Chapter 9

Secret Codes as Munitions and Money

Encryption Becomes Unbreakable

9.1 Senator Gregg Reconsiders

September 13, 2001: Fires were still smoldering in the wreckage of the World Trade Center when Judd Gregg of New Hampshire rose to tell the Senate what had to happen. He recalled the warnings issued by the FBI years before the country had been attacked: that the FBI's most serious problem was "the encryption capability of the people who have an intention to hurt America." "It used to be," the senator went on, "that we had the capability to break most codes because of our sophistication."[1] No more. "The technology has outstripped the code breakers,"[2] he warned. Even civil libertarian cryptographer Phil Zimmermann agreed that the terrorists were probably encoding their messages. Zimmermann's software had been posted on the Internet in 1991 for use by human rights workers around the world, but he had to acknowledge that the bad guys also "would want to hide their activities using encryption."[3]

Encryption is the art of encoding messages so they can't be understood by eavesdroppers or adversaries into whose hands the messages might fall. De-scrambling an encrypted message requires knowing the sequence of symbols—the "key"—that was used to encrypt it. An encrypted message may be visible to all the world, but without the key, it may as well be hidden in a locked box.

What was needed, Senator Gregg asserted, was "the cooperation of the community that is building the software, producing the software, and building the equipment that creates the encoding technology." Cooperation, that is, enforced by legislation. Whoever made encryption software, Senator Gregg proposed, would have to enable the government to bypass the locks and retrieve the decrypted messages. What about encryption programs written abroad, which could be shared around the world in the blink of an eye, as Zimmermann's had been? The US should use "the market of the United States as leverage" in getting foreign manufacturers to follow requirements for "back doors" that could be used by the US government.

By September 27 Gregg's legislation was beginning to take shape. The keys used to encrypt messages would be held in escrow by the government under tight security. There would be a "quasi-judicial entity," appointed by the Supreme Court, that would decide when law enforcement had made its case for release of the keys. Civil libertarians squawked, and doubts were raised as to whether the key escrow idea could actually work. No matter, opined the Senator in late September. "Nothing's ever perfect. If you don't try, you're never going to accomplish it."[4]

And then abruptly, Senator Gregg dropped his legislative plan. "We are not working on an encryption bill," said the Senator's spokesman on October 17.[5]

On October 24 Congress passed the USA PATRIOT Act, giving the FBI sweeping new powers to combat terrorism. But the PATRIOT Act does not even mention encryption. No serious attempt has been made to legislate control over cryptographic software since Gregg's proposal. Why not?

9.2 Why Not Regulate Encryption?

Throughout the decade of the 1990s, the FBI had made control of encryption its top legislative priority. Senator Gregg's proposal was a milder form of a bill, drafted by the FBI and reported out favorably by the House Select Committee on Intelligence in 1997, that would have mandated a five-year prison sentence for selling encryption products unless they enabled immediate decryption by authorized officials.[6]

How could regulatory measures deemed critical for fighting terrorism by US law enforcement in 1997 drop completely off the legislative agenda four years later—in the aftermath of the worst terrorist attack ever suffered by the United States of America?

No technological breakthrough in cryptography in the fall of 2001 had legislative significance. There were no diplomatic breakthroughs either. Nothing else transpired to make the use of encryption by terrorists and criminals unimportant. It was just that something else about encryption had become *more* important. And that was to ensure that encryption tools could be in the hands of banks and their customers, airlines and their customers, Ebay and Amazon and L. L. Bean and their customers. That is, in the hands of anyone using the Internet for commerce.

For a decade, government officials had been debating the tension between secure conduct of electronic commerce and secret communication among outlaws. Senator Gregg was but the last of the voices calling for restrictions on encryption. The National Research Council had issued a report of nearly 700 pages in 1996 weighing the alternatives. The report concluded that on balance, efforts to control encryption would be ineffective, and that their costs would exceed any imaginable reward.[7] The intelligence and defense establishment remained unpersuaded. FBI Director Louis Freeh testified before Congress in 1997 that uncontrolled public access to encryption "ultimately will devastate our ability to fight crime and prevent terrorism."[8]

Yet only four years later, even in the face of the September 11$^{\text{th}}$ attack, electronic commerce demanded encryption software for every business in the country and every home computer from which a commercial transaction might take place. At the moment when Freeh was cautioning

Congress about encryption software, elected officials might never have bought anything on line and their families might never have used computers. By 2001, computers had become consumer appliances, Internet connections were common in American homes—and average citizens were well aware of electronic fraud. Consumers did not want their credit card numbers and social security numbers exposed to everyone on the Internet.

Why is encryption so important to Internet communications that Congress was willing risk terrorists using encryption, so that American businesses and consumers could use it too? After all, information security is not a new idea. People communicating by postal mail have reasonable assurances of privacy without any use of encryption.

The Internet is different from the postal system, despite the metaphor of electronic "mail." Data packets zipping across the Net are not like envelopes with an address on the outside and contents sealed inside. Packets are more like postcards, with everything exposed for anyone to see. Every data packet passing through the Internet gets handled at every router: stored, examined, checked, analyzed, and sent on its way. The routers are not under any form of central control or security certification. Even if the routers could be controlled and all the fibers and wires subject to wiretap regulations, wireless networks allow bits to be grabbed out of the air without detection. Indeed, by 2001, a lot of bits were traveling though the air, and snoopers could easily look at them.

The way to make Internet communications secure—to make sure that no one but the intended recipient knows what is in a message—is for the sender to encrypt the information so that only the recipient can decrypt it. If the contents of data packets are encrypted, then routers, sniffers, and eavesdroppers along the route from sender to receiver can examine the packets all they want. All they will find is an undecipherable scramble of bits.

In 2001, electronic commerce accounted for less than 1% of retail sales in the US. That percentage has grown to 3% today, around $130 billion. A great deal of the money that fuels the American economy now moves from consumers to businesses, and between businesses, only as bits. Altering those bits would be tantamount to stealing money.[9]

As the world awakened to Internet commerce, encryption could no longer be thought of as it had been from ancient times until the turn of the third millennium: as armor used by generals and diplomats to protect information critical to national security. Suddenly, encryption was more like the armored cars used to transport cash on city streets, except that *everyone* needed these armored cars. Encryption was no longer a munition; it was more like money.

The commoditization of a critical military tool was more than a technology shift. It sparked reconsideration of fundamental notions of privacy and of the tension between security and freedom in a democratic society.

"The question," posed MIT's Ron Rivest, one of the world's leading cryptographers, during one of the many debates over encryption policy, "is whether people should be able to conduct private conversations, immune from government surveillance, even when that surveillance is fully authorized by a Court order."[10] By 2001 commercial realities had overtaken such debates. The same technology that protects your credit card numbers when you place an order over the web also enables you to conspire to overthrow the government without the government knowing what you are saying.

To fit the needs of electronic commerce, encryption software had to be widely available and had to work perfectly and quickly. And although encryption was more than four millennia old, no method known until the late twentieth century would have worked for Internet commerce. An astonishing mathematical discovery in the 1970s changed everything. Suddenly every man, woman, and child could transmit credit card numbers to Amazon more securely than any Second World War general had been able to communicate military orders on which the fate of nations depended.

9.3 Historical cryptography

Cryptography—"secret writing"—has been around almost as long as writing itself.

