01074305 Computer Security

Chapter 2

Elementary Encryption

Charles P. Pfleeger & Shari Lawrence Pfleeger, Security in Computing, 4th Ed., Pearson Education, 2007

- **In this chapter**
  - Concepts of encryption
  - Cryptanalysis: how encryption systems are "broken"
  - Symmetric (secret key) encryption and the DES and AES algorithms
  - Asymmetric (public key) encryption and the RSA algorithm
  - Key exchange protocols and certificates
  - Digital signatures
  - Cryptographic hash functions

2.1. Terminology and Background

- Consider the steps involved in sending messages
  - from a **sender**, S, to a **recipient**, R
  - If S entrusts the message to T, who then delivers it to R, T then becomes the **transmission medium**
  - an outsider, O, wants to access the message (to read, change, or even destroy it), we call O an **interceptor** or **intruder**.
  - Any time after S transmits it via T, the message is vulnerable to exploitation

- O might try to access the message in any of the following ways:
  - **Block** it, by preventing its reaching R, thereby affecting the availability of the message.
  - **Intercept** it, by reading or listening to the message, thereby affecting the confidentiality of the message.
  - **Modify** it, by seizing the message and changing it in some way, affecting the message's integrity.
  - **Fabricate** an authentic-looking message, arranging for it to be delivered as if it came from S, thereby also affecting the integrity of the message.

- **Terminology**
  - **Encryption** is the process of encoding a message so that its meaning is not obvious;
  - **decryption** is the reverse process, transforming an encrypted message back into its normal, original form.
  - Alternatively, the terms **encode** and **decode** or **encipher** and **decipher** are used instead of encrypt and decrypt.
  - A system for encryption and decryption is called a **cryptosystem**.

Figure 2-1 Encryption.

- The original form of a message is known as **plaintext**, and the encrypted form is called **ciphertext**.
- Denote a plaintext message P as a sequence of individual characters $P = <p_1, p_2, ..., p_n>$.
- Similarly, ciphertext is written as $C = <c_1, c_2, ..., c_m>$.
- For instance, the plaintext message "I want cookies" can be denoted as the message string <I, ,w,a,n,t, , c,o,o,k,i,e,s>.
- It can be transformed into ciphertext $<c_1, c_2, ..., c_{14}>$, and the encryption algorithm tells us how the transformation is done.

Figure 2-1 Encryption.

- we write $C = E(P)$ and $P = D(C)$
  where C represents the ciphertext, E is the encryption rule,
  P is the plaintext D is the decryption rule.
- What we seek is a cryptosystem for which $P = D(E(P))$.
- In other words, we want to be able to convert the message to protect it from an intruder, but we also want to be able to get the original message back so that the receiver can read it properly.

- **Encryption Algorithms**
  - The cryptosystem involves a set of rules for how to encrypt the plaintext and how to decrypt the ciphertext.
  - The encryption and decryption rules, called **algorithms**, often use a device called a **key**, denoted by K,
  - The resulting ciphertext depends on the original plaintext message, the algorithm, and the key value.
  - We write this dependence as **C = E(K, P)**.

- **Encryption Algorithms (Cont'd)**
  - Sometimes the encryption and decryption keys are the same, so P = D(K, E(K,P)).
  - This form is called **symmetric** encryption because D and E are mirror-image processes.

- **Encryption Algorithms (Cont'd)**
  - At other times, encryption and decryption keys come in pairs. Then, a decryption key, K_D, inverts the encryption of key K_E so that P = D(K_D, E(K_E,P)).
  - Encryption algorithms of this form are called **asymmetric** because converting C back to P involves a series of steps and a key that are different from the steps and key of E.

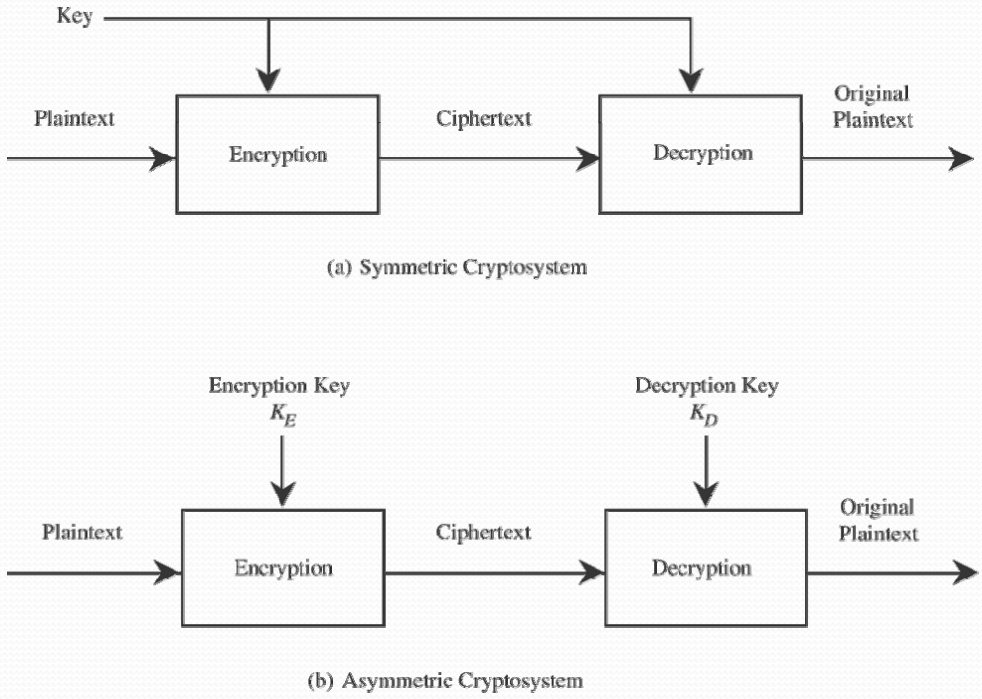


Figure 2-2 Encryption with Keys.

- **Encryption Algorithms (Cont'd)**
  - A key gives us flexibility in using an encryption scheme. We can create different encryptions of one plaintext message just by changing the key.
  - Moreover, using a key provides additional security. If the encryption algorithm should fall into the interceptor's hands, future messages can still be kept secret because the interceptor will not know the key value.
  - An encryption scheme that does not require the use of a key is called a **keyless cipher**.

- **Encryption Algorithms (Cont'd)**
  - The word **cryptography** means hidden writing, and it refers to the practice of using encryption to conceal text.
  - A **cryptanalyst** studies encryption and encrypted messages, hoping to find the hidden meanings.
  - Normally, a cryptographer works on behalf of a legitimate sender or receiver, whereas a cryptanalyst works on behalf of an unauthorized interceptor.
  - **Cryptology** is the research into and study of encryption and decryption; it includes both cryptography and cryptanalysis.

- **Cryptanalysis**
  - A cryptanalyst's chore is to **break** an encryption.
  - That is, the cryptanalyst attempts to deduce the original meaning of a ciphertext message.

- **Cryptanalysis**
  - a cryptanalyst can attempt to do any or all of six different things:
    - break a single message
    - recognize patterns in encrypted messages, to be able to break subsequent ones by applying a straightforward decryption algorithm
    - infer some meaning without even breaking the encryption, such as noticing an unusual frequency of communication or determining something by whether the communication was short or long
    - deduce the key, to break subsequent messages easily
    - find weaknesses in the implementation or environment of use of encryption
    - find general weaknesses in an encryption algorithm, without necessarily having intercepted any messages

- **Breakable Encryption**
  - An encryption algorithm is called breakable when, given enough time and data, an analyst can determine the algorithm.
  - However, an algorithm that is theoretically breakable may in fact be impractical to try to break.

