it suspects that there is an eavesdropper on the line. The only truly secure way to send the key is to hand it over in person, which is clearly a time-consuming task. A less secure but more practical solution is to send the key via a courier. In the 1970s, banks attempted to distribute keys by employing special dispatch riders who had been vetted and who were among the company's most trusted employees. These dispatch riders would race across the world with padlocked briefcases, personally distributing keys to everyone who would receive messages from the bank over the next week. As business networks grew in size, as more messages were sent, and as more keys had to be delivered, the banks found that this distribution process became a horrendous logistical nightmare, and the overhead costs became prohibitive.

The problem of key distribution has plagued cryptographers throughout history. For example, during the Second World War the German High Command had to distribute the monthly book of day keys to all its Enigma operators, which was an enormous logistical problem. Also, U-boats, which tended to spend extended periods away from base, had to somehow obtain a regular supply of keys. In earlier times, users of the Vigenère cipher had to find a way of getting the keyword from the sender to the receiver. No matter how secure a cipher is in theory, in practice it can be undermined by the problem of key distribution.

To some extent, government and the military have been able to deal with the problem of key distribution by throwing money and resources at it. Their messages are so important that they will go to any lengths to ensure secure key distribution. The U.S. Government keys are managed and distributed by COMSEC, short for Communications Security. In the 1970s, COMSEC was responsible for transporting tons of keys every day. When ships carrying COMSEC material came into dock, crypto-custodians would march onboard, collect stacks of cards, paper tapes,

floppy disks, or whatever other medium the keys might be stored on, and then deliver them to the intended recipients.

Key distribution might seem a mundane issue, but it became the overriding problem for postwar cryptographers. If two parties wanted to communicate securely, they had to rely on a third party to deliver the key, and this became the weakest link in the chain of security. The dilemma for businesses was straightforward–if governments with all their money were struggling to guarantee the secure distribution of keys, then how could civilian companies ever hope to achieve reliable key distribution without bankrupting themselves?

Despite claims that the problem of key distribution was unsolvable, a team of mavericks triumphed against the odds and came up with a brilliant solution in the mid-1970s. They devised an encryption system that appeared to defy all logic. Although computers transformed the implementation of ciphers, the greatest revolution in twentieth-century cryptography has been the development of techniques to overcome the problem of key distribution. Indeed, this breakthrough is considered to be the greatest cryptographic achievement since the invention of the monoalphabetic cipher, over two thousand years ago.

God Rewards Fools

Whitfield Diffie is one of the most ebullient cryptographers of his generation. The mere sight of him creates a striking and somewhat contradictory image. His impeccable suit reflects the fact that for most of the 1990s he has been employed by one of America's giant computer companies–currently his official job title is Distinguished Engineer at Sun Microsystems. However, his shoulder-length hair and long white beard betray the fact that his heart is still stuck in the 1960s. He spends much of his time in front of a computer workstation, but he looks as if he would be equally comfortable in a Bombay ashram. Diffie is aware that his dress and personality can have quite an impact on others, and comments that, "People always think that I am taller than I really am, and I'm told it's the Tigger effect–'No matter his weight in pounds, shillings and ounces, he always seems bigger because of the bounces.'"

Diffie was born in 1944, and spent most of his early years in Queens, New York. As a child he became fascinated by mathematics, reading books ranging from *The Chemical Rubber Company Handbook of Mathematical Tables* to G.H. Hardy's *Course of Pure Mathematics*. He went on to study mathematics at the Massachusetts Institute of Technology, graduating in 1965. He then took a series of jobs related to computer security, and by the early 1970s he had matured into one of the few truly independent security experts, a freethinking cryptographer, not employed by the government or by any of the big corporations. In hindsight, he was the first cypherpunk.

Diffie was particularly interested in the key distribution problem, and he realized that whoever could find a solution would go down in history as one of the all-time great cryptographers. Diffie was so captivated by the

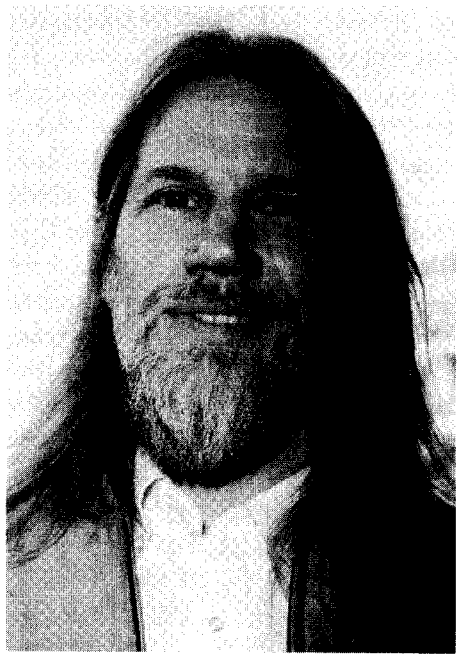

Figure 62 Whitfield Diffie.

problem of key distribution that it became the most important entry in his special notebook entitled "Problems for an Ambitious Theory of Cryptography." Part of Diffie's motivation came from his vision of a wired world. Back in the 1960s, the U.S. Department of Defense began funding a cutting-edge research organization called the Advanced Research Projects Agency (ARPA), and one of ARPA's front-line projects was to find a way of connecting military computers across vast distances. This would allow a computer that had been damaged to transfer its responsibilities to another one in the network. The main aim was to make the Pentagon's computer infrastructure more robust in the face of nuclear attack, but the network would also allow scientists to send messages to each other, and perform calculations by exploiting the spare capacity of remote computers. The ARPANet was born in 1969, and by the end of the year there were four connected sites. The ARPANet steadily grew in size, and in 1982 it spawned the Internet. At the end of the 1980s, nonacademic and nongovernmental users were given access to the Internet, and thereafter the number of users exploded. Today, more than a hundred million people use the Internet to exchange information and send electronic mail messages, or e-mails.

While the ARPANet was still in its infancy, Diffie was farsighted enough to forecast the advent of the information superhighway and the digital revolution. Ordinary people would one day have their own computers, and these computers would be interconnected via phone lines. Diffie believed that if people then used their computers to exchange e-mails, they deserved the right to encrypt their messages in order to guarantee their privacy. However, encryption required the secure exchange of keys. If governments and large corporations were having trouble coping with key distribution, then the public would find it impossible, and would effectively be deprived of the right to privacy.

Diffie imagined two strangers meeting via the Internet, and wondered how they could send each other an encrypted message. He also considered the scenario of a person wanting to buy a commodity on the Internet. How could that person send an e-mail containing encrypted credit card details so that only the Internet retailer could decipher them? In both cases, it seemed that the two parties needed to share a key, but how could they securely exchange keys? The number of casual contacts

and the amount of spontaneous e-mails among the public would be enormous, and this would mean that key distribution would be impractical. Diffie was fearful that the necessity of key distribution would prevent the public from having access to digital privacy, and he became obsessed with the idea of finding a solution to the problem.

In 1974, Diffie, still an itinerant cryptographer, paid a visit to IBM's Thomas J. Watson Laboratory, where he had been invited to give a talk. He spoke about various strategies for attacking the key distribution problem, but all his ideas were very tentative, and his audience was skeptical about the prospects for a solution. The only positive response to Diffie's presentation was from Alan Konheim, one of IBM's senior cryptographic experts, who mentioned that someone else had recently visited the laboratory and given a lecture that addressed the issue of key distribution. That speaker was Martin Hellman, a professor from Stanford University in California. That evening Diffie got in his car and began the 5,000 km journey to the West Coast to meet the only person who seemed to share his obsession. The alliance of Diffie and Hellman would become one of the most dynamic partnerships in cryptography.

Martin Hellman was born in 1945 in a Jewish neighborhood in the Bronx, but at the age of four his family moved to a predominantly Irish Catholic neighborhood. According to Hellman, this permanently changed his attitude to life: "The other kids went to church and they learned that the Jews killed Christ, so I got called 'Christ killer.' I also got beat up. To start with, I wanted to be like the other kids, I wanted a Christmas tree and I wanted Christmas presents. But then I realized that I couldn't be like all the other kids, and in self-defense I adopted an attitude of 'Who would want to be like everybody else?'" Hellman traces his interest in ciphers to this enduring desire to be different. His colleagues had told him he was crazy to do research in cryptography, because he would be competing with the NSA and their multibillion-dollar budget. How could he hope to discover something that they did not know already? And if he did discover anything, the NSA would classify it.

Just as Hellman was beginning his research, he came across *The Codebreakers* by the historian David Kahn. This book was the first detailed discussion of the development of ciphers, and as such it was the perfect primer for a budding cryptographer. *The Codebreakers* was Hellman's only

research companion, until September 1974, when he received an unexpected phone call from Whitfield Diffie, who had just driven across the Continent to meet him. Hellman had never heard of Diffie, but grudgingly agreed to a half-hour appointment later that afternoon. By the end of the meeting, Hellman realized that Diffie was the best-informed person he had ever met. The feeling was mutual. Hellman recalls: "I'd promised my wife I'd be home to watch the kids, so he came home with me and we had dinner together. He left at around midnight. Our personalities are very different–he is much more counterculture than I am–but eventually the personality clash was very symbiotic. It was just such a breath of fresh air for me. Working in a vacuum had been really hard."