A Roman historian described how Julius Caesar's encrypted his letters to the orator Cicero, with whom Caesar was plotting in the dying days of the Roman Republic: "...if he [Caesar] had anything confidential to say, he wrote it in cipher, that is, by so changing the order of the letters of the alphabet, that not a word could be made out. If anyone wishes to decipher these, and get at their meaning, he must substitute the fourth letter of the alphabet, namely D, for A, and so with the others."[11] In other words, Caesar used a letter by letter translation to encrypt his messages:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
ZYXWVUTSRQPONMLKJIHGFEDCBA
```

To encrypt a message with Caesar's method, replace each letter in the top row by the corresponding letter in the bottom row. For example, the opening of Caesar's Commentaries "Omnia Gallia in tres partes divisa est" would be encrypted as:

```
Plaintext:  OMNIA GALLIA IN TRES PARTES DIVISA EST
Ciphertext: LJKFX DXIIFX FK QOBP MXOQBP AFSFPX BPQ
```

The original message is called the *plaintext* and the encoded message is called the *ciphertext.* Messages are decrypted by doing the reverse substitutions.

This method is called the *Caesar shift* or the *Caesar cipher.* The encryption/decryption rule is easy to remember: "Shift the alphabet three places." Of course the same idea would work if the alphabet were shifted more than three places, or fewer, so the Caesar cipher is really a family of ciphers, with 25 possible variations, one for each different amount of shifting.[12]

Caesar ciphers are very simple, and an enemy who knew that that Caesar was simply shifting the plaintext could easily try all the 25 possible shifts of the alphabet to decrypt the message. But Caesar's method is a representative of a larger class of ciphers, called *substitution ciphers,* in which one symbol is substituted for another according to a uniform rule (the same letter is always translated the same way).[13]

There are a great many more substitution ciphers than just shifts. For example, we could scramble the letters according to the rule:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
XAPZRDWIBMQEOFTYCGSHULJVKN
```

A becomes X, B becomes A, C becomes P, and so on. There is a similar substitution for every way of reordering the letters of the alphabet. The sequence `XAPZRDWIBMQEOFTYCGSHULJVKN` is the key—know it and you can decipher the message, but if you don't know it, trying different possibilities would take a very long time! The number of different reorderings is

$$26 \times 25 \times 24 \times \cdots \times 3 \times 2$$

which is about 4×10^{26} different keys—ten thousand times the number of stars in the universe! So substitution ciphers must be secure. Or so it might seem.

9.4 Breaking substitution ciphers

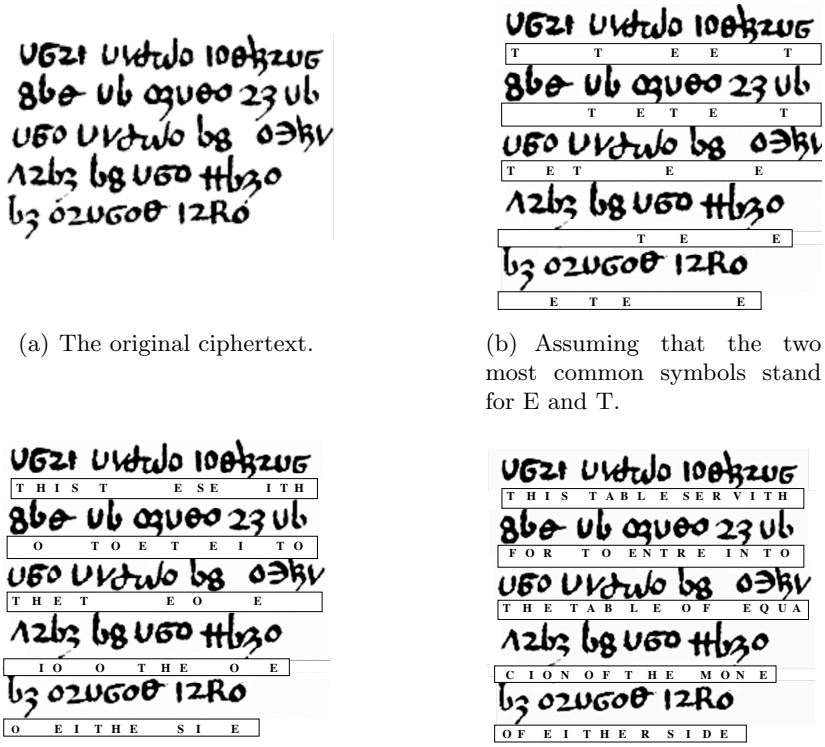
Geoffrey Chaucer was the greatest English poet before the Renaissance, and he was a scientific writer to boot. He wrote a manual for use of the Astrolabe, an instrument for predicting the positions of the sun and stars. Parts of his writings on the Astrolabe, including the passage shown in Figure 9.1(a), were written in a substitution cipher. This puzzle turns out to be not as hard as it looks. We know it is written in English—well, Middle English, but let's see how far we can get thinking of it as encrypted English.

Though it looks like gibberish, it does contain some patterns that may be clues. For example, some symbols occur more frequently than others. There are twelve ◆s and ten ∪s, and no other symbol occurs as frequently as these. In ordinary English texts the two most frequently occurring letters are E and T, so a fair guess is that these two symbols correspond to these two letters. Figure 9.1(b) shows what happens if we assume that ◆ = E and ∪ = T. The pattern ∪6◆ appears twice and apparently represents a three letter word beginning with T and ending with E. It could be TIE or TOE, but THE seems more likely, so a reasonable assumption is that 6 = H. If that is true, what is the four letter word at the beginning of the text, which begins with TH? Not THAT, because it ends with a new symbol, nor THEN, because the third letter is also new. Perhaps THIS. And there is a two-letter word beginning with T that appears twice in the second line—that must be TO. Filling in the equivalencies for H, I, S, and O yields Figure 9.1(c).

At this point the guessing gets easier — probably the last two words are EITHER SIDE — and the last few symbols can be inferred with a knowledge of Middle English and some idea of what the text is about. The complete plaintext is: *This table servith for to entre in to the table of equacion of the mone on either side* (Figure 9.1(d)).

The technique used to crack Chaucer's code is *frequency analysis*. Since the cipher is a simple substitution of symbols for letters, crucial information about which symbols represent which letters can be gathered from how often the various symbols appear in the ciphertext. This idea was first described by the Arabic philosopher and mathematician Al-Kindi, who lived in Baghdad in the 9^{th} century.

By the Renaissance, this kind of informed guesswork had been reduced to a fine art which was well known to European governments. In a famous example of the insecurity of substitution ciphers, Mary Queen of Scots was beheaded in 1587 due to her misplaced reliance on a simple substitution cipher to conceal her correspondence with plotters against Queen Elizabeth I. She was not the last



(a) The original ciphertext.

(b) Assuming that the two most common symbols stand for E and T.

(c) More conjectural decodings.

(d) Fully decoded.

Figure 9.1: Chaucer's cipher.

to have put too much confidence in an encryption scheme that looked hard to crack, but wasn't. Substitution ciphers were in common use as late as the 1800s, even though they had been insecure for a millennium by that time!

9.5 Secret Keys and One-Time Pads

In cryptography, every advance in code-breaking yields an innovation in code-making. Given how easily Chaucer's code was broken, what could make it more secure, or *stronger,* as cryptographers would say? A method named for the 16th century French diplomat Blaise de Vigenère uses multiple Caesar ciphers. For example, we can pick ten Caesar ciphers and use the first cipher for encrypting the first, eleventh, 21^{st}, 31^{st}, letters of the plaintext, the second cipher for encrypting the second, twelfth, 22^{nd}, plaintext letters, and so on.

Figure 9.2 shows a Vigenère cipher based on eight Caesar substitutions. A plaintext message beginning SECURE... would be encrypted to produce the cyphertext USFYSM..., as indicated by the underlined characters in the figure—S is encrypted using the first row, E encrypted using the second row, and so on. After we use the bottom row of the table, we start again at the top row, and repeat the process over and over.