- **Breakable Encryption (Cont'd)**
  - To see why, consider a 25-character message that is expressed in just uppercase letters.
  - A given cipher scheme may have 26^{25} (approximately 10^{35}) possible decipherments, so the task is to select the right one out of the 26^{25}.

- **Breakable Encryption (Cont'd)**
  - If your computer could perform on the order of 10^{10} operations per second, finding this decipherment would require on the order of 10^{16} seconds, or roughly 10^{11} years.
  - In this case, although we know that theoretically we could generate the solution, determining the deciphering algorithm by examining all possibilities can be ignored as infeasible with current technology.

- **Breakable Encryption (Cont'd)**
  - Two other important issues must be addressed when considering the breakability of encryption algorithms.
    - First, the cryptanalyst cannot be expected to try only the hard, long way. In the example just presented, the obvious decryption might require 26^{25} machine operations, but a more ingenious approach might require only 10^{15} operations. At the speed of 10^{10} operations per second, 10^{15} operations take slightly more than one day.

- **Breakable Encryption (Cont'd)**
  - Two other important issues must be addressed when considering the breakability of encryption algorithms.
    - Second, estimates of breakability are based on current technology.
    - A conjecture known as "Moore's Law" asserts that the speed of processors doubles every 1.5 years, and this conjecture has been true for over two decades.
    - It is risky to pronounce an algorithm secure just because it cannot be broken with current technology, or worse, that it has not been broken yet.

- **Representing Characters**
  - we use the convention that plaintext is written in **UPPERCASE** letters, and ciphertext is in **lowercase** letters.
  - Because most encryption algorithms are based on mathematical transformations, they can be explained or studied more easily in mathematical form.

- **Representing Characters**

Letter	A	B	C	D	E	F	G	H	I	J	K	L	M
Code	0	1	2	3	4	5	6	7	8	9	10	11	12

Letter	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Code	13	14	15	16	17	18	19	20	21	22	23	24	25

- **Representing Characters**
  - There are many types of encryption.
  - Two simple forms of encryption:
    - substitutions, in which one letter is exchanged for another
    - transpositions, in which the order of the letters is rearranged.

2.2. Substitution Ciphers

- The goal of substitution is *confusion*
- **The Caesar Cipher**
  - each letter is translated to the letter a fixed number of places after it in the alphabet.
  - Caesar used a shift of 3, so plaintext letter p_i was enciphered as ciphertext letter c_i by the rule

$c_i = E(p_i) = p_i + 3$

Plaintext	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
Ciphertext	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

Plaintext	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
Ciphertext	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

- The Caesar Cipher (Cont'd)
  - Using this encryption, the message

    TREATY IMPOSSIBLE

    would be encoded as

    T R E A T Y I M P O S S I B L E
    w u h d w b l p s r v v l e o h

- The Caesar Cipher (Cont'd)
  - Advantages and Disadvantages of the Caesar Cipher
    - The Caesar cipher is quite simple.
    - During Caesar's lifetime, the simplicity did not dramatically compromise the safety of the encryption because anything written, even in plaintext, was rather well protected; few people knew how to read!
    - Its obvious pattern is also the major weakness of the Caesar cipher. A secure encryption should not allow an interceptor to use a small piece of the ciphertext to predict the entire pattern of the encryption.

- **Cryptanalysis of the Caesar Cipher**
  - have many clues from the ciphertext
  - For example, the break between the two words is preserved in the ciphertext, and double letters are preserved

```
T R E A T Y   I M P O S S I B L E
w u h d w b   l p s r v v l e o h
```

- **Cryptanalysis of the Caesar Cipher**
  - Suppose you are given the following ciphertext message, and you want to try to determine the original plaintext.

```
wklv phvvdjh lv qrw wrr kdug wr euhdn
```

The message has actually been enciphered with a 27-symbol alphabet: A through Z plus the "blank"

- In **substitutions**, the alphabet is scrambled, and each plaintext letter maps to a unique ciphertext letter.
- A **permutation** is a reordering of the elements of a sequence
  - For instance, we can permute the numbers l to 10 in many ways, including the permutations n_1 = 1, 3, 5, 7, 9, 10, 8, 6, 4, 2; and n_2 = 10, 9, 8, 7, 6, 5, 4, 3, 2, 1.
  - A permutation is a function, so we can write expressions such as $n_1(3) = 5$ meaning that the letter in position 3 is to be replaced by the fifth letter.

- One way to scramble an alphabet is to use a **key**, a word that controls the permutation
- For instance,
  - if the key is word, the sender or receiver first writes the alphabet and then writes the key under the first few letters of the alphabet.
  - The sender or receiver then fills in the remaining letters of the alphabet, in some easy-to-remember order, after the keyword.

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
wordabcefghijklmnpqstuvxyz
```

- **Complexity of Substitution Encryption and Decryption**
  - An important issue in using any cryptosystem is the time it takes to turn plaintext into ciphertext, and vice versa.
  - The timing is directly related to the complexity of the encryption algorithm.

- **Cryptanalysis of Substitution Ciphers**
  - Try a guess and continue to work to substantiate that guess until you have all the words in place
  - Consider the difficulty of breaking a substitution cipher.
    - By using a **brute force attack**, the cryptanalyst could try all 26! permutations of a particular ciphertext message.
    - Working at one permutation per microsecond, it would still take over a thousand years to test all 26! possibilities.

- **Cryptanalysis of Substitution Ciphers (Cont'd)**
  - We can use our knowledge of language to simplify this problem.
    - For example, in English, some letters are used more often than others. The letters *E, T, O*, and *A* occur far more often than *J*, *Q*, *X*, and *Z*, for example.
    - When messages are long enough, the **frequency distribution analysis** quickly betrays many of the letters of the plaintext.

- **The Cryptographer's Dilemma**
  - Cryptanalyst works by finding patterns. Short messages give the cryptanalyst little to work with, so short messages are fairly secure with even simple encryption.
  - An encryption algorithm must be regular for it to be algorithmic and for cryptographers to be able to remember it. Unfortunately, the regularity gives clues to the cryptanalyst.

- One-Time Pads
  - A **one-time pad** is sometimes considered the perfect cipher.
  - The sender would write the keys one at a time above the letters of the plaintext and encipher the plaintext with a prearranged chart (called a **Vigenère tableau**). The sender would then destroy the used keys.
  - The receiver needs a pad identical to that of the sender
  - The one-time pad method has two problems:
    - the need for absolute synchronization between sender and receiver
    - the need for an unlimited number of keys.

- The Vernam Cipher
  - The **Vernam cipher** is a type of one-time pad devised by Gilbert Vernam for AT&T.
  - The basic encryption involves an arbitrarily long nonrepeating sequence of numbers that are combined with the plaintext.
  - As long as the key tape does not repeat or is not reused, this type of cipher is immune to cryptanalytic attack

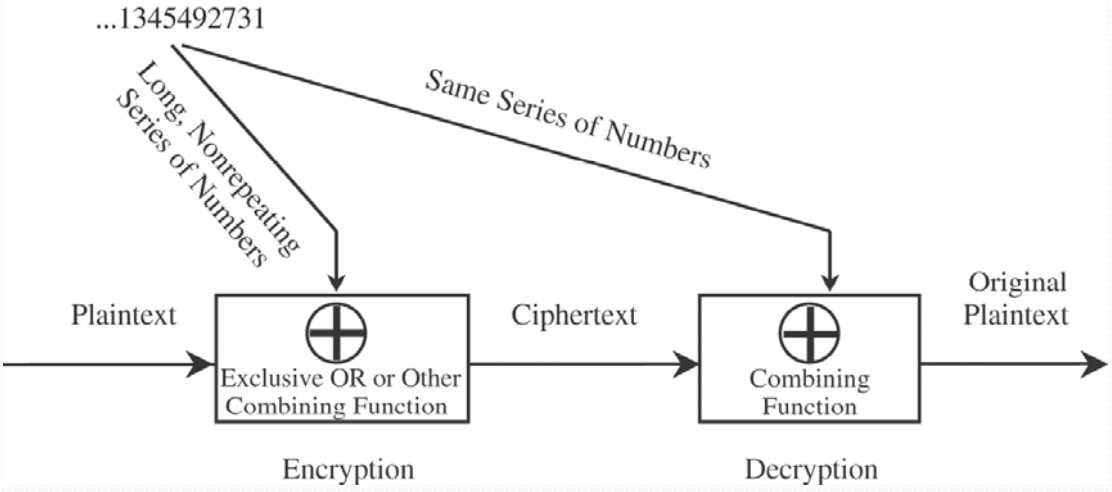


Figure 2-3 Vernam Cipher.

- A simple Vernam encryption

Plaintext	V	E	R	N	A	M	C	I	P	H	E	R
Numeric Equivalent	21	4	17	13	0	12	2	8	15	7	4	17
+ Random Number	76	48	16	82	44	3	58	11	60	5	48	88
= Sum	97	52	33	95	44	15	60	19	75	12	52	105
= mod 26	19	0	7	17	18	15	8	19	23	12	0	1
Ciphertext	t	a	h	r	s	p	i	t	x	m	a	b

- **Book Ciphers**
  - Another source of supposedly "random" numbers is any book, piece of music, or other object of which the structure can be analyzed.
  - Both the sender and receiver need access to identical objects.
  - As an example of a book cipher, you might select a passage from Descarte's meditation: *I am, I exist, that is certain.*
  - To encipher the message *MACHINES CANNOT THINK* by using the Descartes key

