

# Cryptography Demystified

By Harry E. Smith, CISSP

[harry\\_e\\_smith@timberlinetechnologies.com](mailto:harry_e_smith@timberlinetechnologies.com)

## Introduction

The well-known science fiction author Arthur C. Clarke once noted that, "Any sufficiently advanced technology is indistinguishable from magic." This certainly applies to the field of cryptography. We information security professionals have increasingly come to depend on cryptographic techniques to protect information assets because we have become frustrated with the difficulty doing so with the standard authentication and access control tools available to us.

But is our faith in the "magic" of cryptography justified? Can an over-dependence on a single technology as a universal information security solution lead to our building a "Maginot line in cyberspace?" I believe that this is what Bruce Schneier is suggesting in the introduction to *Secrets and Lies*: "Readers believed that cryptography was a kind of magic security dust that they could sprinkle over their software and make it secure. They could invoke magic spells like '128-bit key' and 'public-key infrastructure.' A colleague once told me that the world was full of bad security systems designed by people who read *Applied Cryptography*."

Even those who have mastered the intricacies of other security-related disciplines are sometimes reluctant to dig too deeply into the details of cryptography because of the belief that only PhD-level mathematicians and other members of the "cryptography priesthood" can understand them. Unfortunately, this disinclination to explore the technology has resulted in many misapplications of it. Storing keys on the same systems as the protected data (i.e. "hiding the key under the doormat") and recycling "one-time" pads are but two examples of this.

Understanding a few core ideas will help the reader to overcome this barrier and will eventually enable the reader to make much more reliable purchasing and implementation decisions. This article will explore three of these basic concepts:

1. The most complicated cryptographic techniques consist of repetitive applications of three elementary operations.
2. Public-key approaches allow cryptography to be extended beyond secrecy applications.
3. Key-length is a fundamental determinant of encryption strength.

This article will explain these basic ideas and will then suggest some references for further exploration.

## Basic Terminology

There are essentially two ways of keeping a secret:

1. You can disguise the fact that the secret information exists—this is called *steganography*.

2. You can alter the representation of the secret information so that only certain people will know what it means – this is called *cryptography*.

If you are a fan of the "cloak and dagger" films of the 30s and 40s, you have seen many examples of steganography. The microdot photograph of the secret formula that is "hidden in plain sight" in a pack of cigarettes is one example. So is the typed letter in which certain characters are ever so slightly raised. Still another form of steganography is the *acrostic*—a poem in which the first letters of each line are used to communicate a secret message. (The last poem in Lewis Carroll's *Through the Looking Glass*, for example, is an acrostic in which the first letter of each stanza spells out the name of the young woman who inspired the "Alice" books—Alice Pleasance Liddell.)

Cryptography, on the other hand, does not hide the fact that a message exists (although there is no reason why cryptography could not be used in combination with steganography). The "cryptogram" puzzle that is found in the features sections of most newspapers is there for all to see; but its secret content is available only to those who are willing to spend the time and energy required to perform the *cryptanalysis* of it. Cryptography (the art of disguising the meaning of a message) and cryptanalysis (the art of coaxing the secret out of an encrypted message) are complementary components of the field of *cryptology*.

There are two basic forms of cryptography. One may disguise the meaning of a message by using either a code or a *cipher*. Codes are generally schemes for representing whole phrases or concepts by symbols. Examples of codes would include "one if by land, two if by sea"; a baseball catcher's hand signals; Navajo "code talk"; and compression algorithms. Morse code is actually a cipher. The "Bible Code" is actually steganography (depending, of course, on your theological viewpoint). Ciphers, on the other hand, translate low-level symbols such as the letters of the English language, or individual bits, into an unrecognizable form. Examples of ciphers are the "Captain Midnight" decoder ring (I know that most of you are not old enough to remember these, but I still have mine); the Advanced Encryption Standard (AES); and Pig Latin.