We can use the cipher of Figure 9.2 without having to send our correspondent the entire table.

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

Figure 9.2: Vigènere cipher.

Scanning down the first column spells out CODEBITS, which is the key for the message. To communicate using Vigenère encryption, the correspondents must first agree on a key. Then they use the key to construct a substitution table for encrypting and decrypting messages.

Observe in this example that the two occurrences of E at the 2^{nd} and 6^{th} positions in the plaintext are represented by different ciphertext letters, and the two occurrences of the ciphertext letter S represent different plaintext letters. This illustrates how the Vigenère cipher confounds simple frequency analysis, which was the main tool of cryptanalysts at the time. Although the idea may seem simple, the discovery of the Vigenère cipher is regarded as a fundamental advance in cryptography, and the method was considered to be unbreakable for hundreds of years.

The Vigenère cipher was actually broken in the mid 1800s by the English mathematician Charles Babbage, who is now recognized as a founding figure in the field of computing. Babbage realized that if one could guess or infer the length of the key, and hence the length of the cycle on which the Vigenère cipher is repeated, a more sophisticated frequency analysis would crack the code. Babbage never published his technique, perhaps at the request of British Intelligence. His contribution lay undiscovered until 1970, and the first public description was provided independently in 1863 by a Prussian Army officer, William Kasiski. Since then, the Vigenère cipher has been insecure.

The one way to encrypt messages that is guaranteed to be immune from any kind of mathematical analysis is to use a key that is as long as the plaintext. With no repetitions, there would be nothing to analyze. For example, if we wanted to encrypt a message of length 100 we might use 100 shift ciphers in an arrangement like that of Figure 9.2, extended to 100 rows, and every table row would be used only once. A code like this is commonly referred to as a *one-time pad.*[14]

The term "one-time pad" is based on a particular way of physically implementing the cipher. Let's imagine that Bob wants to get a message to Alice. Alice and Bob have identical pads of printed paper. Each page of the pad has a key. Bob uses the top page to encrypt a message. When Alice receives it, she uses the top page of his pad to decrypt the message. Both Alice and Bob tear off and destroy the top page of the pad when they have used it. It is essential that the pages not be re-used, as doing so could create patterns on which a frequency analysis could be performed.

Figure 9.3: One-time pad.

Figure 9.3 shows a one-time pad used by Soviet KGB agents, seized by British Intelligence. The revolutionary Che Guevara used one-time pads for exchanging messages with Fidel Castro. The pad shown in was found on Guevara's body after had had been killed by the Bolivian army in 1967. It uses a simple code to translate letters into sequences of one or two digits. Governments still use one-time pads today for sensitive communications, with large amounts of keying material carefully generated and distributed on CDs or DVDs.

A one-time pad, if used correctly, cannot be broken by cryptanalysis. There are simply no patterns to be found in the ciphertext. The one-time pad is, in principle, as good as it gets in cryptography. It is absolutely unbreakable—in theory.

But as Yogi Berra said, "In theory there is no difference between theory and practice. In practice there is." Transmitting a pad between the parties without detection is just as difficult as it would be to communicate the plaintext of the message itself. Typically the parties would share a pad ahead of time and hope to conceal it in their travels. But big pads are harder to conceal than small pads, so the temptation arises to re-use pages—the kiss of death for security.

The Soviet KGB fell victim to exactly this temptation, which led to the partial or complete decryption of over 3000 diplomatic and espionage messages by US and British intelligence during the years 1942–1946.[15] The National Security Agency's VENONA project, publicly revealed only in 1995, was responsible for exposing major KGB agents such as Klaus Fuchs and Kim Philby. The Soviet messages were doubly encrypted, using a one-time pad on top of other techniques, which made the codebreaking project enormously difficult. It was successful only because, as World War II wore on and material conditions deteriorated, the Soviets re-used the pads.

9.6 Lessons for the Internet Age

All ciphers developed until recently are distant descendants of substitution methods. In computer implementations of these ciphers, the ASCII-encoded plaintext message is divided into blocks and the bits in the block are transformed according to some method that depends on a key. The key is itself a sequence of bits which the communicating parties must share but must keep secret from everyone else. No shortcuts are known (publicly, at least) for breaking industrial-strength ciphers such as DES, the Data Encryption Standard, which became a national standard in the 1970s. The

only way to decrypt a ciphertext without knowing the secret key is by brute-force exhaustive search, trying all possible keys.

The amount of computation required to break a cipher by exhaustive search grows exponentially in the size of the key. Increasing the key length by one bit doubles the amount of work required to break the cipher, but only slightly increases the work required to encrypt and decrypt. This is what makes these ciphers so useful: the work required to break the cipher can be made to grow exponentially by picking longer and longer keys.

The history of secret communication has important lessons for today's Internet, even with the vast computing power now available for encryption and cryptanalysis. These lessons are as valid today as in the days of simple substitution ciphers, and yet they are often ignored.

Breakthroughs can render previously reliable cryptographic methods insecure. And news of cryptanalytic breakthroughs sometimes travels slowly. Al-Kindi's method of frequency analysis undermined the security of simple substitution codes, but they remained in use for a thousand years afterwards. When Babbage and Kasiski broke the Vigenère cipher in the mid 19th century, state secrets were compromised. Yet 50 years later, Scientific American still described the Vigenère method as "impossible of translation."[16]

Similarly, there are no guarantees that even the best contemporary ciphers won't be broken, or haven't been broken already. Some of the ciphers have the potential to be validated by mathematical proofs, but establishing those proofs seems to depend on deep breakthroughs in problems in pure mathematics. If anyone can break the modern codes, it is probably the National Security Agency or a comparable agency of a foreign government, and those folks don't tend to say much publicly.

In the absence of a formal proof of security, all one can do is to rely on what has been dubbed the[17] **Fundamental Tenet of Cryptography.** *If lots of smart people have failed to solve a problem, then it probably won't be solved (soon).*

Of course that is not a very useful principle in practice, since by definition, breakthroughs are unlikely to happen "soon." But they do happen, and when they do, indigestion among cryptographers is widespread. In August 2004, at an annual cryptography conference, researchers announced that they had been able to break a popular algorithm (MD5) for computing cryptographic operations called message digests, which are fundamental security elements in almost all Web servers, password programs, and office products. Cryptographers recommended switching to a stronger algorithm (SHA-1) but within a year, weaknesses were uncovered in this method as well.[18]

Making strong encryption systems available does not guarantee they will be used. Vigenère published his encryption method in 1586. But foreign-office cipher secretaries commonly avoided the Vigenère cipher because it was cumbersome to use. They stayed with simple substitution ciphers, hoping for the best even though it was well-known that these ciphers were readily broken. By the 18th century, most European governments had skilled "Black Chambers" through which all mail to and from foreign embassies was routed for decryption as a matter of course. Finally the embassies switched to Vigenère ciphers, which themselves continued to be used after information about how to crack them had become widely known.

And so it is today. Technological inventions, no matter how solid in theory, will not be used for everyday purposes if they are inconvenient or expensive. The risks of weak systems are often rationalized away in attempts to avoid the trouble of switching to more secure alternatives.

The encryption standard for 802.11 wireless computer communication introduced in 1999 (WEP—Wired Equivalent Privacy) was found in 2001 to have serious flaws that made it easy to eavesdrop on wireless networks, a result that became widely known in the security community.[19] Despite this, wireless equipment companies continued to sell WEP products, while industry pundits comforted people that "WEP is better than nothing." The key length was doubled, from 64 bits to 128 bits, to make WEP harder to crack. But a longer key length was an inadequate response to the underlying problem: cracking the code did not require an exponentially costly exhaustive search. A new standard (WPA—Wi-Fi Protected Access) was finally introduced in 2002, but it wasn't until September 2003 that products were required to use the new standard in order to be certified.