```
iamie xistt hatis cert
MACHI NESCA NNOTT HINK
```

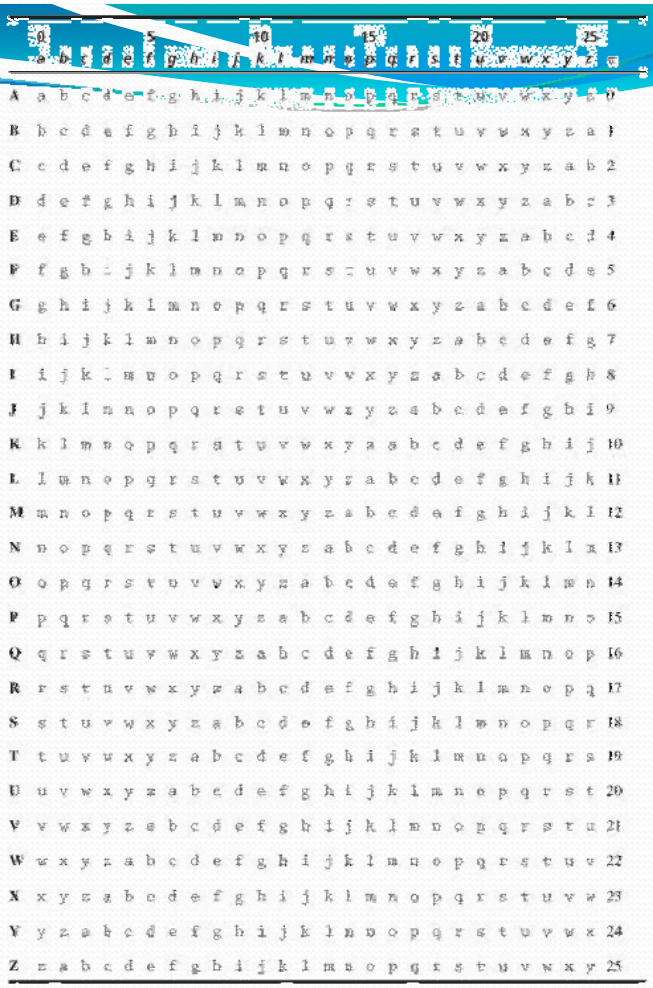



Table 2-1. Vigenère Tableau.

- If we use the substitution table shown as Table 2-1, this message would be encrypted as

```
uaopm kmkvt unhbl jmed
```

because row M column *i* is u, row A column *a* is a, and so on.

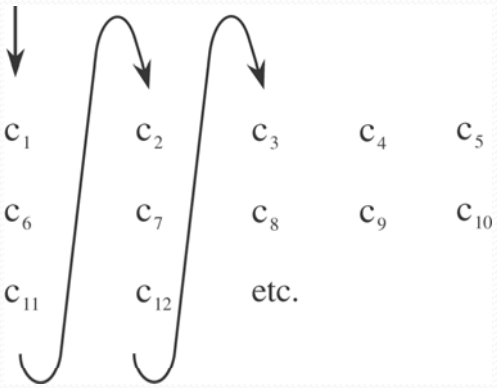
- It would seem as though this cipher, too, would be impossible to break. Unfortunately, that is not true. The flaw lies in the fact that neither the message nor the key text is evenly distributed; in fact, the distributions of both cluster around high-frequency letters

Ciphertext	u	a	o	p	m		k	m	k	v	t		u	n	h	b	l		j	m	e	d
Possible	?	A	A	?	E		?	E	?	?	A		?	A	O	N	?		?	E	A	?
plaintexts			O		I			I			T			N	T	T				I	E	
					T			T												T		

2.3. Transpositions (Permutations)

- A **transposition** is an encryption in which the letters of the message are rearranged.
- With transposition, the cryptography aims for ***diffusion***, widely spreading the information from the message or the key across the ciphertext.
- Transpositions try to break established patterns.
- Because a transposition is a rearrangement of the symbols of a message, it is also known as a **permutation**.

- **Columnar Transpositions**
  - he **columnar transposition** is a rearrangement of the characters of the plaintext into columns.



- **Columnar Transpositions** (Con'td)
  - For instance, suppose you want to write the plaintext message THIS IS A MESSAGE TO SHOW HOW A COLUMNAR TRANSPOSITION WORKS. We arrange the letters in five columns as