Since Hellman did not have a great deal of funding, he could not afford to employ his new soulmate as a researcher. Instead, Diffie was enrolled as a graduate student. Together, Hellman and Diffie began to study the key distribution problem, desperately trying to find an alternative to the tiresome task of physically transporting keys over vast distances. In due course they were joined by Ralph Merkle. Merkle was an intellectual refugee, having emigrated from another research group where the professor had no sympathy for the impossible dream of solving the key distribution problem. Says Hellman:

> Ralph, like us, was willing to be a fool. And the way to get to the top of the heap in terms of developing original research is to be a fool, because only fools keep trying. You have idea number 1, you get excited, and it flops. Then you have idea number 2, you get excited, and it flops. Then you have idea number 99, you get excited, and it flops. Only a fool would be excited by the 100th idea, but it might take 100 ideas before one really pays off. Unless you're foolish enough to be continually excited, you won't have the motivation, you won't have the energy to carry it through. God rewards fools.

The whole problem of key distribution is a classic catch-22 situation. If two people want to exchange a secret message over the phone, the sender must encrypt it. To encrypt the secret message the sender must use a key, which is itself a secret, so then there is the problem of transmitting the secret key to the receiver in order to transmit the secret message. In short, before two people can exchange a secret (an encrypted message) they must already share a secret (the key).

When thinking about the problem of key distribution, it is helpful to consider Alice, Bob and Eve, three fictional characters who have become the industry standard for discussions about cryptography. In a typical situation, Alice wants to send a message to Bob, or vice versa, and Eve is trying to eavesdrop. If Alice is sending private messages to Bob, she will encrypt each one before sending it, using a separate key each time. Alice is continually faced with the problem of key distribution because she has to convey the keys to Bob securely, otherwise he cannot decrypt the messages. One way to solve the problem is for Alice and Bob to meet up once

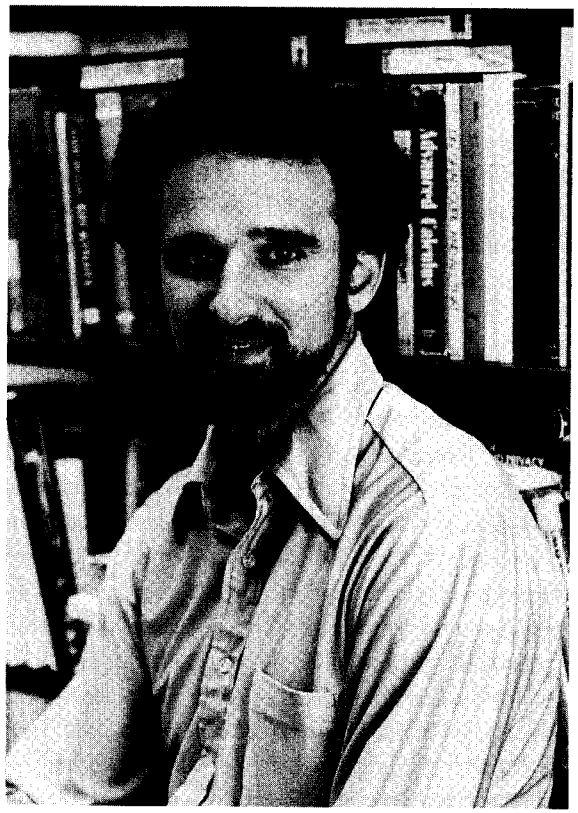

Figure 63 Martin Hellman.

a week and exchange enough keys to cover the messages that might be sent during the next seven days. Exchanging keys in person is certainly secure, but it is inconvenient and, if either Alice or Bob is taken ill, the system breaks down. Alternatively, Alice and Bob could hire couriers, which would be less secure and more expensive, but at least they have delegated some of the work. Either way, it seems that the distribution of keys is unavoidable. For two thousand years this was considered to be an axiom of cryptography–an indisputable truth. However, there is a thought experiment that seems to defy the axiom.

Imagine that Alice and Bob live in a country where the postal system is completely immoral, and postal employees will read any unprotected correspondence. One day, Alice wants to send an intensely personal message to Bob. She puts it inside an iron box, closes it and secures it with a padlock and key. She puts the padlocked box in the post and keeps the key. However, when the box reaches Bob, he is unable to open it because he does not have the key. Alice might consider putting the key inside another box, padlocking it and sending it to Bob, but without the key to the second padlock he is unable to open the second box, so he cannot obtain the key that opens the first box. The only way around the problem seems to be for Alice to make a copy of her key and give it to Bob in advance when they meet for coffee. So far, I have just restated the same old problem in a new scenario. Avoiding key distribution seems logically impossible–surely, if Alice wants to lock something in a box so that only Bob can open it, she must give him a copy of the key. Or, in terms of cryptography, if Alice wants to encipher a message so that only Bob can decipher it, she must give him a copy of the key. Key exchange is an inevitable part of encipherment–or is it?

Now picture the following scenario. As before, Alice wants to send an intensely personal message to Bob. Again, she puts her secret message in an iron box, padlocks it and sends it to Bob. When the box arrives, Bob adds his own padlock and sends the box back to Alice. When Alice receives the box, it is now secured by two padlocks. She removes her own padlock, leaving just Bob's padlock to secure the box. Finally she sends the box back to Bob. And here is the crucial difference: Bob can now open the box because it is secured only with his own padlock, to which he alone has the key.

The implications of this little story are enormous. It demonstrates that a secret message can be securely exchanged between two people without necessarily exchanging a key. For the first time we have a suggestion that key exchange might not be an inevitable part of cryptography. We can reinterpret the story in terms of encryption. Alice uses her own key to encrypt a message to Bob, who encrypts it again with his own key and returns it. When Alice receives the doubly encrypted message, she removes her own encryption and returns it to Bob, who can then remove his own encryption and read the message.

It seems that the problem of key distribution might have been solved, because the doubly encrypted scheme requires no exchange of keys. However, there is a fundamental obstacle to implementing a system in which Alice encrypts, Bob encrypts, Alice decrypts and Bob decrypts. The problem is the order in which the encryptions and decryptions are performed. In general, the order of encryption and decryption is crucial, and should obey the maxim "last on, first off." In other words, the last stage of encryption should be the first to be decrypted. In the above scenario, Bob performed the last stage of encryption, so this should have been the first to be decrypted, but it was Alice who removed her encryption first, before Bob removed his. The importance of order is most easily grasped by examining something we do every day. In the morning we put on our socks, and then we put on our shoes, and in the evening we remove our shoes before removing our socks–it is impossible to remove the socks before the shoes. We must obey the maxim "last on, first off."

Some very elementary ciphers, such as the Caesar cipher, are so simple that order does not matter. However, in the 1970s it seemed that any form of strong encryption must always obey the "last on, first off" rule. If a message is encrypted with Alice's key and then with Bob's key, then it must be decrypted with Bob's key before it can be decrypted with Alice's key. Order is crucial even with a monoalphabetic substitution cipher. Imagine that Alice and Bob have their own keys, as shown on the next page, and let us take a look at what happens when the order is incorrect. Alice uses her key to encrypt a message to Bob, then Bob reencrypts the result using his own key; Alice uses her key to perform a partial decryption, and finally Bob attempts to use his key to perform the full decryption.

Alice's key

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
H	F	S	U	G	T	A	K	V	D	E	O	Y	J	B	P	N	X	W	C	Q	R	I	M	Z	L

Bob's key

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
C	P	M	G	A	T	N	O	J	E	F	W	I	Q	B	U	R	Y	H	X	S	D	Z	K	L	V

Message	m	e	e	t	m	e	a	t	n	o	o	n
Encrypted with Alice's key	Y	G	G	C	Y	G	H	C	J	B	B	J
Encrypted with Bob's key	L	N	N	M	L	N	O	M	E	P	P	E
Decrypted with Alice's key	Z	Q	Q	X	Z	Q	L	X	K	P	P	K
Decrypted with Bob's key	w	n	n	t	w	n	y	t	x	b	b	x

The result is nonsense. However, you can check for yourself that if the decryption order were reversed, and Bob decrypted before Alice, thus obeying the "last on, first off" rule, then the result would have been the original message. But if order is so important, why did the padlock system seem to work in the anecdote about locked boxes? The answer is that order is not important for padlocks. I can apply twenty padlocks to a box and undo them in any order, and at the end the box will open. Unfortunately, encryption systems are far more sensitive than padlocks when it comes to order.

Although the doubly padlocked box approach would not work for real-world cryptography, it inspired Diffie and Hellman to search for a practical method of circumventing the key distribution problem. They spent month after month attempting to find a solution. Although every idea ended in failure, they behaved like perfect fools and persevered. Their research concentrated on the examination of various mathematical *functions*. A function is any mathematical operation that turns one number into another number. For example, "doubling" is a type of function, because it turns the number 3 into 6, or the number 9 into 18. Furthermore, we can think of all forms of computer encryption as functions because they turn one number (the plaintext) into another number (the ciphertext).

Most mathematical functions are classified as two-way functions because they are easy to do, and easy to undo. For example, "doubling"

is a two-way function because it is easy to double a number to generate a new number, and just as easy to undo the function and get from the doubled number back to the original number. For example, if we know that the result of doubling is 26, then it is trivial to reverse the function and deduce that the original number was 13. The easiest way to understand the concept of a two-way function is in terms of an everyday activity. The act of turning on a light switch is a function, because it turns an ordinary lightbulb into an illuminated lightbulb. This function is two-way because if a switch is turned on, it is easy enough to turn it off and return the lightbulb to its original state.