Similarly, many of today's "smart card" systems that use RFID (Radio Frequency Identification) tags are insecure. In January 2005, computer scientists from Johns Hopkins University and RSA Data Security announced that they had cracked an RFID-based automobile anti-theft and electronic payment system built into millions of automobile key tags, and they demonstrated this by making multiple gasoline purchases at an Exxon/Mobile station. A spokesman for Texas Instruments, which developed the system, countered that the methods the team used were "wildly beyond the reach of most researchers," saying "I don't see any reason to change this approach."[20]

Kerckhoffs's Principle: The Enemy Knows Your System. The third lesson from history is counterintuitive. It is that a cryptographic method should be regarded as more reliable if it is widely known and seems not to have been broken, than if the method has been kept secret.

The Flemish linguist Auguste Kerckhoffs articulated this principle in an 1883 essay on military cryptography.[21] As he explained it, "The system must not require secrecy, and it could fall into the hands of the enemy without causing trouble.... [I]f a system requiring secrecy were to find itself in the hands of too many individuals, it could be compromised upon each engagement in which any of them take part"[22] In other words, if a cryptographic method is put in widespread use, the method is bound not to remain secret for long. It should therefore be designed so that it will remain secure even if everything but the key becomes exposed.

Kerckhoffs's Principle is frequently violated in modern Internet security practice. Internet start-up security companies routinely make bold announcements about some new breakthrough proprietary encryption method, which they refuse to subject to public scrutiny, explaining that the method must be kept secret in order to protect its security. Cryptographers generally regard such "security though obscurity" claims with extreme skepticism.

Even well-established organizations run afoul of Kerckhoffs's Principle. The Content Scrambling System (CSS) used on DVDs (Digital Versatile Disks) was developed by a consortium of motion picture studios and consumer electronics companies in 1996. It encrypts DVD contents in order to limit unauthorized copying. The method was kept secret to prevent the manufacture of unlicensed DVD players,[23] but as a result it was never publicly scrutinized by encryption experts. The algorithm turned out to be flawed, and was cracked by a 15-year-old. Today, CSS decryption programs, together with numerous unauthorized "ripped" DVD contents, circulate widely on the Internet.

Adherence to Kerckhoffs's Principle has been institutionalized in the form of public encryption standards. DES is one such standard, widely used in business and finance. DES has survived all attempts to crack it. The progress of Moore's Law has made searching through all possible keys more feasible in recent years, so a newer standard, Advanced Encryption Standard or AES, was adopted in 2002 after a thorough review.[24] It is precisely because these encryption methods are so widely known that confidence in them can be high. They have been subjected to both professional analysis and amateur experimentation, and no serious deficiencies have been discovered.

9.7 Secrecy Suddenly Changes Forever

Let's again personify the communication problem by imagining that Bob wants to send a secret message to Alice. Unhappily, an adversary is trying to eavesdrop on what Bob is telling Alice—let's use "Eve" for the name of the evil eavesdropper.

For four thousand years, cryptography has been about making sure Eve could not read Bob's message if Eve intercepted it *en route.* All the effort was on making sure that that the plaintext could not be recovered from the ciphertext if the key was kept secret. All bets were off if the key was discovered. Keeping the key secret was essential, and in practice was a very uncertain business.

If Alice and Bob worked out the key when they met, how could they keep the key secret during their dangerous travels? Protecting keys was a military and diplomatic priority of supreme importance. Pilots and soldiers were trained that, even in the face of certain death from enemy attack, their first responsibility was to destroy their codebooks, since discovery of the codes could cost thousands of lives. The secrecy of the codes was everything.

More fundamentally, if Alice and Bob never met, then how could they agree on a key without *already* having a secure method for transmitting the key? That seemed a fundamental limitation: secure communication through cryptography was possible only for people who could meet beforehand, or who already had access to a method of secure communication for carrying the key between them.

And then, in the 1970s, everything changed. Whitfield Diffie was a 32-year-old mathematical free spirit who had been obsessed with cryptography since his years as an MIT undergraduate. 31-year-old Martin Hellman was a hard-nosed graduate of the Bronx High School of Science and an Assistant Professor at Stanford. Diffie had traveled the length of the country in search of collaborators on the mathematics of secret communication—not an easy field to enter, since most serious work in this area was being done behind the firmly locked doors of the National Security Agency. Ralph Merkle, a 24-year-old computer science graduate student at Berkeley, was exploring a new approach to secure communication.

In the most important single discovery in the entire history of cryptography, Diffie and Hellman found a practical realization of Merkle's ideas, which they presented in a paper entitled *New Directions in Cryptography.* This is what the paper described:[25]

> *A way for Alice and Bob to agree on a secret key by using messages between them that are not secret at all.*

In other words, as long as Alice and Bob can communicate with each other, they can establish a secret key. It does not matter if they are at opposite ends of the earth, have never met face to face, and have agreed on nothing in advance. It also does not matter if Eve or the entire world can hear everything they say to each other. Alice and Bob can come to a consensus on a secret key, known only to the two of them. There is no way for Eve to use what she overhears to figure out what that secret key is.

The impact of this discovery cannot be overstated. Since the dawn of writing, the art of secret communication had been a state monopoly. Governments had a monopoly on encryption in part because governments had the largest interests in secrecy and in part because the smartest scientists worked for governments. But there was another reason why governments had done all the serious cryptography. Only governments had the military power to protect and distribute the keys on which secret communication depended.

By contrast, after the astonishing discovery of Diffie, Hellman, and Merkle, everyone could use cryptography, because the keys could be worked out over insecure, public communication channels. You just had to know how; you did not need an army to protect the keys in transit. And, true to Kerckhoff's Principle, the know-how itself was not secret. You could read it in math journals, or just download the software from the Internet.

The method of Diffie, Hellman, and Merkle, which they called *public-key cryptography,* laid the foundation for electronic commerce. Suppose, for example, that Bob is a book buyer and Alice is Amazon. Bob wants to send Alice his credit card number, and wants to encrypt it using a secret key. In this scenario there is no possibility of Bob and Alice meeting—what would it even mean to go physically to Amazon to arrange for a key? The encryption has to be worked out on the spot, or rather, on the two separate spots separated by the Internet. Diffie-Hellman-Merkle, and a suite of related methods that followed, made secure Internet transactions possible. If you have ever used a password to log into a bank account, you have used public-key cryptography without realizing it. Your computer and the bank's played the roles of Bob and Alice.

It is wildly counterintuitive that Alice and Bob could agree on a secret key over a public communication channel. It was not so much that the scientific community had tried and failed to do what Diffie, Hellman, and Merkle did. It never occurred to anyone to try, because it seemed so obvious that for Bob to encrypt his message, he somehow had to give Alice the key. Not true.

9.8 Alice and Bob work out a secret key, in public

9.8.1 One-way computations

The possibility of public-key encryption derives from the fact that some things are much easier to compute than they are to "uncompute."[26] The simplest example of this idea is multiplication of integers. It is not hard to multiply two large numbers. The product of two ten-digit numbers, say $28487532223 \times 72342452989$, is around 20 digits long (in this case, 2060857961121397335 47). Even without use of a calculator, grinding out the product does not take that long—there are only

100 digit-by-digit products to compute, plus about 20 additions. But going in the other direction seems much harder—starting with the product and figuring out two numbers to multiply to give that result. If you have to try all possible ways of breaking a number into two parts and multiply all the possible combinations, it will take a very long time to be sure you have tried them all. The number of combinations you would have to try is on the order of the size of the number, or what is the same thing, exponential in the number of bits in the number—something like 2^{1000} combinations, if you want to try to factor a 1000-bit number.