```
T H I S I
S A M E S
S A G E T
O S H O W
H O W A C
O L U M N
A R T R A
N S P O S
I T I O N
W O R K S
```

The resulting ciphertext would then be read down the columns as

tssoh oaniw haaso lrsto
imghw utpir seeoa mrook
istwc nasns

- **Combinations of Approaches**
  - Substitution and transposition can be considered as building blocks for encryption.
  - Other techniques can be based on each of them, both of them, or a combination with yet another approach.
  - A combination of two ciphers is called a **product cipher**.
  - Product ciphers are typically performed one after another, as in $E_2(E_1(P,k_1), k_2)$.
  - Just because you apply two ciphers does not necessarily mean the result is any stronger than, or even as strong as, either individual cipher.

2.4. Making "Good" Encryption Algorithms

- What Makes a "Secure" Encryption Algorithm?
  - Shannon's Characteristics of "Good" Ciphers
    1. The amount of secrecy needed should determine the amount of labor appropriate for the encryption and decryption.
    2. The set of keys and the enciphering algorithm should be free from complexity.
    3. The implementation of the process should be as simple as possible.
    4. Errors in ciphering should not propagate and cause corruption of further information in the message.
    5. The size of the enciphered text should be no larger than the text of the original message.

- Properties of "Trustworthy" Encryption Systems
  - when we say that encryption is "commercial grade," or "trustworthy," we mean that it meets these constraints:
    - *It is based on sound mathematics.*
    - *It has been analyzed by competent experts and found to be sound.*
    - *It has stood the "test of time."*

- Symmetric and Asymmetric Encryption Systems
  - Symmetric algorithms
    - use one key, which works for both encryption and decryption.
    - The symmetric systems provide a two-way channel to their users:
      - A and B share a secret key, and they can both encrypt information to send to the other as well as decrypt information from the other.
      - As long as the key remains secret, the system also provides **authentication** proof that a message received was not fabricated by someone other than the declared sender.
    - symmetric encryption systems require a means of **key distribution**

- Symmetric and Asymmetric Encryption Systems
  - Asymmetric algorithms
    - Public key systems excel at key management.
    - By the nature of the public key approach, you can send a public key in an e-mail message or post it in a public directory.
    - Only the corresponding private key, which presumably is kept private, can decrypt what has been encrypted with the public key.
  - For all encryption algorithms, **key management** is a major issue. It involves storing, safeguarding, and activating keys.

- **Stream and Block Ciphers**
  - **Stream ciphers** convert one symbol of plaintext immediately into a symbol of ciphertext.
  - Some kinds of errors, such as skipping a character in the key during encryption, affect the encryption of all future characters

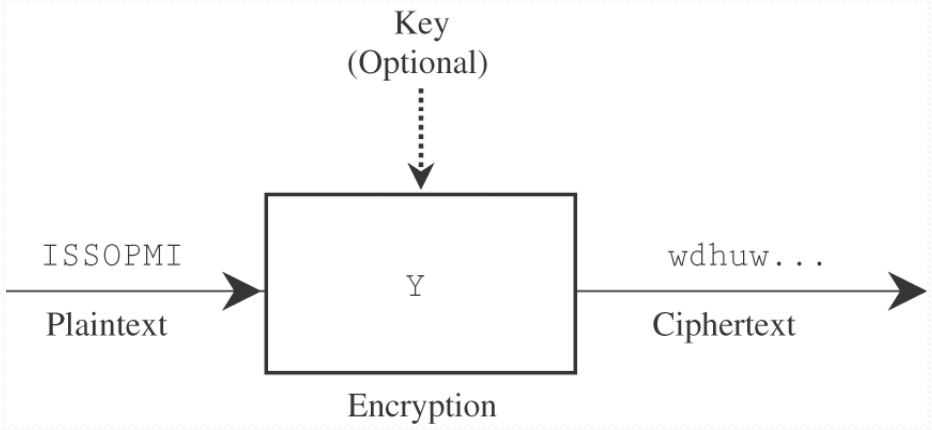


Figure 2-6 Stream Encryption.

- **Stream and Block Ciphers**
  - A **block cipher** encrypts a *group* of plaintext symbols as *one* block.

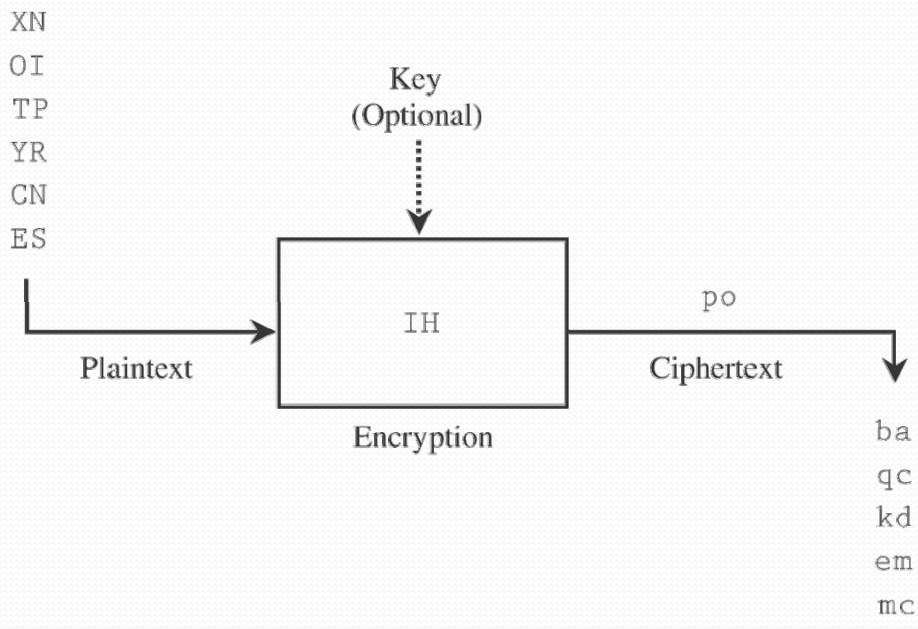


Figure 2-7 Block Cipher Systems.

- **Confusion and Diffusion**
  - **Confusion** - the interceptor should not be able to predict what will happen to the ciphertext by changing one character in the plaintext.
  - **Diffusion** - the cipher should also spread the information from the plaintext over the entire ciphertext so that changes in the plaintext affect many parts of the ciphertext.

- **Cryptanalysis Breaking Encryption Schemes**
  - Four possible situations confront the cryptanalyst, depending on what information is available:
    - ciphertext
    - full plaintext
    - partial plaintext
    - algorithm

- Cryptanalysis Breaking Encryption Schemes (Cont'd)
  - **Ciphertext Only - t**he decryption had to be based on probabilities, distributions, and characteristics of the available ciphertext, plus publicly available knowledge.
  - **Full or Partial Plaintext**