However, Diffie and Hellman were not interested in two-way functions. They focused their attention on one-way functions. As the name suggests, a one-way function is easy to do but very difficult to undo. In other words, two-way functions are reversible, but one-way functions are not reversible. Once again, the best way to illustrate a one-way function is in terms of an everyday activity. Mixing yellow and blue paint to make green paint is a one-way function because it is easy to mix the paint, but impossible to unmix it. Another one-way function is the cracking of an egg, because it is easy to crack an egg but impossible then to return the egg to its original condition. For this reason, one-way functions are sometimes called Humpty Dumpty functions.

Modular arithmetic, sometimes called *clock arithmetic* in schools, is an area of mathematics that is rich in one-way functions. In modular arithmetic, mathematicians consider a finite group of numbers arranged in a loop, rather like the numbers on a clock. For example, Figure 64 shows a clock for modular 7 (or mod 7), which has only the 7 numbers from 0 to 6. To work out 2 + 3, we start at 2 and move around 3 places to reach 5, which is the same answer as in normal arithmetic. To work out 2 + 6 we start at 2 and move around 6 places, but this time we go around the loop and arrive at 1, which is not the result we would get in normal arithmetic. These results can be expressed as:

$$2 + 3 = 5 \pmod 7 \quad \text{and} \quad 2 + 6 = 1 \pmod 7$$

Modular arithmetic is relatively simple, and in fact we do it every day when we talk about time. If it is 9 o'clock now, and we have a meeting 8 hours from now, we would say that the meeting is at 5 o'clock, not

17 o'clock. We have mentally calculated 9 + 8 in (mod 12). Imagine a clock face, look at 9, and then move around 8 spaces, and we end up at 5:

$9 + 8 = 5 \pmod{12}$

Rather than visualizing clocks, mathematicians often take the shortcut of performing modular calculations according to the following recipe. First, perform the calculation in normal arithmetic. Second, if we want to know the answer in (mod x), we divide the normal answer by x and note the remainder. This remainder is the answer in (mod x). To find the answer to $11 \times 9 \pmod{13}$, we do the following:

$11 \times 9 = 99$

$99 \div 13 = 7$, remainder 8

$11 \times 9 = 8 \pmod{13}$

Functions performed in the modular arithmetic environment tend to behave erratically, which in turn sometimes makes them one-way functions. This becomes evident when a simple function in normal arithmetic is compared with the same simple function in modular arithmetic. In the former environment the function will be two-way and easy to reverse; in the latter environment it will be one-way and hard to reverse. As an example, let us take the function 3^x. This means take a number x, then multiply 3 by itself x times in order to get the new number. For example, if $x = 2$, and we perform the function, then:

$3^x = 3^2 = 3 \times 3 = 9.$

In other words, the function turns 2 into 9. In normal arithmetic, as the value of x increases so does the result of the function. Hence, if we were given the result of the function it would be relatively easy to work backward and deduce the original number. For example, if the result is 81, we can deduce that x is 4, because $3^4 = 81$. If we made a mistake and guessed that x is 5, we could work out that $3^5 = 243$, which tells us that our choice of x is too big. We would then reduce our choice of x to 4, and we would have the right answer. In short, even when we guess wrongly we can home in on the correct value of x, and thereby reverse the function.

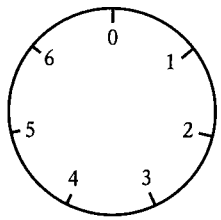

Figure 64 Modular arithmetic is performed on a finite set of numbers, which can be thought of as numbers on a clock face. In this case, we can work out 6 + 5 in modular 7 by starting at 6 and moving around five spaces, which brings us to 4.

However, in modular arithmetic this same function does not behave so sensibly. Imagine that we are told that 3^x in (mod 7) is 1, and we are asked to find the value of x. No value springs to mind, because we are generally unfamiliar with modular arithmetic. We could take a guess that $x = 5$, and we could work out the result of $3^5 \pmod{7}$. The answer turns out to be 5, which is too big, because we are looking for an answer of just 1. We might be tempted to reduce the value of x and try again. But we would be heading in the wrong direction, because the actual answer is $x = 6$.

In normal arithmetic we can test numbers and can sense whether we are getting warmer or colder. The environment of modular arithmetic gives no helpful clues, and reversing functions is much harder. Often, the only way to reverse a function in modular arithmetic is to compile a table by calculating the function for many values of x until the right answer is found. Table 25 shows the result of calculating several values of the function in both normal arithmetic and modular arithmetic. It clearly demonstrates the erratic behavior of the function when calculated in modular arithmetic. Although drawing up such a table is only a little tedious when we are dealing with relatively small numbers, it would be excruciatingly painful to build a table to deal with a function such as $453^x \pmod{21{,}997}$. This is a classic example of a one-way function, because I could pick a value for x and calculate the result of the function, but if I gave you a

Table 25 Values of the function 3^x calculated in normal arithmetic (row 2) and modular arithmetic (row 3). The function increases continuously in normal arithmetic, but is highly erratic in modular arithmetic.

x	1	2	3	4	5	6
3^x	3	9	27	81	243	729
$3^x \pmod 7$	3	2	6	4	5	1

result, say 5,787, you would have enormous difficulty in reversing the function and deducing my choice of x. It took me just seconds to do my calculation and generate 5,787, but it would take you hours to draw up the table and work out my choice of x.

After two years of focusing on modular arithmetic and one-way functions, Hellman's foolishness began to pay off. In the spring of 1976 he hit upon a strategy for solving the key exchange problem. In half an hour of frantic scribbling, he proved that Alice and Bob could agree on a key without meeting, thereby disposing of an axiom that had lasted for centuries. Hellman's idea relied on a one-way function of the form $Y^x \pmod P$. Initially, Alice and Bob agree on values for Y and P. Almost any values are fine, but there are some restrictions, such as Y being smaller than P. These values are not secret, so Alice can telephone Bob and suggest that, say, $Y = 7$ and $P = 11$. Even if the telephone line is insecure and nefarious Eve hears this conversation, it does not matter, as we shall see later. Alice and Bob have now agreed on the one-way function $7^x \pmod{11}$. At this point they can begin the process of trying to establish a secret key without meeting. Because they work in parallel, I explain their actions in the two columns of Table 26.

Having followed the stages in Table 26, you will see that, without meeting, Alice and Bob have agreed on the same key, which they can use to encipher a message. For example, they could use their number, 9, as the key for a DES encryption. (DES actually uses much larger numbers as the key, and the exchange process described in Table 26 would be performed with much larger numbers, resulting in a suitably large DES key.) By using Hellman's scheme, Alice and Bob have been able to agree on a key, yet they did not have to meet up and whisper the key to each other. The extraordinary achievement is that the secret key was agreed via an exchange of information on a normal telephone line. But if Eve tapped this line, then surely she also knows the key?

Let us examine Hellman's scheme from Eve's point of view. If she is tapping the line, she knows only the following facts: that the function is $7^x \pmod{11}$, that Alice sends $\alpha = 2$ and that Bob sends $\beta = 4$. In order to find the key, she must either do what Bob does, which is turn α into the key by knowing B, or do what Alice does, which is turn β into the key by knowing A. However, Eve does not know the value of A or B because

Table 26 The general one-way function is $Y^x \pmod P$. Alice and Bob have chosen values for Y and P, and hence have agreed on the one-way function $7^x \pmod{11}$.

	Alice	**Bob**
Stage 1	Alice chooses a number, say 3, and keeps it secret. We label her number A.	Bob chooses a number, say 6, and keeps it secret. We will label his number B.
Stage 2	Alice puts 3 into the one-way function and works out the result of $7^A \pmod{11}$: $7^3 \pmod{11} = 343 \pmod{11} = 2$	Bob puts 6 into the one-way function and works out the result of $7^B \pmod{11}$: $7^6 \pmod{11} = 117{,}649 \pmod{11} = 4$
Stage 3	Alice calls the result of this calculation α, and she sends her result, 2, to Bob.	Bob calls the result of this calculation β, and he sends his result, 4, to Alice.
The swap	Ordinarily this would be a crucial moment, because Alice and Bob are exchanging information, and therefore this is an opportunity for Eve to eavesdrop and find out the details of the information. However, it turns out that Eve can listen in without it affecting the ultimate security of the system. Alice and Bob could use the same telephone line that they used to agree the values for Y and P, and Eve could intercept the two numbers that are being exchanged, 2 and 4. However, these numbers are not the key, which is why it does not matter if Eve knows them.	
Stage 4	Alice takes Bob's result, and works out the result of $\beta^A \pmod{11}$: $4^3 \pmod{11} = 64 \pmod{11} = 9$	Bob takes Alice's result, and works out the result of $\alpha^B \pmod{11}$: $2^6 \pmod{11} = 64 \pmod{11} = 9$
The key	Miraculously, Alice and Bob have ended up with the same number, 9. This is the key!	

Alice and Bob have not exchanged these numbers, and have kept them secret. Eve is stymied. She has only one hope: in theory, she could work out A from α, because α was a consequence of putting A into a function, and Eve knows the function. Or she could work out B from β, because β was a consequence of putting B into a function, and once again Eve knows the function. Unfortunately for Eve, the function is one-way, so whereas it was easy for Alice to turn A into α and for Bob to turn B into β, it is very difficult for Eve to reverse the process, especially if the numbers are very large.