A computation that is easy to do but hard to undo is called a *one-way computation.* The one-way computation Diffie and Hellman used was not multiplication and factoring, but rather *modular exponentiation* and *discrete logarithms.*

9.8.2 Modular arithmetic and computing powers quickly

To understand the discrete logarithm, we first need to get the idea of modular arithmetic. Suppose p is some fixed integer—say 24 or 7. To work with numbers modulo p is to care only about the remainder when the numbers are divided by p. For example, if $p = 24$, then $81 \bmod p = 9$, since 24 goes into 81 three times, with a remainder of 9.

When adding or multiplying numbers, if all you want is the result modulo p, you can reduce the numbers modulo p before adding or multiplying them. For example, to compute $701 \times 706 \bmod 7$, you could calculate $701 \times 706 = 494,906$, and then divide that by 7 and keep the remainder. But it is much simpler to notice that $701 = 1 \bmod 7$, $706 = 6 \bmod 7$, and $1 \times 6 = 6 \bmod 7$. That is, when adding and multiplying modulo p, you can reduce modulo p any time it is convenient to do so—throwing away everything but the remainder.

This idea can make it easy to compute powers: What is $5^{100} \bmod 24$? Calculating 5^{100} (a 70-digit number), and then dividing it by 24 to find the remainder, does not sound so easy. But we don't need to do that, since $5^2 = 25 = 1 \bmod 24$, so $5^{100} = (5^2)^{50} = 1^{50} = 1 \bmod 24$.

When computing powers modulo p, this method keeps the numbers small and also reduces the number of multiplications to be done. As an example, let's compute $12^{35} \bmod 23$.

$$\begin{aligned}
12^2 &= 144 &&= 6 \bmod 23 \\
12^4 &= (12^2)^2 &&= 6^2 = 13 \bmod 23 \\
12^8 &= (12^4)^2 &&= 13^2 = 8 \bmod 23 \\
12^{16} &= (12^8)^2 &&= 8^2 = 18 \bmod 23 \\
12^{32} &= (12^{16})^2 &&= 18^2 = 2 \bmod 23 \\
12^{35} &= 12^{32} \times 12^3 &&= 2 \times 12^3 = 3 \bmod 23
\end{aligned}$$

We kept the numbers small by reducing the product modulo 23 at each step and by substituting a smaller quantity for a larger if they are equal modulo 23. (For example, in the second line we substituted 6 for 12^2, since they are equal modulo 23 according to the first line.) Instead of doing 34 multiplications, we did only 6, because we kept squaring the result—thus doubling the exponent—at each step (except the last). This method for computing powers quickly is called *repeated squaring.*

In general, computing $g^n \bmod p$ by repeated squaring requires at most $2 \lg n$ multiplications, and the biggest numbers involved are smaller than p^2. The bottom line: powers modulo p can be computed quickly (especially with a computer).

9.8.3 Discrete Logarithms

Now here is the crucial point. *The reverse computation to modular exponentiation is hard,* at least as far as anyone knows. That is, suppose p is a fixed number known to everyone, and suppose also that g is publicly known. Now someone gives us a number q less than p, and wants to know what power of g would yield the result $q \bmod p$. For example, what value of n has the property that $54,321^n = 18,789 \bmod 70,707$? You could try plugging in $n = 1, 2, 3, 4 \ldots$, each time reducing the result mod $70,707$. But the answers bounce all over, with no apparent pattern: 54,321, 26,517, 57,660, 40,881, ... Even using repeated squaring, it would take you a long time to discover that $n = 43,210$ works to produce the desired answer 18,789.

That is, we want to solve the equation $g^n = q \bmod p$ for n, given values for g, q, and p. *No one knows how to do this, in general, in a way that is much faster than trying all possible values of n.*

An n that satisfies $g^n = q \bmod p$ is called a *discrete logarithm of q modulo p.* If the "mod p" were not there and n did not have to be an integer, this equation would be easy to solve: the answer would be $n = \log_g q$, and anyone with a pocket calculator could calculate it in a few keystrokes. But doing the arithmetic modulo p changes everything. As far as anyone knows, there is no way much better than exhaustive search to compute discrete logarithms.

If the numbers involved are, say, 1000 bits in length, it is easy to compute powers modulo p (requiring fewer than 2000 multiplications), but searching through some 2^{1000} possibilities for the discrete logarithm would be impossibly slow. In other words, modular powers are a one-way computation. They are easy to compute, but the reverse computation (discrete logarithm) is hard.

9.8.4 The key agreement protocol

Diffie and Hellman's insight was to use modular exponentiation as the basis for what is now called the *Diffie-Hellman-Merkle key agreement protocol*, which Alice and Bob can follow to obtain a shared key. Here's how it works:

Alice and Bob both start off knowing a particular number p, modulo which all subsequent calculations will take place. This number, which has perhaps 1000 bits, is some industry standard; the whole world knows what it is. (Realistically, they may not personally know the number; it is part of standard encryption software running on their computers.) Likewise, Alice and Bob know the value of an industry-standard base number g, also publicly known. Now they proceed as follows:

1. Alice and Bob each choose a random number less than p. Alice's number we'll call a and Bob's number we'll call b. We'll refer to a and b as Alice and Bob's *secret keys.* Alice and Bob

keep their secret keys secret. *No one except Alice knows the value of a, and no one except Bob knows the value of b.*

2. Alice and Bob raise g to the powers a and b respectively, modulo p. (Not too hard to do, even with 1000 bit numbers, using repeated squaring and modular arithmetic.) The results are called their *public keys* A and B:

$$A = g^a \bmod p, B = g^b \bmod p$$

3. Alice sends Bob the value of A and Bob sends Alice the value of B. It doesn't matter if Eve overhears these communications; A and B are not secret numbers.

4. When she has received Bob's public key B, Alice computes $B^a \bmod p$, using her secret key a as well as Bob's public key B. Likewise When Bob receives A from Alice, he computes $A^b \bmod p$, using his secret key and Alice's public key.

5. Now here's the crucial point: even though Alice and Bob have done different computations, they have ended up with **the same value**, since

$$B^a = (g^b)^a = (g^a)^b = A^b \bmod p.$$

This shared value, call it K, is the key Alice and Bob will use for encrypting and decrypting their subsequent messages, using whatever standard method of encryption they choose.

Suppose Eve has been listening to Alice and Bob's communications. Can she do anything with the all information she has? She has overheard A and B, and she knows p and g because they are industry standards. She knows all the algorithms and protocols that Alice and Bob are using; Eve has read Diffie and Hellman's paper too! But to compute the key K, Eve would have to know either a or b. Finding either a or b would require solving a discrete logarithm problem: finding the value of a for which $g^a = A \bmod p$, or similarly for b. On numbers of a thousand bits, no one knows how to do that without searching through impossibly many trial values. In contrast, Alice and Bob only have to modular powers, fast computations if hey use repeated squaring.

For 1000-bit numbers, Alice and Bob can carry out their required computations with personal computers. But even the most powerful computers aren't remotely fast enough to let Eve break the system by computing discrete logarithms, at least not by any method known.[27]

Exploiting this difference in computational effort was Diffie, Hellman, and Merkle's breakthrough. They showed how to create shared secret keys over insecure channels.