    - The analyst may be fortunate enough to have a sample message and its decipherment
    - The interceptor has both *C* and *P* and needs only to deduce the *E* for which $C = E(P)$ to find *D*
    - In this case the analyst is attempting to find *E*(or *D*) by using a **known plaintext** attack.

- Cryptanalysis Breaking Encryption Schemes (Cont'd)
  - **Full or Partial Plaintext (Cont'd)**
    - The analyst may have additional information - for example, the analyst may know that the message was intercepted from a diplomatic exchange between Germany and Austria. From that information, the analyst may guess that the words Bonn, Vienna, and Chancellor appear in the message.
    - The analyst can use what is called a **probable plaintext** analysis
    - After cryptanalysis has provided possible partial decipherments, a probable plaintext attack may permit a cryptanalyst to fill in some blanks.

- Cryptanalysis Breaking Encryption Schemes (Cont'd)
  - Ciphertext of Any Plaintext
    - A **chosen plaintext** attack - The analyst may have infiltrated the sender's transmission process so as to be able to cause messages to be encrypted and sent at will.
  - Algorithm and Ciphertext
    - A **chosen ciphertext** attack - The analyst can run the algorithm on massive amounts of plaintext to find one plaintext message that encrypts as the ciphertext.

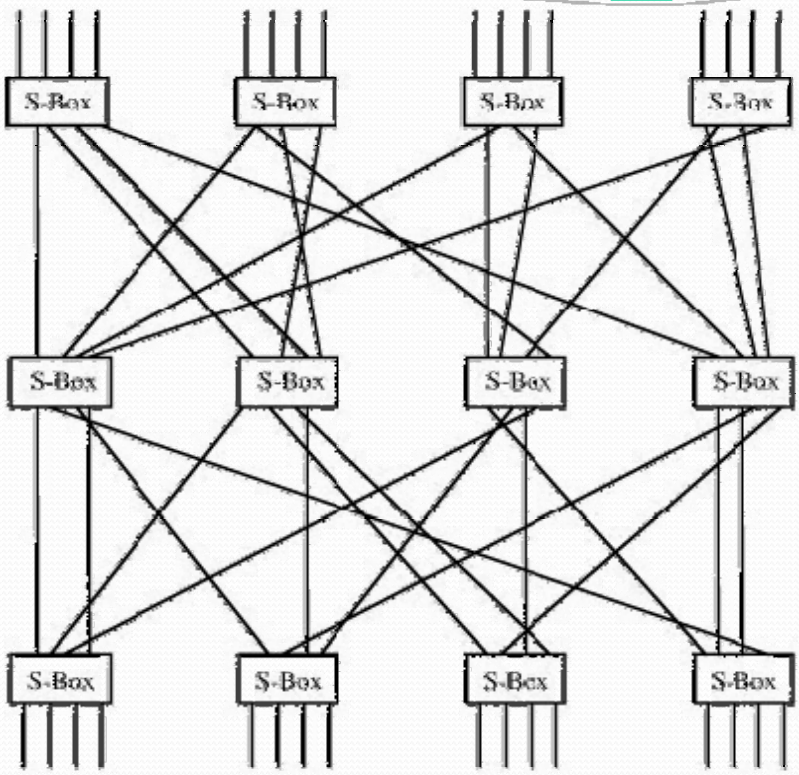
- Cryptanalysis Breaking Encryption Schemes (Cont'd)
  - Ciphertext and Plaintext
    - The cryptanalyst may be lucky enough to have some pairs of plaintext and matching ciphertext.
    - Then, the game is to deduce the key by which those pairs were encrypted so that the same key can be used in cases in which the analyst has only the ciphertext

- **Cryptanalysis Breaking Encryption Schemes (Cont'd)**
  - **Weaknesses**
    - A cryptanalyst works against humans, who can be hurried, lazy, careless, naïve, or uninformed.

Symmetric Encryption

- Confusion is the act of creating ciphertext so that its corresponding plaintext is not apparent. Substitution is the basic tool for confusion
- Diffusion is the act of spreading the effect of a change in the plaintext throughout the resulting ciphertext.

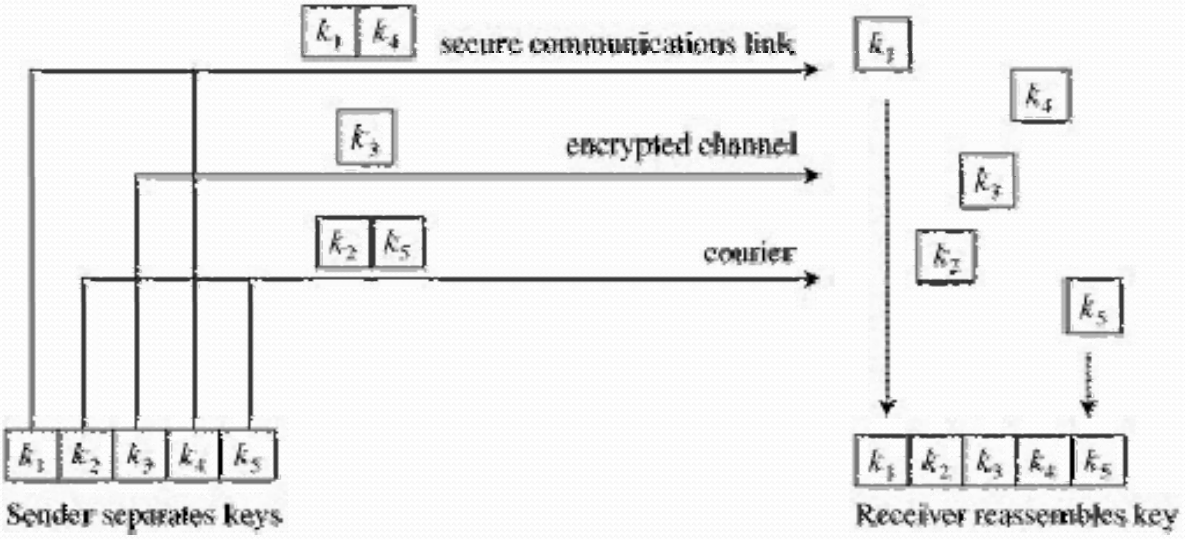
- Substitution is sometimes represented by so-called **S-boxes**, which are nothing other than table-driven substitutions.
- Diffusion can be accomplished by permutations, or "**P-boxes.**"
- Strong cryptosystems may use several iterations of a substitute-permute cycle.



Substitutions and Permutations.

- **Problems of Symmetric Key Systems**
  1. As with all key systems, if the key is revealed (stolen, guessed, bought, or otherwise compromised), the interceptors can immediately decrypt all the encrypted information they have available. Furthermore, an impostor using an intercepted key can produce bogus messages under the guise of a legitimate sender. For this reason, in secure encryption systems, the keys are changed fairly frequently so that a compromised key will reveal only a limited amount of information.

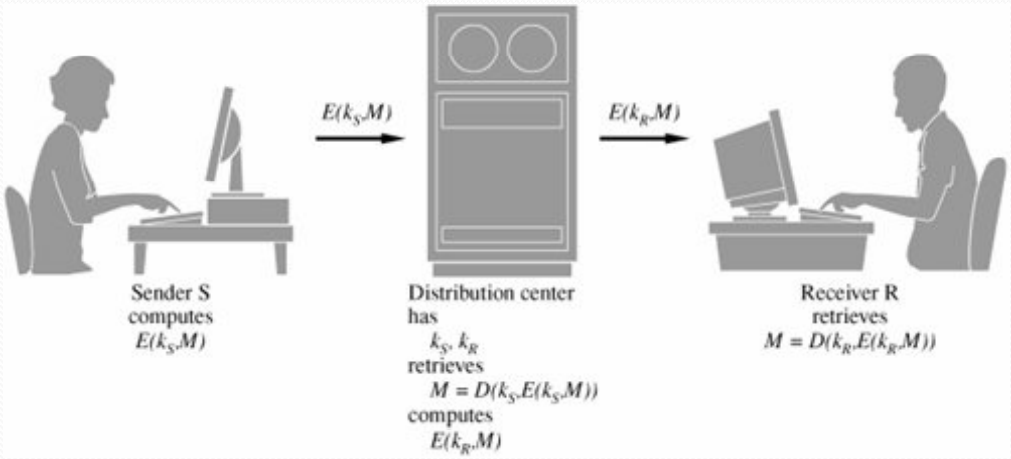
- **Problems of Symmetric Key Systems (Cont'd)**
  2. Distribution of keys becomes a problem. Keys must be transmitted with utmost security since they allow access to all information encrypted under them. For applications that extend throughout the world, this can be a complex task. Often, couriers are used to distribute the keys securely by hand. Another approach is to distribute the keys in pieces under separate channels so that any one discovery will not produce a full key.



Key Distribution in Pieces.

- **Problems of Symmetric Key Systems (Cont'd)**

3. the number of keys increases with the square of the number of people exchanging secret information. This problem is usually contained by having only a few people exchange secrets directly so that the network of interchanges is relatively small. If people in separate networks need to exchange secrets, they can do so through a central "**clearing house**" or "**forwarding office**" that accepts secrets from one person, decrypts them, reencrypts them using another person's secret key, and transmits them.