A *cryptosystem* includes the techniques or *algorithms* used to mask the meaning of a communication and it also includes the *cryptographic protocols*—the form and sequence of messages—used by the communicating parties to make the whole scheme work. It is frequently easier for a cryptanalyst to find a weakness in the protocol than it is to find a weakness in the algorithm. Which brings us to one of the most valued and time-honored principles, known in cryptography circles as Kerckhoffs's Law: "The trustworthiness of a cryptosystem must depend only on the secrecy of the decryption *key*." This is to say: Only published algorithms are subjected to such rigorous testing that we can have confidence that the algorithm does not contain a fundamental flaw. This is a major point of contention among those who oppose the US government's "escrowed encryption standard." The Skipjack algorithm, implemented on the Clipper chip, is secret. An

algorithm for which security depends on keeping its operational details secret is known as a *restricted algorithm*.

I won't have very much to say about codes in this article, just as I don't have a lot to say about steganography. Codes are of some historical significance but, compared with ciphers, they have one great defect: They can be used only to translate the phrases that can be stored in a "code book." If the British had decided to come neither by land nor by sea, but by hot air balloon, for example, there would have been no way to communicate this late-breaking intelligence to Paul Revere. Ciphers, however, can be used to conceal any conceivable message from all except those in possession of the secret key. Codes were useful when *deciphering* a message was a tedious, manual process, but the advent of computer technology has changed all of that.

Ciphers can be implemented in the form of programs or chips. Hardware implementations are faster, but software implementations are more flexible. Hardware implementations can process significantly more data in the same amount of time. The problem with hardware implementations is that both parties to the communication must have compatible "crypto cards" and must be connected by a direct link. Software allows for encryption at the higher layers of the OSI model (sometimes known as "end-to-end encryption"). The problem with software encryption is that paging and swapping operations leave the intermediate products of encryption operations (including keys) on the hard disk. This is, of course, a potential exposure.

## Elementary Cipher Operations

It may seem at first that there is no way to gain a basic understanding of cryptography without dealing with dizzying complexity. There is, however, one core idea that cuts through this complexity: Every cipher algorithm, no matter how intricate, is composed of only three elementary operations. (And one of these is rarely used!) The three basic operations that transform *plaintext* into *ciphertext* are:

1. Transposition—scrambling the order of characters (or bits) in a message.
2. Substitution—exchanging one character (or bit) for another.
3. Null injection—adding "noise" characters (or bits) to the cipher output.

A cipher algorithm is a general description of the way in which the elementary operations are strung together to accomplish the encryption task. The key is the specific data that determines the precise details of each operation. The Data Encryption Standard (DES), for example, employs 16 "rounds" in which each round consists of a set of transposition and substitution operations. The general description of the sequence of these operations is a description of the DES algorithm. The specific transpositions and substitutions that are performed, on the other hand, are dependent on the values contained in the 56-bit key. Knowledge of this key information is necessary to reverse the effect of the 16 rounds of elementary operations and to thus *decrypt* the message.

(As you have probably guessed already, null injection is rarely if ever used in modern cipher algorithms. The reason for this is that we almost never want to lengthen the message because of the performance implications.)

Believe it or not, the *monoalphabetic substitution cipher*, which is the cipher algorithm used to construct the daily newspaper cryptogram, was considered to be "unbreakable" for about a thousand years! You can create your own monoalphabetic substitution cipher by taking one each the 26 letters of the English alphabet from a Scrabble™ game, putting them in a box, giving the box a shake and then drawing them out one at a time. If

you line up the letters in this random arrangement under a set of letters in proper order, you have a "substitution alphabet," which you can use to encipher and decipher messages. (There are "26 factorial" ways in which this can be done – which is a pretty big number.)

Arab scholars came up with the idea of using "frequency analysis" in the 9th century to crack messages encrypted with monoalphabetic substitution ciphers. Frequency analysis works on the principle that some letters are used more frequently than others. In English, for example, the letter "e" is used most often; "t" is most often used as the first letter in a word; etc. The fact that this technique remained undiscovered for so long gives us a profound insight into the nature of human creativity and the limitations of human progress.

## Symmetric vs. Asymmetric Cryptography

A novel solution to the "key exchange" problem led to a fascinating development in the history of cryptography in 1976. The idea that two people who had never met might be able to establish a secure communications channel (such as must be done to transmit credit card numbers or other sensitive information via the Internet) had always been stymied by a "chicken and egg" problem. How can one person communicate the secret key to another before the secure channel has been established? And if you have a way to communicate a secret key, why not just use it to communicate the secret information?