9.8.5 Public keys for private messages

Public key cryptography was designed to solve the ancient problem of secret communication between two cooperating parties in the presence of adversaries and eavesdroppers. But the remarkable details of the solution yield much more. Once we realize that it doesn't matter if public keys are known to an adversary, we might as well truly make them public—by publishing them in a directory, for example. But what would be the use of doing that?

Suppose Alice wants anyone in the world to be able to send her secret messages. She doesn't want to work out a new key with everyone who wants to reach her; she just wants to sit back and receive secrets. Alice picks a secret key a and computes the corresponding public key A exactly as before. But now she publishes A in a public directory, without waiting for anyone in particular to get in touch with her.

If Bob decides to send Alice a secret, he gets Alice's public key from the directory. Next he picks his own secret key b and computes B as before. He also uses Alice's public key A from the directory to compute an encryption key K just as with the key-agreement protocol: $K = A^b \bmod p$.

Using K as a key, Bob encrypts his message to Alice. He sends Alice the ciphertext of his message. *At the same time, he sends B in cleartext—he just includes his own public key in the same communication with his encrypted message!*

When Alice receives Bob's encrypted message, she takes the B that came with message, together with her secret key a, just as in the key agreement protocol, computes the same K: $K = B^a \bmod p$.

Alice now uses K as the key for decrypting the message. Eve, or others who see Bob's message, can't decrypt it, because they don't know his secret key. Knowing B doesn't help, unless Eve somehow can compute discrete logarithms. The whole world can send Alice their secrets, and no one who intercepts the messages in transit can read them!

9.8.6 Digital signatures

But even that is not all! A further breakthrough of public-key cryptography is authentication by means of *digital signatures.*

Suppose Alice wants to send a message by electronic mail or create a public announcement. How can people who see the message be sure that it really comes from Alice—that it's not a forgery? The message has to be marked in a way that can be easily verified but not easily forged. Such a mark is called a digital signature.

Like public key encryption, digital signature protocols use public keys and secret keys. There are two related computations, one for processing a message to create the signature, and one for verifying signatures. Alice uses her secret key to create the digital signature. Anyone can then use Alice's public key to verify the signature. Everyone can know the public key and thus verify the signature, but only Alice knows the secret key and could have produced the signature. This is the reverse of public-key encryption, where anyone can encrypt the message, but only the person with the secret can decrypt the message.

A digital signature scheme requires a computational method that makes signing if easy if you have the secret key and verifying is easy if you have the public key, but makes it computationally infeasible to produce a verifiable signature if you don't know the secret key. Given a good signature scheme, a verified signature attests to:

- *Message authenticity:* The message was signed by the person who knows the secret key.
- *Message integrity:* The message that was received is the message that was signed (no one tampered with it *en route*).

So unlike an ordinary signature, a digital signature depends on the message, as well as who is doing the signing—that's why it can be used for integrity as well as for authenticity.

9.8.7 RSA

Diffie and Hellman introduced the concept of digital signatures in their 1976 paper, and they suggested some ways that signatures might be implemented, but they did not present a concrete method. The problem of devising a practical digital signature scheme was left as a challenge to the computer science community.

The challenge was met in 1977 by Ron Rivest, Adi Shamir, and Len Adleman of the MIT Laboratory for Computer Science. The RSA (Rivest-Shamir-Adleman) algorithm is both a practical digital signature scheme, and also a confidential messaging scheme.

With RSA, each person generates a pair of keys—a public key and a secret key. We'll again call Alice's public key A and her secret key a. The public and private keys are inverses: If you transform a value with a, then transforming the result with A recovers the original value. If you transform something with A, then transforming the result with a recovers the original value.

Here's how RSA key pairs are used. People publish their public keys and keep their secret keys to themselves. If Bob wants to send Alice a message, he picks a standard encryption algorithm and a key K, and transforms K using Alice's public key A. Alice transforms the result using her secret key a to recover K. As with all public-key encryption, only Alice knows her secret key, so only Alice can recover K and decrypt the message.[28]

To produce a digital signature, Alice transforms the message using a and uses the result as the signature to be sent along with the message. Anyone can then check the signature by transforming it with A to verify that this matches the original message. Since only Alice knows her secret key, only Alice could have produced something that when transformed with her public key, will reproduce the original message.[29]

What makes RSA secure is that it is infeasible to compute the secret key corresponding to a public key. RSA depends not on discrete logarithms, but on the fact that it seems to take exponentially longer to factor an n-digit number than to multiply two $n/2$-digit numbers. Multiplying numbers takes time proportional to the number of digits, but factoring seems to require time proportional to the number itself, which is exponentially larger than its length in digits. But no proof has been discovered that factoring is as hard as it seems to be. Because a great deal of Internet security now uses RSA, which depends on the difficulty of factoring, enormous effort has gone into the search for fast ways to factor numbers—a subject that used to be merely a mathematical recreation.

9.8.8 Certificates and certification authorities

There's a problem with the public-key methods we've described so far. How can Bob know that the "Alice" he's communicating with really is Alice? Anyone could be at the other end of the key-agreement communication pretending to be Alice. Or, for secure messaging, after Alice places her public key in the directory, Eve might tamper with the directory, substituting her own key in place of Alice's. Then anyone who tries to use the key to create secret messages intended for Alice, will actually be creating messages that Eve, not Alice, can read.

Digital signatures can help. Alice goes—physically, perhaps—to a trusted authority, to whom she presents her public key together with proof of her identity. The authority digitally signs Alice's key—producing a signed key called a *certificate*. Now, instead of just presenting her key when she wants to communicate, Alice presents the certificate. Anyone who wants to use the key to communicate with Alice first checks the authority's signature to see that the key is legitimate.[30]

People check a certificate by checking the trusted authority's signature. How do they know that the signature on the certificate really is the trusted authority's signature, and not some fraud that Eve set up for the purpose of issuing fake certificates? The answer is that the authority's signature is itself guaranteed by another certificate, signed by another authority, and so on, until we reach an authority whose certificate is well-known. In this way, Alice's public key is vouched for, not only by a certificate and a single signature, but by a chain of certificates, each one with a signature guaranteed by the next certificate.

Organizations that issue certificates are called *certification authorities*. Certification authorities can be set up for limited use (for example, a corporation might serve as a certification authority that issues certificates for use on its corporate network). There are also companies that make a business of selling certificates for public use. Notice that the trust you should put in a certificate depends on two things: (1) your assessment of the reliability of the signature on the certificate and *also* (2) your assessment of the certification authority's diligence in signing things.[31]

9.9 Cryptography for everyone

In real life none of us is aware that we are performing modular arithmetic or computing exponentials while browsing the Web. But every time we order a book from Amazon or check our bank or credit card balance or pay for a purchase using PayPal, that is exactly what happens. The tell-tale sign that an encrypted web transaction is taking place is that the URL of the web site begins with "`https`" (the "`s`" is for "secure") instead of "`http`." The consumer's computer and the computer of the store or the bank negotiate the encryption, using Diffie-Hellman-Merkle or RSA, unbeknownst to the human beings involved in the transaction. The store attests to its identity by presenting a certificate signed by a certification authority that the consumer's computer has been preconfigured to recognize. New keys are generated for each new transaction. Keys are cheap. Secret messages are everywhere on the Internet. We are all cryptographers now.