Distribution Center for Encrypted Information

2.5. The Data Encryption Standard

- **Background and History**
  - In the early 1970s, the U.S. National Bureau of Standards (NBS) recognized that the general public needed a secure encryption technique for protecting sensitive information
  - Based on IBM **Lucifer** algorithm
  - The DES was officially adopted as a U.S. federal standard in November 1976, authorized by NBS for use on all public and private sector unclassified communication

- **Overview of the DES Algorithm**
  - A careful and complex combination of two fundamental building blocks of encryption: **substitution** and **transposition**
  - The algorithm derives its strength from repeated application of these two techniques, one on top of the other, for a total of 16 cycles.

- **Overview of the DES Algorithm (Cont'd)**
  - The algorithm begins by encrypting the plaintext as blocks of 64 bits.
  - The key is 64 bits long, but in fact it can be any 56-bit number. (The extra 8 bits are often used as check digits and do not affect encryption in normal implementations.)

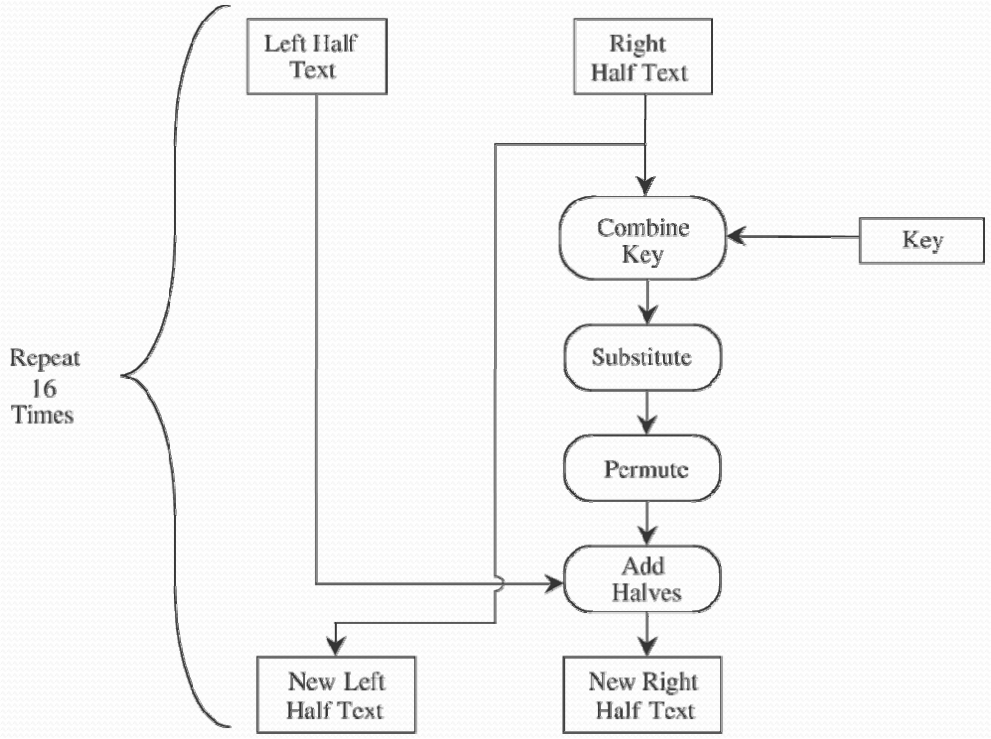
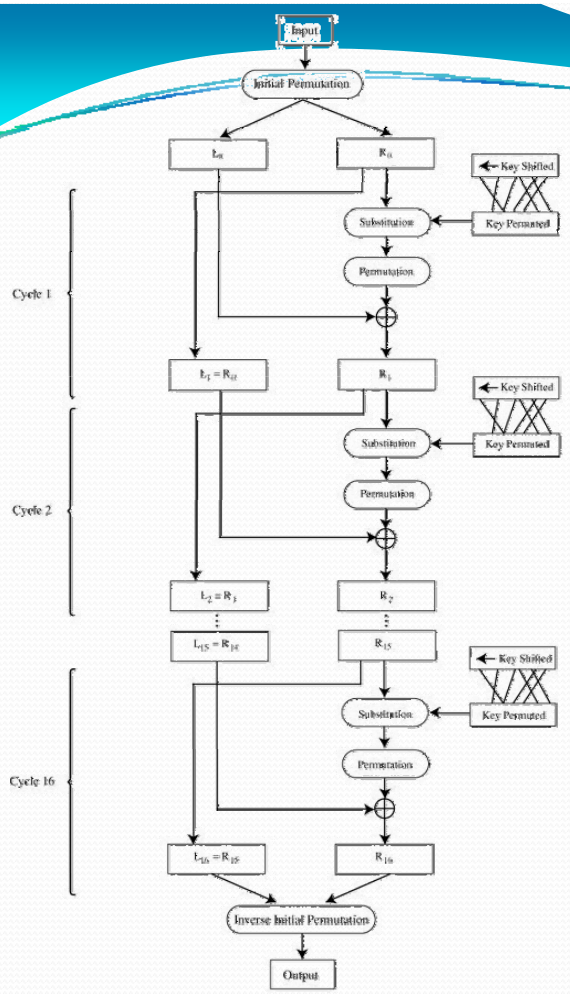
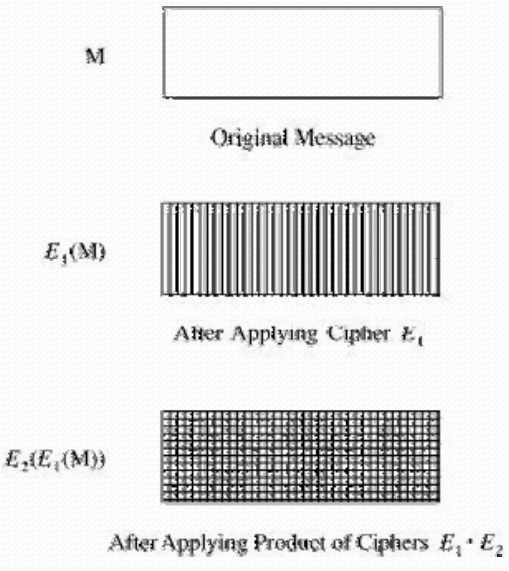


Figure 2-8 Cycles of Substitution and Permutation.



- The basis of the DES is two different ciphers, applied alternately
- a structure called a **product cipher**.

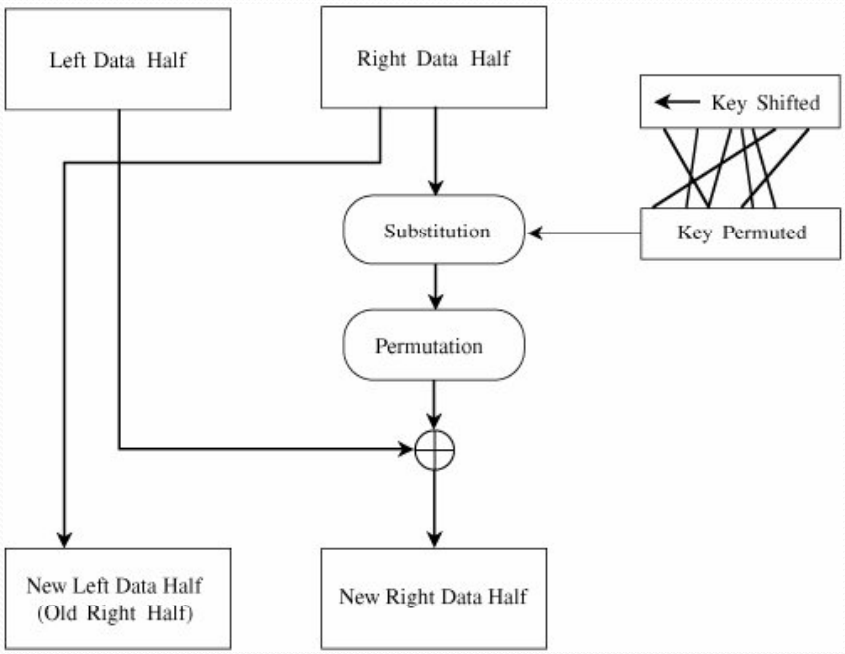


- **Details of the Encryption Algorithm**
  - After initialization, the DES algorithm operates on blocks of data.
  - It splits a data block in half, scrambles each half independently, combines the key with one half, and swaps the two halves.
  - This process is repeated 16 times.
  - It is an iterative algorithm using just table lookups and simple bit operations.

- **Details of the Encryption Algorithm (Cont'd)**
  - Input to the DES is divided into blocks of 64 bits.
  - The 64 data bits are permuted by a so-called initial permutation.
  - The data bits are transformed by a 64-bit key (of which only 56 bits are used).
  - The key is reduced from 64 bits to 56 bits by dropping bits 8, 16, 24, ... 64 (where the most significant bit is named bit "1").
  - These bits are assumed to be parity bits that carry no information in the key.

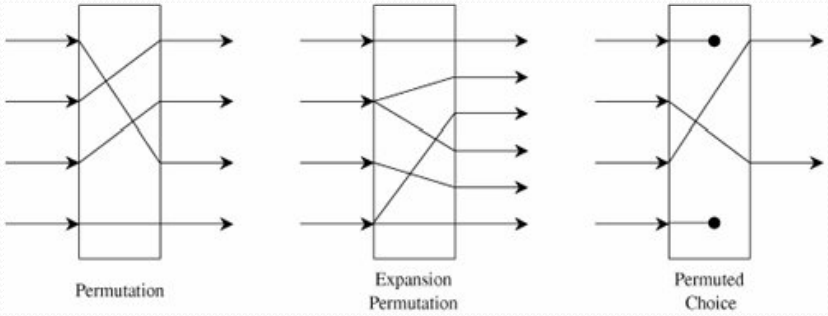
- **Details of the Encryption Algorithm (Cont'd)**
  - Next begins the sequence of operations known as a **cycle**.
  - The 64 permuted data bits are broken into a left half and a right half of 32 bits each.
  - The key is shifted left by a number of bits and permuted.
  - The key is combined with the right half, which is then combined with the left half.

- **Details of the Encryption Algorithm (Cont'd)**
  - The result of these combinations becomes the new right half; the old right half becomes the new left half.
  - This sequence of activities, which constitutes a cycle.
  - The cycles are repeated 16 times.
  - After the last cycle is a final permutation, which is the inverse of the initial permutation.

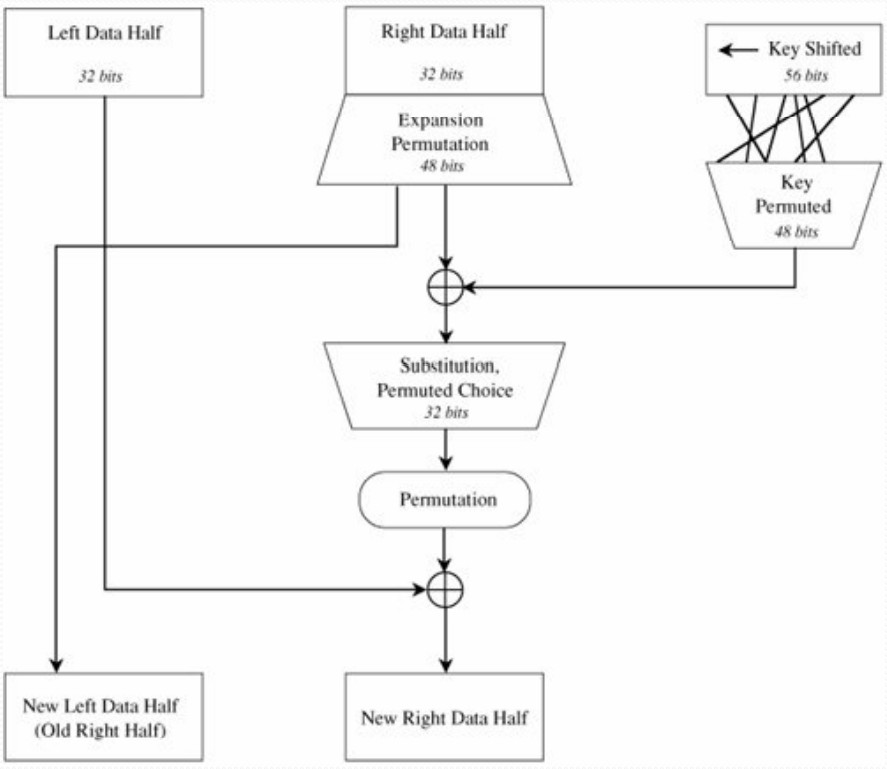


A Cycle in the DES.

- **Details of the Encryption Algorithm (Cont'd)**
  - For a 32-bit right half to be combined with a 64-bit key, two changes are needed. First, the algorithm expands the 32-bit half to 48 bits by repeating certain bits, while reducing the 56-bit key to 48 bits by choosing only certain bits.

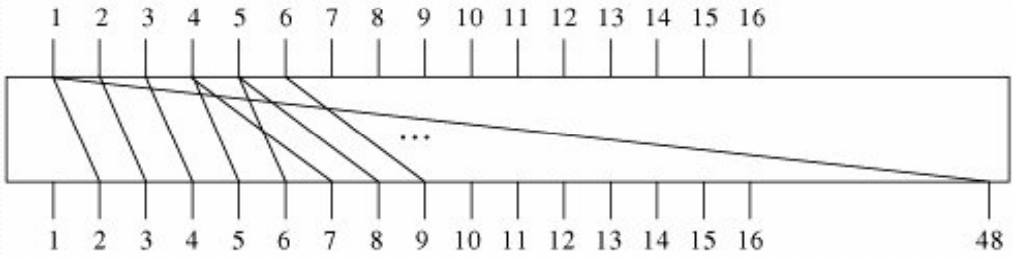


- **Details of Each Cycle of the Algorithm**
  - Each cycle of the algorithm is really four separate operations.
  - First, a right half is expanded from 32 bits to 48.
  - Then, it is combined with a form of the key.
  - The result of this operation is then substituted for another result and condensed to 32 bits at the same time.
  - The 32 bits are permuted and then combined with the left half to yield a new right half



- **Expansion Permutation**
  - Each right half is expanded from 32 to 48 bits by means of the expansion permutation.
  - The expansion permutes the order of the bits and also repeats certain bits.
  - The expansion has two purposes:
    - To make the intermediate halves of the ciphertext comparable in size to the key