Whitfield Diffie and Martin Hellman saw a way to "bootstrap" the process by using a curious property of discrete mathematics. When you do arithmetic using *modular* addition, subtraction, multiplication and division (i.e. where the results of arithmetic operations are represented as remainders after division by the *modulus* value) you encounter situations in which working a problem in one direction is extremely easy but working the problem in the other direction is essentially impossible. So, for example, it is easy to compute the modular product of two very large numbers, but no one has figured out how to factor a large number into its associated discrete multiplicands. Similarly, it is easy to raise a large number to a power in a modular fashion, but virtually impossible compute a discrete logarithm.

This fundamental breakthrough led to something much more significant: asymmetric (also known as "public-key") cipher algorithms. Symmetric-key ciphers employ a single, secret key for both encryption and decryption operations. Asymmetric-key cryptography employs a "public" key for encryption operations and uses a "private" key for decryption operations. Credit for the conceptual insight goes to Whitfield Diffie, but Ron Rivest, Adi Shamir and Leonard Adleman, for whom the "RSA" algorithm was named, developed the first working public-key algorithm. (Actually, three British scientists—James Ellis, Clifford Cocks and Malcolm Williamson—have a prior claim, but their work was classified and they could not publish their results.) There is no key distribution problem with asymmetric-key methods. Everyone generates a unique public/private key pair and publishes his or her public key. When Alice wants to send a secret message to Bob, she looks up his public key and uses it to encrypt the message. Only Bob can unravel the ciphertext because he has never shared his private key with anyone. When Bob wants to answer Alice's message he looks up her public key and uses it to encrypt his reply.

But how do you know that you have an accurate copy of the public key of someone with whom you wish to communicate or whose digital signature you want to check? This is where *public key certificates* enter the picture. A public key certificate is a document containing the name of a person and a copy of that person's public key; and which has been signed someone you trust. Public key certificates are also known as *digital certificates*. Some companies, such as Verisign and Entrust, have established a reputation as Certificate Authorities

(CAs). A certificate issued by a CA can be trusted based on the CA's reputation. The ISO X.509 standard establishes a common public key certificate layout. This makes interoperability between certificate issuing and validation tools, including those produced by different vendors, possible.

**Public key infrastructures (PKI)** are high-level designs for implementing public key technology. PKIs define a root CA and possibly several subordinate CAs. They also have a mechanism for deploying certificate revocation lists (CRLs) and some form of certificate expiration. The use of public key technology depends to no small extent on the level of confidence one has in the public keys that one possesses.

(PGP takes a different approach. PGP users establish "webs of trust" by having digital certificates that contain multiple signatures. The idea is to have your public key signed by as many trustworthy people as you can. When you receive a new public key that is signed by someone you trust, you trust the key. PGP also incorporates the concept of "levels of trust.")

## The Significance of Key-length

Anyone who has ever been involved in a cryptography discussion sooner or later gets to the topic of key length. Trying every possible key can break every cipher algorithm (except the "one-time pad"). This is known as an *exhaustive search of the key space* or a *brute-force attack*. Every additional bit of key length doubles the number of keys that have to be tried.

For a given cipher algorithm, the time required to try every key is given by

$$T = 2^n * t$$

where "t" is the time needed to try a single key and n is the key length in bits. Thus, an encryption scheme is "practically" unbreakable if the time needed to try every possible key is something comparable to the age of the universe (and there is no more efficient way to decrypt the message than to try every key).

The primary significance of key-length (or key complexity) is that the longer (or more complex) the key, the greater the number of possible keys that a potential attacker has to try. It is extremely important, however, that all keys in the key space should be equally likely. If the cryptanalyst can restrict the brute-force attack to a small subset of all possible keys (for example, words found in the dictionary), then the security of the cryptosystem falls apart. "Key-clustering," different keys producing the same output, is likewise undesirable.

How can we ever be sure that there is not a shortcut method of breaking an encryption algorithm so that a brute force attack is not necessary? This was the question that the National Institute of Standards and Technology (NIST) took up, and that led to the adoption of the Data Encryption Standard in 1977. DES uses a 56-bit key to encrypt 64-bit blocks of data. No one has ever found a way to crack the DES algorithm itself, but the Electronic Frontier Foundation built a \$250,000 brute force "DES Cracker" in 1998 that could perform an exhaustive search in an average of 3 days. Even though advances in processing speed have caught up with the 56-bit DES algorithm, DES still has a role to play. "Triple DES" (encrypting a message 3 times) has an effective key-length of 112 bits and most experts do not expect that advances in processor speed will make a brute-force approach to breaking triple DES feasible any time soon.