When it was discovered, public-key encryption was treated as a mathematical curiosity, its break-

through value largely unappreciated. Len Adleman, one of the inventors of RSA, thought that the RSA paper would be "the least interesting paper I would ever be on."[32] As late as 1977, even the National Security Agency was not overly concerned about the spread of these methods. They simply did not appreciate how the personal computer revolution would soon enable anyone with a home PC to exchange encrypted messages that even NSA could not decipher.[33]

But as the 1980s progressed and Internet use increased, intelligence agencies began to worry about widespread cryptography. Law enforcement feared that encrypted communications could put at end to government wiretaps—a powerful law-enforcement tool. At the same time, industry was beginning to appreciate that their customers would want private communication, especially an era of electronic commerce. In the late 1980s and early 1990s, the Bush and the Clinton administrations floated proposals aimed at controlling the spread of cryptographic systems.

In 1994, the Clinton administration unveiled a plan for an "Escrowed Encryption Standard" that would be used on telephones that provided encrypted communications. The technology, dubbed "Clipper," was an encryption chip developed by the NSA that included a back door, an extra key held by the government, that would let law enforcement and intelligence agencies decrypt the phone communications. According to the proposal, the government would purchase only Clipper phones for secure communication. Anyone wanting to do business with the government over a secure telephone would also have to use a Clipper phone. Industry reception was cold, and the plan was dropped. But in a sequence of modified proposals beginning in 1995, the White House attempted to convince industry to create encryption products that had similar back doors. The carrot here, and the stick, was export control law. Under US law, cryptographic products, being munitions, could not be exported without a license. Violating export controls was a criminal act. The administration proposed that encryption software would receive export licenses only if it contained back doors.

The ensuing, often heated, negotiations, sometimes referred to as the "Crypto wars," played out over the remainder of the 1990s. Law enforcement and national security argued the need for encryption controls. On the other side of the debate were the technology companies, who did not want government regulation, and civil liberties groups, who warned against the potential for growing communication surveillance. In essence, policymakers could not come to grips with the transformation of a major military technology to an everyday personal tool.

To appreciate the tensions that accompanied this transition, consider the career of Phil Zimmermann, a journeyman programmer and civil libertarian who had been interested in cryptography since his youth. He had read a *Scientific American* column about RSA encryption in 1977, but did not have access to the kinds of computers that would be needed to implement arithmetic on huge integers as the RSA algorithms demanded. As the 1980s progressed, and it began to be feasible implement RSA on home computers, Zimmermann set about to produce encryption software for the people, to counter the threat of government surveillance.

As Zimmermann later testified before Congress, computers had tipped the balance against the privacy of citizens. Surveillance used to be costly. Wiretapping and steaming open envelopes required as many agents as surveillance targets. "This is analogous to catching fish with a hook and a line, one fish at a time," Zimmermann said. "Fortunately for freedom and democracy, this kind of labor-intensive monitoring is not practical on a large scale." But email could be scanned "easily, routinely, automatically, and undetectable on a grand scale. This is analogous to driftnet

fishing—making a quantitative and qualitative Orwellian difference to the health of democracy."[34] Cryptography was the answer. If governments were to have unlimited surveillance powers over electronic communications, people everywhere needed easy-to-use, cheap, uncrackable cryptography so they could communicate without governments being able to understand them.

Zimmermann faced obstacles which would have stopped less zealous souls. RSA was a patented invention. MIT had licensed it exclusively to the RSA Data Security Company, which produced commercial encryption software for corporations, and RSA Data Security had no interest in granting Zimmermann the license he would need to distribute his RSA code freely, as he wished to do.

And there was government policy, which was, of course, exactly the problem to which Zimmermann felt his encryption software was the solution. On January 24, 1991, Senator Joseph Biden, a cosponsor of antiterrorist legislation Senate Bill 266, inserted some new language into the bill to require that communication equipment manufacturers and service providers ensure that "communications systems permit the government to obtain the plaintext contents of voice, data, and other communications." This language received a furious reaction from civil liberties groups and wound up not surviving, but Zimmermann decided to take matters into his own hands.

By June of 1991, Zimmermann had completed a working version of his software, now dubbed PGP for "Pretty Good Privacy," after Ralph's mythical Pretty Good Groceries that sponsored Garrison Keillor's *Prairie Home Companion*. The software mysteriously appeared on several US computers, available for anyone in the world to download. Soon copies were everywhere, not just in the US but all over the world. In Zimmermann's own words: "This technology belongs to everybody."[35] The genie was out of the bottle and was not going back in.

Zimmermann paid a price for his libertarian gesture. First, RSA Data Security was confident that this technology belonged to *it*, not to "everybody." The company was enraged that its patented technology was being given away. Second, the government was furious. It instituted a criminal investigation for violation of the export control laws, though it was unclear what laws, if any, Zimmermann had violated. Eventually MIT brokered an agreement that let Zimmermann use the RSA patent, and devised a way to put PGP on the Internet for use in the US, and in conformance with export controls.

By 2000, the progress of electronic commerce had overtaken the key escrow debate. The government ended its criminal investigation without an indictment. Zimmermann built a business around PGP while still allowing free downloads for individuals. His web site contains testimonials from human rights groups in Eastern Europe and Guatemala attesting to the liberating force of secret communication among parties resisting oppressive regimes. Zimmermann had won. Sort of.

9.10 Cryptography unsettled

Today, every credit card transaction over the World Wide Web is encrypted. There is widespread concern about information security, identity theft, and degradation of personal privacy. PGP and other good quality email encryption programs are widely available—many for free.

But very little email is encrypted today. Human rights groups use encrypted email. People with something to hide probably encrypt their email. But most of us don't bother. Why not?

Few people realize how easily email can be captured as the packets flow through the Internet. The password requests we know are needed to get our email out of the mail server may provide the illusion of security, but they do nothing to protect the bits from being sniffed as they float through fibers, wires, and the air. The world's biggest eavesdropping enterprise is little known: the international ECHELON system, which automatically monitors data communications to and from communications satellites that relay Internet traffic. If your email uses words that turn up in ECHELON's dictionary, your email may get a closer look.[36] ECHELON is Zimmermann's driftnet. And most people, unaware of the implications of such vast data harvests, figure that no one is watching them, since they have done nothing worth looking for. They are wrong.

Encrypted email is rare also because it is not built into the Internet infrastructure as encrypted web browsing is. You have to use nonstandard software, and the people you communicate with have to use some compatible software. You have to think about what you are doing. In commercial settings, companies may not want to make encryption easy for office workers. Companies have an interest in watching out for criminal activities, and in regulated industries, such as finance, the government may require them to monitor their employees' activities.

It is not just email and credit card numbers that might be encrypted: Instant Messaging and Internet-based telephone conversations (Voice over IP) are just more packets flowing through the Internet. Some Internet phone software (such as Skype) encrypts conversations, and there are several other products under development, including one led by Zimmermann himself, to create easy-to-use encryption software for Internet telephone conversations.

Overall, the public seems not very concerned about the privacy of communication issues that permeated the crypto wars of a decade ago. In a very real sense, the dystopian predictions of both sides of that debate are being realized: On the one hand, encryption technology is readily available around the world, and people with something to hide can hide the contents of their messages, just as law enforcement feared—Al Qaeda may well use PGP, for example.[37] At the same time, the spread of the Internet has been accompanied by an increase in surveillance, just as the opponents of encryption regulation feared. Even universities are required to fashion their computer networks to facilitate government surveillance of email.[38] Vast amounts of web-based email are now held not by the senders and recipients alone, but also by service providers such as Google, whose interests in protecting its contents may not be as high as the interests of the communicating parties.

So although outright prohibitions on encryption are now impossible, the social and systems aspects of encryption remain in an unstable equilibrium.[39] Will some information privacy catastrophe spark a massive re-education of the Internet-using public, or massive regulatory changes to corporate practice? Will Microsoft or some other software vendor make encryption so easy that it becomes the default option and most communications wind up encrypted?