    - To provide a longer result that can later be compressed.

Bit	1	2	3	4	5	6	7	8
Moves to Position	2,48	3	4	5,7	6,8	9	10	11,13
Bit	9	10	11	12	13	14	15	16
Moves to Position	12,14	15	16	17,19	18,20	21	22	23,25
Bit	17	18	19	20	21	22	23	24
Moves to Position	24,26	27	28	29,31	30,32	33	34	35,37
Bit	25	26	27	28	29	30	31	32
Moves to Position	36,38	39	40	41,43	42,44	45	46	47,1



Pattern of Expansion Permutation

- **Key Transformation**
  - The 64-bit key immediately becomes a 56-bit key by deletion of every eighth bit.
  - At each step in the cycle, the key is split into two 28-bit halves, the halves are shifted left by a specified number of digits, the halves are then pasted together again, and 48 of these 56 bits are permuted to use as a key during this cycle.

- **Key Transformation (Cont'd)**
  - Next, the key for the cycle is combined by an exclusive OR function with the expanded right half.
  - That result moves into the S-boxes we are about to describe.
  - At each cycle, the halves of the key are independently shifted left circularly by a specified number of bit positions.

Cycle Number	Bits Shifted
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

- **Key Transformation (Cont'd)**
  - After being shifted, 48 of the 56 bits are extracted for the exclusive OR combination with the expanded right half

Key Bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Selected for Position	5	24	7	16	6	10	20	18	—	12	3	15	23	1
Key Bit	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Selected for Position	9	19	2	—	14	22	11	—	13	4	—	17	21	8
Key Bit	29	30	31	32	33	34	35	36	37	38	39	40	41	42
Selected for Position	47	31	27	48	35	41	—	46	28	—	39	32	25	44
Key Bit	43	44	45	46	47	48	49	50	51	52	53	54	55	56
Selected for Position	—	37	34	43	29	36	38	45	33	26	42	—	30	40

- **S-Boxes**
  - Substitutions are performed by eight **S-boxes.**
  - An S-box is a permuted choice function by which six bits of data are replaced by four bits.
  - The 48-bit input is divided into eight 6-bit blocks, identified as $B_1B_2 \ldots B_8$; blockB_j is operated on by S-box S_j.

Box	Row	Column 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

- P-Boxes
  - After an S-box substitution, all 32 bits of a result are permuted by a straight permutation, *P*.

Bit	Goes to Position							
1-8	9	17	23	31	13	28	2	18
9-16	24	16	30	6	26	20	10	1
17-24	8	14	25	3	4	29	11	19
25-32	32	12	22	7	5	27	15	21

- Initial permutation

Bit	Goes to Position							
1-8	40	8	48	16	56	24	64	32
9-16	39	7	47	15	55	23	63	31
17-24	38	6	46	14	54	22	62	30
25-32	37	5	45	13	53	21	61	29
33-40	36	4	44	12	52	20	60	28
41-48	35	3	43	11	51	19	59	27
49-56	34	2	42	10	50	18	58	26
57-64	33	1	41	9	49	17	57	25

- Final permutation (or inverse initial permutation)

Bit	Goes to Position							
1-8	58	50	42	34	26	18	10	2
9-16	60	52	44	36	28	20	12	4
17-24	62	54	46	38	30	22	14	6
25-32	64	56	48	40	32	24	16	8
33-40	57	49	41	33	25	17	9	1
41-48	59	51	43	35	27	19	11	3
49-56	61	53	45	37	29	21	13	5
57-64	63	55	47	39	31	23	15	7

- **Decryption of the DES**
  - The same DES algorithm is used both for encryption *and decryption*.
  - The only change is that the keys must be taken in reverse order (k_{16}, k_{15}, ..., k_1) for decryption.

- **Questions About the Security of the DES**
  - **Design of the Algorithm**
    - Initially, there was concern with the basic algorithm itself.
      - During development of the algorithm, the National Security Agency (NSA) indicated that key elements of the algorithm design were "sensitive" and would not be made public.
      - These elements include the rationale behind transformations by the S-boxes, the P-boxes, and the key changes.
      - There are many possibilities for the S-box substitutions, but one particular set was chosen for the DES.

- Questions About the Security of the DES (Cont'd)
  - Design of the Algorithm (Cont'd)
    - Two issues arose about the design's secrecy.
      - The first involved a fear that certain "trapdoors" had been embedded in the DES algorithm so that a covert, easy means was available to decrypt any DES-encrypted message. For instance, such trapdoors would give NSA the ability to inspect private communications.