Bob and Alice exchanged just enough information to allow them to establish a key, but this information was insufficient for Eve to work out the key. As an analogy for Hellman's scheme, imagine a cipher that somehow uses color as the key. First, let us assume that everybody, including Alice, Bob and Eve, has a three-liter pot containing one liter of yellow paint. If Alice and Bob want to agree on a secret key, each of them adds one liter of their own secret color to their own pot. Alice might add a peculiar shade of purple, while Bob might add crimson. Each sends their own mixed pot to the other. Finally, Alice takes Bob's mixture and adds one liter of her own secret color, and Bob takes Alice's mixture and adds one liter of his own secret color. Both pots should now be the same color, because they both contain one liter of yellow, one liter of purple and one liter of crimson. It is the exact color of the doubly contaminated pots that is used as the key. Alice has no idea what color was added by Bob, and Bob has no idea what color was added by Alice, but they have both achieved the same end. Meanwhile, Eve is furious. Even if she intercepts the intermediate pots she cannot work out the color of the final pots, which is the agreed key. She might see the color of the mixed pot containing yellow and Alice's secret color on its way to Bob, and she might see the color of the mixed pot containing yellow and Bob's secret color on its way to Alice, but in order to work out the key she really needs to know Alice and Bob's original secret colors. However, Eve cannot work out Alice and Bob's secret colors by looking at the mixed pots. Even if she takes a sample from one of the mixed paints, she cannot unmix the paint to find out the secret color, because mixing paint is a one-way function.

Hellman's breakthrough came while he was working at home late one night, so by the time he had finished his calculations it was too late to call

Diffie and Merkle. He had to wait until the following morning to reveal his discovery to the only two other people in the world who had believed that a solution to the key distribution problem was even possible. "The muse whispered to me," says Hellman, "but we all laid the foundations together." Diffie immediately recognized the power of Hellman's breakthrough: "Marty explained his system of key exchange in all its unnerving simplicity. Listening to him, I realized that the notion had been at the edge of my mind for some time, but had never really broken through."

The Diffie–Hellman–Merkle key exchange scheme, as it is known, enables Alice and Bob to establish a secret via public discussion. It is one of the most counterintuitive discoveries in the history of science, and it forced the cryptographic establishment to rewrite the rules of encryption. Diffie, Hellman and Merkle publicly demonstrated their discovery at the National Computer Conference in June 1976, and astonished the audience of cryptoexperts. The following year they filed for a patent. Henceforth, Alice and Bob no longer had to meet in order to exchange a key. Instead, Alice could just call Bob on the phone, exchange a couple of numbers with him, mutually establish a secret key and then proceed to encrypt.

Although Diffie–Hellman–Merkle key exchange was a gigantic leap forward, the system was not perfect because it was inherently inconvenient. Imagine that Alice lives in Hawaii, and that she wants to send an e-mail to Bob in Istanbul. Bob is probably asleep, but the joy of e-mail is that Alice can send a message at any time, and it will be waiting on Bob's computer when he wakes up. However, if Alice wants to encrypt her message, then she needs to agree a key with Bob, and in order to perform the key exchange it is preferable for Alice and Bob to be on-line at the same time–establishing a key requires a mutual exchange of information. In effect, Alice has to wait until Bob wakes up. Alternatively, Alice could transmit her part of the key exchange, and wait 12 hours for Bob's reply, at which point the key is established and Alice can, if she is not asleep herself, encrypt and transmit the message. Either way, Hellman's key exchange system hinders the spontaneity of e-mail.

Hellman had shattered one of the tenets of cryptography and proved that Bob and Alice did not have to meet to agree a secret key. Next, somebody merely had to come up with a more efficient scheme for overcoming the problem of key distribution.

The Birth of Public Key Cryptography

Mary Fisher has never forgotten the first time that Whitfield Diffie asked her out on a date: "He knew I was a space buff, so he suggested we go and see a launch. Whit explained that he was leaving that evening to see Skylab take off, and so we drove all night, and we got there at about 3 A.M. The bird was on the path, as they used to say in those days. Whit had press credentials, but I didn't. So when they asked for my identification and asked who I was, Whit said 'My wife.' That was 16 November 1973." They did eventually marry, and during the early years Mary supported her husband during his cryptographic meditations. Diffie was still being employed as a graduate student, which meant that he received only a meager salary. Mary, an archaeologist by training, took a job with British Petroleum in order to make ends meet.

While Martin Hellman had been developing his method of key exchange, Whitfield Diffie had been working on a completely different approach to solving the problem of key distribution. He often went through long periods of barren contemplation, and on one occasion in 1975 he became so frustrated that he told Mary that he was just a failed scientist who would never amount to anything. He even told her that she ought to find someone else. Mary told him that she had absolute faith in him, and just two weeks later Diffie came up with his truly brilliant idea.

He can still recall how the idea flashed into his mind, and then almost vanished: "I walked downstairs to get a Coke, and almost forgot about the idea. I remembered that I'd been thinking about something interesting, but couldn't quite recall what it was. Then it came back in a real adrenaline rush of excitement. I was actually aware for the first time in my work on cryptography of having discovered something really valuable. Everything that I had discovered in the subject up to this point seemed to me to be mere technicalities." It was midafternoon, and he had to wait a couple of hours before Mary returned. "Whit was waiting at the door," she recalls. "He said he had something to tell me and he had a funny look on his face. I walked in and he said, 'Sit down, please, I want to talk to you. I believe that I have made a great discovery–I know I am the first person to have done this.' The world stood still for me at that moment. I felt like I was living in a Hollywood film."

Diffie had concocted a new type of cipher, one that incorporated a so-called *asymmetric key*. So far, all the encryption techniques described in this book have been *symmetric*, which means that the unscrambling process is simply the opposite of scrambling. For example, the Enigma machine uses a certain key setting to encipher a message, and the receiver uses an identical machine in the same key setting to decipher it. Similarly, DES encipherment uses a key to perform 16 rounds of scrambling, and then DES decipherment uses the same key to perform the 16 rounds in reverse. Both sender and receiver effectively have equivalent knowledge, and they both use the same key to encrypt and decrypt–their relationship is symmetric. On the other hand, in an asymmetric key system, as the name suggests, the encryption key and the decryption key are not identical. In an asymmetric cipher, if Alice knows the encryption key she can encrypt a message, but she cannot decrypt a message. In order to decrypt, Alice must have access to the decryption key. This distinction between the encryption and decryption keys is what makes an asymmetric cipher special.

At this point it is worth stressing that although Diffie had conceived of the general concept of an asymmetric cipher, he did not actually have a specific example of one. However, the mere concept of an asymmetric cipher was revolutionary. If cryptographers could find a genuine working asymmetric cipher, a system that fulfilled Diffie's requirements, then the implications for Alice and Bob would be enormous. Alice could create her own pair of keys: an encryption key and a decryption key. If we assume that the asymmetric cipher is a form of computer encryption, then Alice's encryption key is a number, and her decryption key is a different number. Alice keeps the decryption key secret, so it is commonly referred to as Alice's *private key*. However, she publishes the encryption key so that everybody has access to it, which is why it is commonly referred to as Alice's *public key*. If Bob wants to send Alice a message, he simply looks up her public key, which would be listed in something akin to a telephone directory. Bob then uses Alice's public key to encrypt the message. He sends the encrypted message to Alice, and when it arrives Alice can decrypt it using her private decryption key. Similarly, if Charlie, Dawn or Edward want to send Alice an encrypted message, they too can look up Alice's public encryption key, and in each case only Alice has access to the private decryption key required to decrypt the messages.

The great advantage of this system is that there is no toing and froing, as there is with Diffie–Hellman–Merkle key exchange. Bob does not have to wait to get information from Alice before he can encrypt and send a message to her, he merely has to look up her public encryption key. Furthermore, the asymmetric cipher still overcomes the problem of key distribution. Alice does not have to transport the public encryption key securely to Bob: in complete contrast, she can now publicize her public encryption key as widely as possible. She wants the whole world to know her public encryption key so that anybody can use it to send her encrypted messages. At the same time, even if the whole world knows Alice's public key, none of them, including Eve, can decrypt any messages encrypted with it, because knowledge of the public key will not help in decryption. In fact, once Bob has encrypted a message using Alice's public key, even he cannot decrypt it. Only Alice, who possesses the private key, can decrypt the message.

This is the exact opposite of a traditional symmetric cipher, in which Alice has to go to great lengths to transport the encryption key securely to Bob. In a symmetric cipher the encryption key is the same as the decryption key, so Alice and Bob must take enormous precautions to ensure that the key does not fall into Eve's hands. This is the root of the key distribution problem.

Returning to padlock analogies, asymmetric cryptography can be thought of in the following way. Anybody can close a padlock simply by clicking it shut, but only the person who has the key can open it. Locking (encryption) is easy, something everybody can do, but unlocking (decryption) can be done only by the owner of the key. The trivial knowledge of knowing how to click the padlock shut does not tell you how to unlock it. Taking the analogy further, imagine that Alice designs a padlock and key. She guards the key, but she manufactures thousands of replica padlocks and distributes them to post offices all over the world. If Bob wants to send a message, he puts it in a box, goes to the local post office, asks for an "Alice padlock" and padlocks the box. Now he is unable to unlock the box, but when Alice receives it she can open it with her unique key. The padlock and the process of clicking it shut is equivalent to the public encryption key, because everyone has access to the padlocks, and everyone can use a padlock to seal a message in a box. The padlock's key is equivalent to the private decryption key, because only Alice has it, only she can open the padlock, and only she can gain access to the message in the box.

The system seems simple when it is explained in terms of padlocks, but it is far from trivial to find a mathematical function that does the same job, something that can be incorporated into a workable cryptographic system. To turn asymmetric ciphers from a great idea into a practical invention, somebody had to discover an appropriate mathematical function. Diffie envisaged a special type of one-way function, one that could be reversed under exceptional circumstances. In Diffie's asymmetric system, Bob encrypts the message using the public key, but he is unable to decrypt it–this is essentially a one-way function. However, Alice is able to decrypt the message because she has the private key, a special piece of information that allows her to reverse the function. Once again, padlocks are a good analogy–shutting the padlock is a one-way function, because in general it is hard to open the padlock unless you have something special (the key), in which case the function is easily reversed.