And if that happens, will the benefits to personal privacy and free expression outweigh the costs to law enforcement and national intelligence, whose capacity to eavesdrop and wiretap will be at an end?

Notes

[1]Congressional Record, September 13, 2001, page S9357.

[2]Quoted by John Schwartz, Disputes on electronic messages: Encryption takes on new urgency, New York Times, September 25, 2001.

[3]Schwartz, Disputes, NYT.

[4]Schwartz, Disputes, NYT.

[5]Declan McCullagh, Senator backs off backdoors, Wired News, October 17, 2001.

[6]"Whoever, after January 31, 2000, sells in interstate or foreign commerce any encryption product that does not include features or functions permitting duly authorized persons immediate access to plaintext or immediate decryption capabilities shall be imprisoned for not more than 5 years, fined under this title, or both." 105th Congress, H.R. 695. House Report 104-108 part 4 "Security and Freedom Through Encryption (SAFE) Act of 1997," Section 2803. At http://thomas.loc.gov/cgi-bin/cpquery/5?&sid=cp105njDpq&hd_count=5&xform_type=3&db_id=cp105&r_n=hr108p4.105&item=5& (visited, July 22, 2005).

[7]National Research Council, *Cryptography's Role in Securing the Information Society,* Kenneth W. Dam and Herbert S. Lin, Editors, National Academy Press, 1996.

[8]Testimony of FBI Director Louis Freeh, before the Senate Judiciary Committee Hearing on Encryption, United States Senate, Washington, D. C., July 9, 1997.

[9]Source: For growth in online banking, Financial Services Fact Book, http://www.financialservicesfacts.org/financial2/technology/ec visited 12/1/2006; for figures on retail sales, US Census Bureau News, August 17, 2006, Quarterly Retail e-Commerce Sales, 2nd Quarter 2006.

[10]Statement by Ron Rivest at MIT Press Forum on Encryption, MIT, April 7, 1998.

[11]Suetonius, the Lives of the Caesars, The Life of Julius Caesar, Chapter 56, from the Loeb Classical Library, 1913.

[12]Actually, the Romans didn't use J, U, or W, so Caesar had only 22 shifts available.

[13]There is a substitution cipher even in the Old Testament. The Lord trumpets at one point, "How Sheshach will be captured, the boast of the whole earth seized! What a horror Babylon will be among the nations!" Jeremiah 51:41 (New International Version). There is no place called Sheshach. It's just the word for Babylon, with the last letter of the Hebrew alphabet being substituted for the first, the next to last letter of the alphabet for the second, and so on.

[14]Also known as a *Vernam cipher,* after its World War I-era inventor, AT&T telegraph engineer Gilbert Vernam.

[15]http://www.nsa.gov/publications/publi00039.cfm

[16]A new cipher code, Scientific American Supplement, v.58 (Jan. 27, 1917), p. 61. From the Proceedings of the Engineers Club of Philadelphia. [Baker Business Library, Historical Collection, By appointment only]

[17]as coined by Charlie Kaufman, Radia Perlman, and Mike Speciner in Network Security: Private Communication in a Public World, Prentice-Hall, 1995, p. 40.

[18]See "NIST brief comment on recent cryptanalytic attacks on SHA 1 and NIST plan," at http://www.csrc.nist.gov/pki/HashWorksho (visited July 31, 2005).

[19]Nikita Borisov, Ian Goldberg, and David Wagner, Intercepting Mobile Communications: The Insecurity of 802.11, Proceedings of the Seventh Annual International Conference on Mobile Computing And Networking, July 16–21, 2001.

[20]"RFID crack raises spector [sic] of weak encryption: Steal a car — and the gas needed to get away," Computerworld, March 17, 2005, available at http://www.computerworld.com/mobiletopics/mobile/technology/story/0,10801,100459,00.html (visited, July 31, 2005).

[21]Auguste Kerckhoffs, "La cryptographie militaire," Journal des sciences militaires, vol. IX, pp. 5–38, Jan. 1883, pp. 161–191, Feb. 1883. Available on the Web at http://www.petitcolas.net/fabien/kerckhoffs/crypto_militaire_1.pdf

[22]Kerckhoffs, page 12, 14.

[23]Jeffrey A. Bloom, Ingemar J. Cox, Ton Kalker, Jean-Paul M. G. Linnartz, Matthew L. Miller, and C. Brendan S. Traw, "Copy Protection for DVD Video," Proceedings of the IEEE, vol. 87, no. 7, July 1999, pp. 1267-1276.

[24]Federal Information Processing Standards Publication 197, Advanced Encryption Standard, http://csrc.nist.gov/publications/fips/fi 197.pdf

[25]Whitfield Diffie and Martin Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, November 1976. An important piece of the puzzle was contributed by Berkely graduate student Ralph Merkle, and today Diffie, Hellman, and Merkle are typically given joint credit for the innovation. Despite the credit to Diffie, Hellman, and Merkle, it was finally revealed in 1997 that the same techniques had been developed within the British secret Government Communication Headquarters (GCHQ) two years before Diffie and Hellman's work, by James Ellis, Cliiford Cocks, and Malcolm Williamson. See James Ellis, "The History of Non-secret Encryption," available at http://www.cesg.gov.uk/site/publications/media/ellis.pdf (visited August, 2005).

[26]This way of phrasing the idea is due to Ron Rivest.

[27]And if anyone knows a fast method, they aren't talking.

[28]You might ask why you need DES encryption and K at all. Why not just encrypt the message itself using the public key, and decrypt it using the secret key. This would work in principle, but would be less computationally efficient, because the RSA transformations are slower than algorithms like DES for long messages.

[29]In real systems, Alice doesn't actually encrypt the message itself, but rather an abbreviated version of it, called a *message digest*, which is produced by a method called a message digest function. The reason for a message digest is that it is much shorter than the message itself (and thus more efficient to encrypt to produce a signature). With a good message digest function it should be computationally infeasible for an attacker to create a different message that has the same digest. As mentioned above, weaknesses in the popular message digest function MD5 (created by Rivest) were discovered in summer 2004, and this is making the cryptographic community wary about the use of message digests.

[30]This idea, and the use of the word "certificate," were introduced by Loren Kohnfelder in his 1978 MIT bachelor's thesis. Kohnfelder, Loren M., "Towards a Practical Public-key Cryptosystem," MIT S.B. Thesis, May. 1978 (unpublished).

[31]VeriSign, which is currently the major commercial certification authority, issues three classes of personal certificates. Class 1 is for assuring that a browser is associated with a particular email address and makes no claims about anyone's real identity. Class 2 provides a modest level of identity checking: organizations issuing them should require an application with information that can be checked against employee records or credit records. Class 3 certificates require applying in person for verification of identity.

[32]As quoted in "The Science of Secrecy: The birth of the RSA cipher," http://www.channel4.com/science/microsites/S/secrecy/page5 (visited Oct. 10, 2005).

[33]Conversation between Bobby Ray Inman and Hal Abelson, February 1995. Shortly after RSA appeared, the NSA asked MIT not to publish the paper, but MIT refused, citing academic freedom.

[34]Testimony of Philip Zimmermann to Subcommittee for Economic Policy, Trade, and the Environment, US House of Representatives, October 12, 1993.

[35]Zimmermann, Congressional testimony, October 12, 1993.

[36]See, for example, http://www.fas.org/irp/program/process/echelon.htm.

[37] http://www.strategypage.com/dls/articles/2004850.asp

[38]Sam Dillon and Stephen Labaton, Colleges oppose call to upgrade online systems, *New York Times,* October 23, 2003.

[39]Add a note here about prohibitions on encryption in other countries.