Before the EFF showed that (single) DES was not dependable, NIST had already begun the search for a successor algorithm. In 1997, NIST announced a competition for the DES follow-on—the Advanced Encryption Standard (AES). In October of 2000, NIST chose the Rijndael algorithm developed by Joan Daemen and Vincent Rijmen, of Belgium, from among the original 15 candi-

dates. Rijndael supports three key length choices: 128, 192 or 256 bits. There is a fairly convincing argument in Bruce Schneier's *Applied Cryptography* to the effect that there will probably never be a need for cryptographic keys that are longer than 256 bits, assuming that no alternatives to a brute force attack exist. (The property of an algorithm that there is no alternative to a brute-force attack is sometimes called *perfect algorithm strength*.)

## Where to Go from Here

At this point, hopefully, you have been encouraged to explore this fascinating subject a little further. What you should read next depends on your orientation and interests. In the *Suggestions for further Reading* you will find some excellent sources. Here are some additional ideas to guide your choices:

**Game Theory.** If you are into backgammon or chess, you might consider cryptography from the perspective of games and competition. For several millennia cryptographers and cryptanalysts have pitted their skills against each other. One side would have the upper hand for a while, only to give way when a new approach to code making or code breaking shifted the balance of power. (The competition is over, by the way. Guess which side won?)

**Mathematics.** If mathematics is your thing, you might want to investigate "information theory," the discipline created by MIT professor Claude Shannon just after World War II. How can we measure the "quantity" of information in a string of symbols; and how can some languages contain more information than others? Or you might want to tackle some of the "trap door" problems that are investigated as part of number theory, and which make public-key cryptography possible.


**History.** If history engages your interest you may be surprised to know that Mary Queen of Scots might have kept her head a little longer if her supporters had employed a more sophisticated nomenclature. Go back a little further in time and you will discover that the term "Sheshach" in the Old Testament is a code word for "Babel."

**Computers.** Is computer science your specialty? Strategies for integrating cryptosystems into communications protocols have led to such important developments as virtual private networking. Additionally, cryptography is an application for which algorithmic analysis simply cannot be ignored. The performance burden of strong cryptography leaves no "extra cycles" for lazy developers to waste. Similarly, hardware-software tradeoffs have seldom been more important.

**Language.** Presumably, we all learned in school that the discovery of the Rosetta Stone by Napoleon's army led to the understanding of Egyptian hieroglyphs. What is less well known, however, is the fact that many years of hard work by Jean-François Champollion, solving several knotty cryptanalysis problems, were needed before these strange symbols would give up their secrets. (By the way, if you plan to be involved in the search for extraterrestrial intelligence, more commonly known as "SETI," you would be well advised to have a cryptanalyst or two on your team.)

**Law.** Do law and politics appeal to you? How about the question of whether or not the citizens of a free country have a right to keep secrets from their government? This question has been at the center of legislation, law enforcement activity, national security, trade regulation and public debate for the last several decades. When cryptographic techniques that have been classified as military secrets since the 1940s are independently discovered by academic researchers, should the researchers be allowed to publish their results? Should they be allowed to exploit their discoveries in the form of commercial, information security products?

**Economics.** The advent of the World Wide Web holds out the promise of unprecedented wealth accumulation through the creation of a virtual global marketplace. But this marketplace can be exploited only if people who have never met—and never will meet—can find a way to trust each other. Will the answer be found in the form of public key infrastructure, digital signatures, digital cash and other tools of electronic commerce?

**The Future.** Even if none of the above areas of study are important to you, the so-called “esoteric protocols” may give you something to think about. Esoteric protocols deal with the application of cryptographic techniques to problems other than secrecy. Electronic elections, zero-knowledge proofs, anonymous messaging and blind signatures are examples. 

---

*Harry E. Smith is a past president of the ISSA Denver Chapter. He is the founder of Timberline Technologies LLC, an information security consulting firm. He can be reached at [harry\\_e\\_smith@timberlinetechnologies.com](mailto:harry_e_smith@timberlinetechnologies.com).*