Diffie published an outline of his idea in the summer of 1975, whereupon other scientists joined the search for an appropriate one-way function, one that fulfilled the criteria required for an asymmetric cipher. Initially there was great optimism, but by the end of the year nobody had been able to find a suitable candidate. As the months passed, it seemed increasingly likely that special one-way functions did not exist. It seemed that Diffie's idea worked in theory but not in practice. Nevertheless, by the end of 1976 the team of Diffie, Hellman and Merkle had revolutionized the world of cryptography. They had persuaded the rest of the world that there was a solution to the key distribution problem, and had created Diffie–Hellman–Merkle key exchange–a workable but imperfect system. They had also proposed the concept of an asymmetric cipher–a perfect but as yet unworkable system. They continued their research at Stanford University, attempting to find a special one-way function that would make asymmetric ciphers a reality. However, they failed to make the discovery. The race to find an asymmetric cipher was won by another trio of researchers, based 5,000 km away on the East Coast of America.

Prime Suspects

"I walked into Ron Rivest's office," recalls Leonard Adleman, "and Ron had this paper in his hands. He started saying, 'These Stanford guys have this really blah, blah, blah.' And I remember thinking, 'That's nice, Ron, but I have something else I want to talk about.' I was entirely unaware of the history of cryptography and I was distinctly uninterested in what he was saying." The paper that had made Ron Rivest so excited was by Diffie and Hellman, and it described the concept of asymmetric ciphers. Eventually Rivest persuaded Adleman that there might be some interesting mathematics in the problem, and together they resolved to try to find a one-way function that fitted the requirements of an asymmetric cipher. They were joined in the hunt by Adi Shamir. All three men were researchers on the eighth floor of the MIT Laboratory for Computer Science.

Rivest, Shamir and Adleman formed a perfect team. Rivest is a computer scientist with a tremendous ability to absorb new ideas and apply them in unlikely places. He always kept up with the latest scientific papers, which inspired him to come up with a whole series of weird and wonderful candidates for the one-way function at the heart of an asymmetric cipher. However, each candidate was flawed in some way. Shamir, another computer scientist, has a lightning intellect and an ability to see through the debris and focus on the core of a problem. He too regularly generated ideas for formulating an asymmetric cipher, but his ideas were also inevitably flawed. Adleman, a mathematician with enormous stamina, rigor and patience, was largely responsible for spotting the flaws in the ideas of Rivest and Shamir, ensuring that they did not waste time following false leads. Rivest and Shamir spent a year coming up with new ideas, and Adleman spent a year shooting them down. The threesome began to lose hope, but they were unaware that this process of continual failure was a necessary part of their research, gently steering them away from sterile mathematical territory and toward more fertile ground. In due course, their efforts were rewarded.

In April 1977, Rivest, Shamir and Adleman spent Passover at the house of a student, and had consumed significant amounts of Manischewitz wine before returning to their respective homes some time around midnight. Rivest, unable to sleep, lay on his couch reading a mathematics

textbook. He began mulling over the question that had been puzzling him for weeks–is it possible to build an asymmetric cipher? Is it possible to find a one-way function that can be reversed only if the receiver has some special information? Suddenly, the mists began to clear and he had a revelation. He spent the rest of that night formalizing his idea, effectively writing a complete scientific paper before daybreak. Rivest had made a breakthrough, but it had grown out of a yearlong collaboration with Shamir and Adleman, and it would not have been possible without them. Rivest finished off the paper by listing the authors alphabetically; Adleman, Rivest, Shamir.

The next morning, Rivest handed the paper to Adleman, who went through his usual process of trying to tear it apart, but this time he could find no faults. His only criticism was with the list of authors. "I told Ron to take my name off the paper," recalls Adleman. "I told him that it was his invention, not mine. But Ron refused and we got into a discussion about it. We agreed that I would go home and contemplate it for one night, and consider what I wanted to do. I went back the next day and suggested to Ron that I be the third author. I recall thinking that this paper would be the least interesting paper that I will ever be on." Adleman could not have been more wrong. The system, dubbed RSA

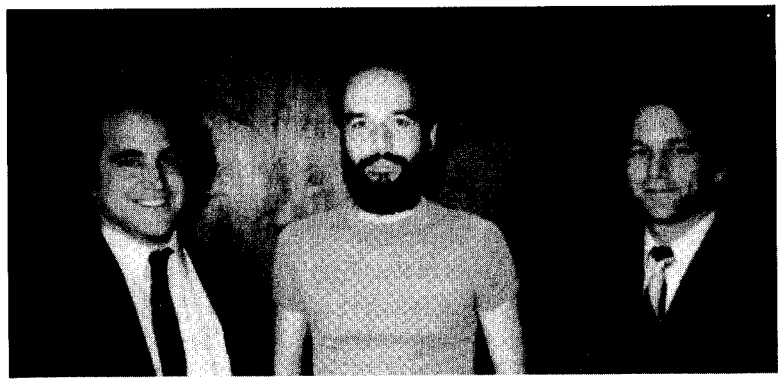

Figure 65 Ronald Rivest, Adi Shamir and Leonard Adleman.

(Rivest, Shamir, Adleman) as opposed to ARS, went on to become the most influential cipher in modern cryptography.

Before exploring Rivest's idea, here is a quick reminder of what scientists were looking for in order to build an asymmetric cipher:

(1) Alice must create a public key, which she would then publish so that Bob (and everybody else) can use it to encrypt messages to her. Because the public key is a one-way function, it must be virtually impossible for anybody to reverse it and decrypt Alice's messages.

(2) However, Alice needs to decrypt the messages being sent to her. She must therefore have a private key, some special piece of information, which allows her to reverse the effect of the public key. Therefore, Alice (and Alice alone) has the power to decrypt any messages sent to her.

At the heart of Rivest's asymmetric cipher is a one-way function based on the sort of modular functions described earlier in the chapter. Rivest's one-way function can be used to encrypt a message–the message, which is effectively a number, is put into the function, and the result is the ciphertext, another number. I shall not describe Rivest's one-way function in detail (for which see Appendix J), but I shall explain one particular aspect of it, known simply as N, because it is N that makes this one-way function reversible under certain circumstances, and therefore ideal for use as an asymmetric cipher.

N is important because it is a flexible component of the one-way function, which means that each person can choose a different value of N, and personalize the one-way function. In order to choose her personal value of N, Alice picks two prime numbers, p and q, and multiplies them together. A prime number is one that has no divisors except itself and 1. For example, 7 is a prime number because no numbers except 1 and 7 will divide into it without leaving a remainder. Likewise, 13 is a prime number because no numbers except 1 and 13 will divide into it without leaving a remainder. However, 8 is not a prime number, because it can be divided by 2 and 4.

So, Alice could choose her prime numbers to be $p = 17{,}159$ and $q = 10{,}247$. Multiplying these two numbers together gives $N = 17{,}159 \times 10{,}247 = 175{,}828{,}273$. Alice's choice of N effectively becomes her public encryption key, and she could print it on her business card, post it on the Internet, or publish it in a public key directory along with everybody else's value of N. If Bob wants to encrypt a message to Alice, he looks up Alice's value of N (175,828,273) and then inserts it into the general form of the one-way function, which would also be public knowledge. Bob now has a one-way function tailored with Alice's public key, so it could be called Alice's one-way function. To encrypt a message to Alice, he takes Alice's one-way function, inserts the message, notes down the result and sends it to Alice.

At this point the encrypted message is secure because nobody can decipher it. The message has been encrypted with a one-way function, so reversing the one-way function and decrypting the message is, by definition, very difficult. However, the question remains–how can Alice decrypt the message? In order to read messages sent to her, Alice must have a way of reversing the one-way function. She needs to have access to some special piece of information that allows her to decrypt the message. Fortunately for Alice, Rivest designed the one-way function so that it is reversible to someone who knows the values of p and q, the two prime numbers that are multiplied together to give N. Although Alice has told the world that her value for N is 175,828,273, she has not revealed her values for p and q, so only she has the special information required to decrypt her own messages.

We can think of N as the public key, the information that is available to everybody, the information required to encrypt messages to Alice. Whereas, p and q are the private key, available only to Alice, the information required to decrypt these messages.

The exact details of how p and q can be used to reverse the one-way function are outlined in Appendix J. However, there is one question that must be addressed immediately. If everybody knows N, the public key, then surely people can deduce p and q, the private key, and read Alice's messages? After all, N was created from p and q. In fact, it turns out that if N is large enough, it is virtually impossible to deduce p and q from N, and this is perhaps the most beautiful and elegant aspect of the RSA asymmetric cipher.

Alice created N by choosing p and q, and then multiplying them together. The fundamental point is that this is in itself a one-way function.

To demonstrate the one-way nature of multiplying primes, we can take two prime numbers, such as 9,419 and 1,933, and multiply them together. With a calculator it takes just a few seconds to get the answer, 18,206,927. However, if instead we were given 18,206,927 and asked to find the prime factors (the two numbers that were multiplied to give 18,206,927) it would take us much longer. If you doubt the difficulty of finding prime factors, then consider the following. It took me just ten seconds to generate the number 1,709,023, but it will take you and a calculator the best part of an afternoon to work out the prime factors.

This system of asymmetric cryptography, known as RSA, is said to be a form of *public key cryptography*. To find out how secure RSA is, we can examine it from Eve's point of view, and try to break a message from Alice to Bob. To encrypt a message to Bob, Alice must look up Bob's public key. To create his public key, Bob picked his own prime numbers, p_B and q_B, and multiplied them together to get N_B. He has kept p_B and q_B secret, because these make up his private decryption key, but he has published N_B, which is equal to 408,508,091. So Alice inserts Bob's public key N_B into the general one-way encryption function, and then encrypts her message to him. When the encrypted message arrives, Bob can reverse the function and decrypt it using his values for p_B and q_B, which make up his private key. Meanwhile, Eve has intercepted the message en route. Her only hope of decrypting the message is to reverse the one-way function, and this is possible only if she knows p_B and q_B. Bob has kept p_B and q_B secret, but Eve, like everybody else, knows N_B is 408,508,091. Eve then attempts to deduce the values for p_B and q_B by working out which numbers would need to be multiplied together to get 408,508,091, a process known as *factoring*.

Factoring is very time-consuming, but exactly how long would it take Eve to find the factors of 408,508,091? There are various recipes for trying to factor N_B. Although some recipes are faster than others, they all essentially involve checking each prime number to see if it divides into N_B without a remainder. For example, 3 is a prime number, but it is not a factor of 408,508,091 because 3 will not perfectly divide into 408,508,091. So Eve moves on to the next prime number, 5. Similarly, 5 is not a factor, so Eve moves on to the next prime number, and so on. Eventually, Eve arrives at 18,313, the 2,000th prime number, which is indeed a factor of 408,508,091. Having found one factor, it is easy to find the other one,

which turns out to be 22,307. If Eve had a calculator and was able to check four primes a minute, then it would have taken her 500 minutes, or more than 8 hours, to find p_B and q_B. In other words, Eve would be able to work out Bob's private key in less than a day, and could therefore decipher the intercepted message in less than a day.

This is not a very high level of security, but Bob could have chosen much larger prime numbers and increased the security of his private key. For example, he could have chosen primes that are as big as 10^{65} (this means 1 followed by 65 zeros, or one hundred thousand, million, million, million, million, million, million, million, million, million, million). This would have resulted in a value for N that would have been roughly $10^{65} \times 10^{65}$, which is 10^{130}. A computer could multiply the two primes and generate N in just a second, but if Eve wanted to reverse the process and work out p and q, it would take inordinately longer. Exactly how long depends on the speed of Eve's computer. Security expert Simson Garfinkel estimated that a 100 MHz Intel Pentium computer with 8 MB of RAM would take roughly 50 years to factor a number as big as 10^{130}. Cryptographers tend to have a paranoid streak and consider worst-case scenarios, such as a worldwide conspiracy to crack their ciphers. So, Garfinkel considered what would happen if a hundred million personal computers (the number sold in 1995) ganged up together. The result is that a number as big as 10^{130} could be factored in about 15 seconds. Consequently, it is now generally accepted that for genuine security it is necessary to use even larger primes. For important banking transactions, N tends to be at least 10^{308}, which is ten million billion billion billion billion billion billion billion billion billion billion billion billion billion billion billion billion billion billion billion times bigger than 10^{130}. The combined efforts of a hundred million personal computers would take more than one thousand years to crack such a cipher. With sufficiently large values of p and q, RSA is impregnable.

The only caveat for the security of RSA public key cryptography is that at some time in the future somebody might find a quick way to factor N. It is conceivable that a decade from now, or even tomorrow, somebody will discover a method for rapid factoring, and thereafter RSA will become useless. However, for over two thousand years mathematicians have tried and failed to find a shortcut, and at the moment factoring

remains an enormously time-consuming calculation. Most mathematicians believe that factoring is an inherently difficult task, and that there is some mathematical law that forbids any shortcut. If we assume they are right, then RSA seems secure for the foreseeable future.

The great advantage of RSA public key cryptography is that it does away with all the problems associated with traditional ciphers and key exchange. Alice no longer has to worry about securely transporting the key to Bob, or that Eve might intercept the key. In fact, Alice does not care who sees the public key–the more the merrier, because the public key helps only with encryption, not decryption. The only thing that needs to remain secret is the private key used for decryption, and Alice can keep this with her at all times.

RSA was first announced in August 1977, when Martin Gardner wrote an article entitled "A New Kind of Cipher that Would Take Millions of Years to Break" for his "Mathematical Games" column in *Scientific American*. After explaining how public key cryptography works, Gardner issued a challenge to his readers. He printed a ciphertext and also provided the public key that had been used to encrypt it:

N = 114,381,625,757,888,867,669,235,779,976,146,612,010,218,296,
721,242,362,562,561,842,935,706,935,245,733,897,830,597,123,563,
958,705,058,989,075,147,599,290,026,879,543,541.

The challenge was to factor N into p and q, and then use these numbers to decrypt the message. The prize was $100. Gardner did not have space to explain the nitty-gritty of RSA, and instead he asked readers to write to MIT's Laboratory for Computer Science, who in turn would send back a technical memorandum that had just been prepared. Rivest, Shamir and Adleman were astonished by the three thousand requests they received. However, they did not respond immediately, because they were concerned that public distribution of their idea might jeopardize their chances of getting a patent. When the patent issues were eventually resolved, the trio held a celebratory party at which professors and students consumed pizzas and beer while stuffing envelopes with technical memoranda for the readers of *Scientific American*.

As for Gardner's challenge, it would take 17 years before the cipher would be broken. On April 26, 1994, a team of six hundred volunteers announced the factors of N:

q = 3,490,529,510,847,650,949,147,849,619,903,898,133,417,764,
638,493,387,843,990,820,577

p = 32,769,132,993,266,709,549,961,988,190,834,461,413,177,
642,967,992,942,539,798,288,533.

Using these values as the private key, they were able to decipher the message. The message was a series of numbers, but when converted into letters it read "the magic words are squeamish ossifrage." The factoring problem had been split among the volunteers, who came from countries as far apart as Australia, Britain, America and Venezuela. The volunteers used spare time on their workstations, mainframes and supercomputers, each of them tackling a fraction of the problem. In effect, a network of computers around the world were uniting and working simultaneously in order to meet Gardner's challenge. Even bearing in mind the mammoth parallel effort, some readers may still be surprised that RSA was broken in such a short time, but it should be noted that Gardner's challenge used a relatively small value of N–it was only of the order of 10^{129}. Today, users of RSA would pick a much larger value to secure important information. It is now routine to encrypt a message with a sufficiently large value of N so that all the computers on the planet would need longer than the age of the universe to break the cipher.

The Alternative History of Public Key Cryptography

Over the past twenty years, Diffie, Hellman and Merkle have become world-famous as the cryptographers who invented the concept of public key cryptography, while Rivest, Shamir and Adleman have been credited with developing RSA, the most beautiful implementation of public key cryptography. However, a recent announcement means that the history books are having to be rewritten. According to the British Government, public key cryptography was originally invented at the Government

Communications Headquarters (GCHQ) in Cheltenham, the top-secret establishment that was formed from the remnants of Bletchley Park after the Second World War. This is a story of remarkable ingenuity, anonymous heroes and a government cover-up that endured for decades.

The story starts in the late 1960s, when the British military began to worry about the problem of key distribution. Looking ahead to the 1970s, senior military officials imagined a scenario in which miniaturization of radios and a reduction in cost meant that every soldier could be in continual radio contact with his officer. The advantages of widespread communication would be enormous, but communications would have to be encrypted, and the problem of distributing keys would be insurmountable. This was an era when the only form of cryptography was symmetric, so an individual key would have to be securely transported to every member of the communications network. Any expansion in communications would eventually be choked by the burden of key distribution. At the beginning of 1969, the military asked James Ellis, one of Britain's foremost government cryptographers, to look into ways of coping with the key distribution problem.

Ellis was a curious and slightly eccentric character. He proudly boasted of traveling halfway around the world before he was even born–he was conceived in Britain, but was born in Australia. Then, while still a baby, he returned to London and grew up in the East End of the 1920s. At school his primary interest was science, and he went on to study physics at Imperial College before joining the Post Office Research Station at Dollis Hill, where Tommy Flowers had built Colossus, the first codebreaking computer. The cryptographic division at Dollis Hill was eventually absorbed into GCHQ, and so on April 1, 1965, Ellis moved to Cheltenham to join the newly formed Communications–Electronics Security Group (CESG), a special section of GCHQ devoted to ensuring the security of British communications. Because he was involved in issues of national security, Ellis was sworn to secrecy throughout his career. Although his wife and family knew that he worked at GCHQ, they were unaware of his discoveries and had no idea that he was one of the nation's most distinguished codemakers.

Despite his skills as a codemaker, Ellis was never put in charge of any of

the important GCHQ research groups. He was brilliant, but he was also unpredictable, introverted and not a natural team worker. His colleague Richard Walton recalled:

> He was a rather quirky worker, and he didn't really fit into the day-to-day business of GCHQ. But in terms of coming up with new ideas he was quite exceptional. You had to sort through some rubbish sometimes, but he was very innovative and always willing to challenge the orthodoxy. We would be in real trouble if everybody in GCHQ was like him, but we can tolerate a higher proportion of such people than most organizations. We put up with a number of people like him.

Figure 66 James Ellis.

One of Ellis's greatest qualities was his breadth of knowledge. He read any scientific journal he could get his hands on, and never threw anything away. For security reasons, GCHQ employees must clear their desks each evening and place everything in locked cabinets, which meant that Ellis's cabinets were stuffed full with the most obscure publications imaginable. He gained a reputation as a cryptoguru, and if other researchers found themselves with impossible problems, they would knock on his door in the hope that his vast knowledge and originality would provide a solution. It was probably because of this reputation that he was asked to examine the key distribution problem.

The cost of key distribution was already enormous, and would become the limiting factor to any expansion in encryption. Even a reduction of 10 per cent in the cost of key distribution would significantly cut the military's security budget. However, instead of merely nibbling away at the problem, Ellis immediately looked for a radical and complete solution. "He would always approach a problem by asking, 'Is this really what we want to do?'" says Walton. "James being James, one of the first things he did was to challenge the requirement that it was necessary to share secret data, by which I mean the key. There was no theorem that said you had to have a shared secret. This was something that was challengeable."

Ellis began his attack on the problem by searching through his treasure trove of scientific papers. Many years later, he recorded the moment when he discovered that key distribution was not an inevitable part of cryptography:

> The event which changed this view was the discovery of a wartime Bell Telephone report by an unknown author describing an ingenious idea for secure telephone speech. It proposed that the recipient should mask the sender's speech by adding noise to the line. He could subtract the noise afterward since he had added it and therefore knew what it was. The obvious practical disadvantages of this system prevented it being actually used, but it has some interesting characteristics. The difference between this and conventional encryption is that in this case the recipient takes part in the encryption process . . . So the idea was born.

Noise is the technical term for any signal that impinges on a communication. Normally it is generated by natural phenomena, and its most irritating feature is that it is entirely random, which means that removing

noise from a message is very difficult. If a radio system is well designed, then the level of noise is low and the message is clearly audible, but if the noise level is high and it swamps the message, there is no way to recover the message. Ellis was suggesting that the receiver, Alice, deliberately create noise, which she could measure before adding it to the communication channel that connects her with Bob. Bob could then send a message to Alice, and if Eve tapped the communications channel she would be unable to read the message because it would be swamped in noise. Eve would be unable to disentangle the noise from the message. The only person who can remove the noise and read the message is Alice, because she is in the unique position of knowing the exact nature of the noise, having put it there in the first place. Ellis realized that security had been achieved without exchanging any key. The key was the noise, and only Alice needed to know the details of the noise.

In a memorandum, Ellis detailed his thought processes: "The next question was the obvious one. Can this be done with ordinary encipherment? Can we produce a secure encrypted message, readable by the authorized recipient without any prior secret exchange of the key? This question actually occurred to me in bed one night, and the proof of the theoretical possibility took only a few minutes. We had an existence theorem. The unthinkable was actually possible." (An existence theorem shows that a particular concept is possible, but is not concerned with the details of the concept.) In other words, until this moment, searching for a solution to the key distribution problem was like looking for a needle in a haystack, with the possibility that the needle might not even be there. However, thanks to the existence theorem, Ellis now knew that the needle was in there somewhere.

Ellis's ideas were very similar to those of Diffie, Hellman and Merkle, except that he was several years ahead of them. However, nobody knew of Ellis's work because he was an employee of the British Government and therefore sworn to secrecy. By the end of 1969, Ellis appears to have reached the same impasse that the Stanford trio would reach in 1975. He had proved to himself that public key cryptography (or nonsecret encryption, as he called it) was possible, and he had developed the concept of separate public keys and private keys. He also knew that he needed to find a special one-way function, one that could be reversed if the receiver had access to a piece of special information. Unfortunately, Ellis was not a mathematician.

He experimented with a few mathematical functions, but he soon realized that he would be unable to progress any further on his own.

At this point, Ellis revealed his breakthrough to his bosses. Their reactions are still classified material, but in an interview Richard Walton was prepared to paraphrase for me the various memoranda that were exchanged. Sitting with his briefcase on his lap, the lid shielding the papers from my view, he flicked through the documents:

> I can't show you the papers that I have in here because they still have naughty words like TOP SECRET stamped all over them. Essentially, James's idea goes to the top man, who farms it out, in the way that top men do, so that the experts can have a look at it. They state that what James is saying is perfectly true. In other words, they can't write this man off as a crank. At the same time they can't think of a way of implementing his idea in practice. And so they're impressed by James's ingenuity, but uncertain as to how to take advantage of it.

For the next three years, GCHQ's brightest minds struggled to find a one-way function that satisfied Ellis's requirements, but nothing emerged. Then, in September 1973, a new mathematician joined the team. Clifford Cocks had recently graduated from Cambridge University, where he had specialized in number theory, one of the purest forms of mathematics. When he joined GCHQ he knew very little about encryption and the shadowy world of military and diplomatic communication, so he was assigned a mentor, Nick Patterson, who guided him through his first few weeks at GCHQ.

After about six weeks, Patterson told Cocks about "a really whacky idea." He outlined Ellis's theory for public key cryptography, and explained that nobody had yet been able to find a mathematical function that fitted the bill. Patterson was telling Cocks because this was the most titillating cryptographic idea around, not because he expected him to try to solve it. However, as Cocks explains, later that day he set to work: "There was nothing particular happening, and so I thought I would think about the idea. Because I had been working in number theory, it was natural to think about one-way functions, something you could do but not undo. Prime numbers and factoring was a natural candidate, and that became my starting point." Cocks was beginning to formulate what would later be known as the RSA asymmetric cipher. Rivest, Shamir and Adleman discovered their formula for public key cryptography in 1977, but four years earlier the young Cambridge graduate was going through exactly the same thought processes. Cocks recalls: "From start to finish, it took me no more than half an hour. I was quite pleased with myself. I thought, 'Ooh, that's nice. I've been given a problem, and I've solved it.'"

Cocks did not fully appreciate the significance of his discovery. He was unaware of the fact that GCHQ's brightest minds had been struggling with the problem for three years, and had no idea that he had made one of the most important cryptographic breakthroughs of the century. Cocks's naivety may have been part of the reason for his success, allowing him to attack the problem with confidence, rather than timidly prodding at it. Cocks told his mentor about his discovery, and it was Patterson who then reported it to the management. Cocks was quite diffident and very much still a rookie, whereas Patterson fully appreciated the context of the

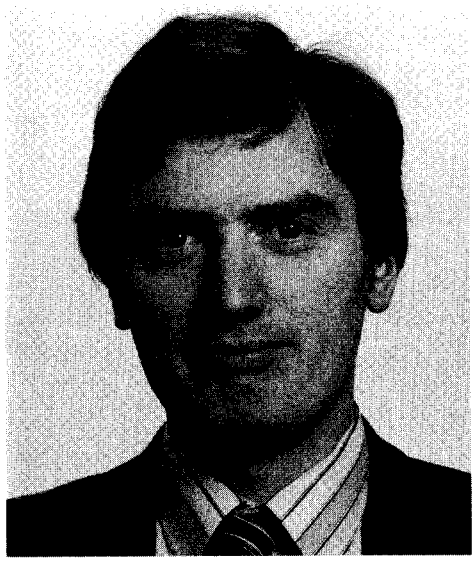

Figure 67 Clifford Cocks.

problem and was more capable of addressing the technical questions that would inevitably arise. Soon complete strangers started approaching Cocks, the wonderkid, and began to congratulate him. One of the strangers was James Ellis, keen to meet the man who had turned his dream into a reality. Because Cocks still did not understand the enormity of his achievement, the details of this meeting did not make a great impact on him, and so now, over two decades later, he has no memory of Ellis's reaction.

When Cocks did eventually realize what he had done, it struck him that his discovery might have disappointed G.H. Hardy, one of the great English mathematicians of the early part of the century. In his *The Mathematician's Apology*, written in 1940, Hardy had proudly stated: "Real mathematics has no effects on war. No one has yet discovered any war-like purpose to be served by the theory of numbers." Real mathematics means pure mathematics, such as the number theory that was at the heart of Cocks's work. Cocks proved that Hardy was wrong. The intricacies of number theory could now be used to help generals plan their battles in complete secrecy. Because his work had implications for military communications, Cocks, like Ellis, was forbidden from telling anybody outside GCHQ about what he had done. Working at a top-secret government establishment meant that he could tell neither his parents nor his former colleagues at Cambridge University. The only person he could tell was his wife, Gill, since she was also employed at GCHQ.

Although Cocks's idea was one of GCHQ's most potent secrets, it suffered from the problem of being ahead of its time. Cocks had discovered a mathematical function that permitted public key cryptography, but there was still the difficulty of implementing the system. Encryption via public key cryptography requires much more computer power than encryption via a symmetric cipher like DES. In the early 1970s, computers were still relatively primitive and unable to perform the process of public key encryption within a reasonable amount of time. Hence, GCHQ were not in a position to exploit public key cryptography. Cocks and Ellis had proved that the apparently impossible was possible, but nobody could find a way of making the possible practical.

At the beginning of the following year, 1974, Cocks explained his work on public key cryptography to Malcolm Williamson, who had recently joined GCHQ as a cryptographer. The men happened to be old friends.

They had both attended Manchester Grammar School, whose school motto is *Sapere aude*, "Dare to be wise." While at school in 1968, the two boys had represented Britain at the Mathematical Olympiad in the Soviet Union. After attending Cambridge University together, they went their separate ways for a couple of years, but now they were reunited at GCHQ. They had been exchanging mathematical ideas since the age of eleven, but Cocks's revelation of public key cryptography was the most shocking idea that Williamson had ever heard. "Cliff explained his idea to me," recalls Williamson, "and I really didn't believe it. I was very suspicious, because this is a very peculiar thing to be able to do."

Williamson went away, and began trying to prove that Cocks had made a mistake and that public key cryptography did not really exist. He probed the mathematics, searching for an underlying flaw. Public key cryptography seemed too good to be true, and Williamson was so determined to find a mistake that he took the problem home. GCHQ

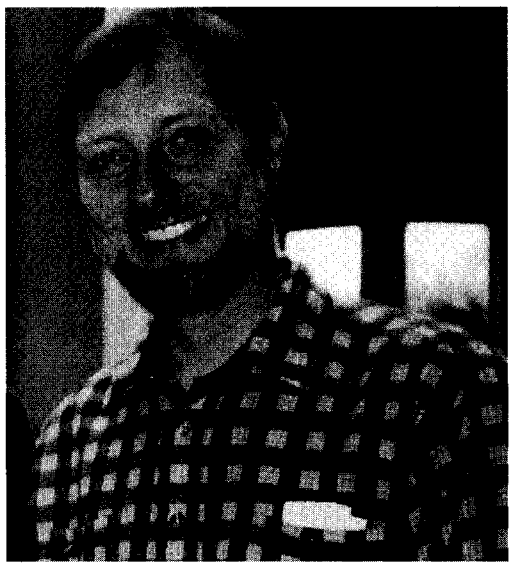

Figure 68 Malcolm Williamson.

employees are not supposed to take work home, because everything they do is classified, and the home environment is potentially vulnerable to espionage. However, the problem was stuck in Williamson's brain, so he could not avoid thinking about it. Defying orders, he carried his work back to his house. He spent five hours trying to find a flaw. "Essentially I failed," says Williamson. "Instead I came up with another solution to the problem of key distribution." Williamson was discovering Diffie–Hellman–Merkle key exchange, at roughly the same time that Martin Hellman discovered it. Williamson's initial reaction reflected his cynical disposition: "This looks great, I thought to myself. I wonder if I can find a flaw in this one. I guess I was in a negative mood that day."

By 1975, James Ellis, Clifford Cocks and Malcolm Williamson had discovered all the fundamental aspects of public key cryptography, yet they all had to remain silent. The three Britons had to sit back and watch as their discoveries were rediscovered by Diffie, Hellman, Merkle, Rivest, Shamir and Adleman over the next three years. Curiously, GCHQ discovered RSA before Diffie–Hellman–Merkle key exchange, whereas in the outside world, Diffie–Hellman–Merkle key exchange came first. The scientific press reported the breakthroughs at Stanford and MIT, and the researchers who had been allowed to publish their work in the scientific

Figure 69 Malcolm Williamson (second from left) and Clifford Cocks (extreme right) arriving for the 1968 Mathematical Olympiad.

journals became famous within the community of cryptographers. A quick look on the Internet with a search engine turns up 15 Web pages mentioning Clifford Cocks, compared to 1,382 pages that mention Whitfield Diffie. Cocks's attitude is admirably restrained: "You don't get involved in this business for public recognition." Williamson is equally dispassionate: "My reaction was 'Okay, that's just the way it is.' Basically, I just got on with the rest of my life."

Williamson's only qualm is that GCHQ failed to patent public key cryptography. When Cocks and Williamson first made their breakthroughs, there was agreement among GCHQ management that patenting was impossible for two reasons. First, patenting would mean having to reveal the details of their work, which would have been incompatible with GCHQ's aims. Second, in the early 1970s it was far from clear that mathematical algorithms could be patented. When Diffie and Hellman tried to file for a patent in 1976, however, it was evident that they could be patented. At this point, Williamson was keen to go public and block Diffie and Hellman's application, but he was overruled by his senior managers, who were not farsighted enough to see the digital revolution and the potential of public key cryptography. By the early 1980s Williamson's bosses were beginning to regret their decision, as developments in computers and the embryonic Internet made it clear that RSA and Diffie–Hellman–Merkle key exchange would both be enormously successful commercial products. In 1996, RSA Data Security, Inc., the company responsible for RSA products, was sold for $200 million.

Although the work at GCHQ was still classified, there was one other organization that was aware of the breakthroughs that had been achieved in Britain. By the early 1980s America's National Security Agency knew about the work of Ellis, Cocks and Williamson, and it is probably via the NSA that Whitfield Diffie heard a rumor about the British discoveries. In September 1982, Diffie decided to see if there was any truth in the rumor, and he traveled with his wife to Cheltenham in order to talk to James Ellis face-to-face. They met at a local pub, and very quickly Mary was struck by Ellis's remarkable character:

> We sat around talking, and I suddenly became aware that this was the most wonderful person you could possibly imagine. The breadth of his

> mathematical knowledge is not something I could confidently discuss, but he was a true gentleman, immensely modest, a person with great generosity of spirit and gentility. When I say gentility, I don't mean old-fashioned and musty. This man was a *chevalier*. He was a good man, a truly good man. He was a gentle spirit.

Diffie and Ellis discussed various topics, from archaeology to how rats in the barrel improve the taste of cider, but whenever the conversation drifted toward cryptography, Ellis gently changed the subject. At the end of Diffie's visit, as he was ready to drive away, he could no longer resist directly asking Ellis the question that was really on his mind: "Tell me about how you invented public key cryptography?" There was a long pause. Ellis eventually whispered: "Well, I don't know how much I should say. Let me just say that you people did much more with it than we did."

Although GCHQ were the first to discover public key cryptography, this should not diminish the achievements of the academics who rediscovered it. It was the academics who were the first to realize the potential of public key encryption, and it was they who drove its implementation. Furthermore, it is quite possible that GCHQ would never have revealed their work, thus blocking a form of encryption that would enable the digital revolution to reach its full potential. Finally, the discovery by the academics was wholly independent of GCHQ's discovery, and on an intellectual par with it. The academic environment is completely isolated from the top-secret domain of classified research, and academics do not have access to the tools and secret knowledge that may be hidden in the classified world. On the other hand, government researchers always have access to the academic literature. One might think of this flow of information in terms of a one-way function–information flows freely in one direction, but it is forbidden to send information in the opposite direction.

When Diffie told Hellman about Ellis, Cocks and Williamson, his attitude was that the discoveries of the academics should be a footnote in the history of classified research, and that the discoveries at GCHQ should be a footnote in the history of academic research. However, at that stage nobody except GCHQ, NSA, Diffie and Hellman knew about the classified research, and so it could not even be considered as a footnote.

By the mid-1980s, the mood at GCHQ was changing, and the management considered publicly announcing the work of Ellis, Cocks and Williamson. The mathematics of public key cryptography was already well established in the public domain, and there seemed to be no reason to remain secretive. In fact, there would be distinct benefits if the British revealed their groundbreaking work on public key cryptography. As Richard Walton recalls:

> We flirted with the idea of coming clean in 1984. We began to see advantages for GCHQ being more publicly acknowledged. It was a time when the government security market was expanding beyond the traditional military and diplomatic customer, and we needed to capture the confidence of those who did not traditionally deal with us. We were in the middle of Thatcherism, and we were trying to counter a sort of "government is bad, private is good" ethos. So, we had the intention of publishing a paper, but that idea was scuppered by that blighter Peter Wright, who wrote *Spycatcher*. We were just warming up senior management to approve this release, when there was all this hoo-ha about *Spycatcher*. Then the order of the day was "heads down, hats on."

Peter Wright was a retired British intelligence officer, and the publication of *Spycatcher*, his memoirs, was a source of great embarrassment to the British government. It would be another 13 years before GCHQ eventually went public–28 years after Ellis's initial breakthrough. In 1997 Clifford Cocks completed some important unclassified work on RSA, which would have been of interest to the wider community, and which would not be a security risk if it were to be published. As a result, he was asked to present a paper at the Institute of Mathematics and its Applications Conference to be held in Cirencester. The room would be full of cryptography experts. A handful of them would know that Cocks, who would be talking about just one aspect of RSA, was actually its unsung inventor. There was a risk that somebody might ask an embarrassing question, such as "Did you invent RSA?" If such a question arose, what was Cocks supposed to do? According to GCHQ policy he would have to deny his role in the development of RSA, thus forcing him to lie about an issue that was totally innocuous. The situation was clearly ridiculous, and GCHQ decided that it was time to change its policy. Cocks was given permission to begin his talk by presenting a brief history of GCHQ's contribution to public key cryptography.

On December 18, 1997, Cocks delivered his talk. After almost three decades of secrecy, Ellis, Cocks and Williamson received the acknowledgment they deserved. Sadly, James Ellis had died just one month earlier on November 25, 1997, at the age of seventy-three. Ellis joined the list of British cipher experts whose contributions would never be recognized during their lifetimes. Charles Babbage's breaking of the Vigenère cipher was never revealed during his lifetime, because his work was invaluable to British forces in the Crimea. Instead, credit for the work went to Friedrich Kasiski. Similarly, Alan Turing's contribution to the war effort was unparalleled, and yet government secrecy demanded that his work on Enigma could not be revealed.

In 1987, Ellis wrote a classified document that recorded his contribution to public key cryptography, which included his thoughts on the secrecy that so often surrounds cryptographic work:

> Cryptography is a most unusual science. Most professional scientists aim to be the first to publish their work, because it is through dissemination that the work realizes its value. In contrast, the fullest value of cryptography is realized by minimizing the information available to potential adversaries. Thus professional cryptographers normally work in closed communities to provide sufficient professional interaction to ensure quality while maintaining secrecy from outsiders. Revelation of these secrets is normally only sanctioned in the interests of historical accuracy after it has been demonstrated that no further benefit can be obtained from continued secrecy.