      - The second issue addressed the possibility that a design flaw would be (or perhaps has been) discovered by a cryptanalyst, this time giving an interceptor the ability to access private communications.

- Questions About the Security of the DES (Cont'd)
  - Design of the Algorithm (Cont'd)
    - The NSA released certain information on the selection of the S-boxes
      - No S-box is a linear or affine function of its input; that is, the four output bits cannot be expressed as a system of linear equations of the six input bits.
      - Changing one bit in the input of an S-box results in changing at least two output bits; that is, the S-boxes diffuse their information well throughout their outputs.

- Questions About the Security of the DES (Cont'd)
  - Design of the Algorithm (Cont'd)
    - The NSA released certain information on the selection of the S-boxes
      - The S-boxes were chosen to minimize the difference between the number of 1s and 0s when any single input bit is held constant; that is, holding a single bit constant as a 0 or 1 and changing the bits around it should not lead to disproportionately many 0s or 1s in the output.

- Questions About the Security of the DES (Cont'd)
  - Number of iterations
    - Many analysts wonder whether 16 iterations are sufficient. Since each iteration diffuses the information of the plaintext throughout the ciphertext, it is not clear that 16 cycles diffuse the information sufficiently
    - Experimentation with both the DES and its IBM predecessor Lucifer was performed by the NBS and by IBM as part of the certification process of the DES algorithm. These experiments have shown that 8 iterations are sufficient to eliminate any observed dependence. Thus, the 16 iterations of the DES should surely be adequate.

- Questions About the Security of the DES (Cont'd)
  - Key Length
    - The length of the key is the most serious objection raised.
      - The key in the original IBM implementation of Lucifer was 128 bits, whereas the DES key is effectively only 56 bits long.
      - The attack strategy is the "brute force" attack
        - If someone could test one every 100 milliseconds, the time to test all keys would be $7.2 * 10^{15}$ seconds, or about 228 million years.
        - If the test took only one microsecond, then the total time for the search is (only!) about 2,280 years.

- Double and Triple DES
  - Double DES
    - The double encryption works in the following way.
      - Take two keys, k_1 and k_2, and perform two encryptions, one on top of the other: $E(k_2, E(k_1,m))$.
      - Merkle and Hellman [MER81] showed that two encryptions are no better than one.
      - The double encryption only doubles the work for the attacker

- Double and Triple DES
  - Triple DES
    - $C = E(k_3, E(k_2, E(k_1,m)))$ - that is, you encrypt with one key, decrypt with the second, and encrypt with a third.
    - This process gives a strength **equivalent to a 112-bit key** (because the double DES attack defeats the strength of one of the three keys).
    - A minor variation of triple DES, which some people also confusingly call triple DES, is $C = E(k_1, D(k_2,E(k_1,m)))$. This approach is subject to another tricky attack, so its strength **is rated at only about 80 bits**.

- Security of the DES
  - Since its was first announced, DES has been controversial
  - Much of this controversy has appeared in the open literature, but certain DES features have neither been revealed by the designers nor inferred by outside analysts
  - In 1997 researchers using over 3,500 machines in parallel were able to infer a DES key in four months' work.
  - In 1998 for approximately $100,000, researchers built a special "DES cracker" machine that could find a DES key in approximately four days.

2.6. The AES Encryption Algorithm

- **The AES Contest**
  - In January 1997, NIST called for cryptographers to develop a new encryption system
  - The algorithms had to be
    - unclassified
    - publicly disclosed
    - available royalty-free for use worldwide
    - symmetric block cipher algorithms, for blocks of 128 bits
    - usable with key sizes of 128, 192, and 256 bits

- **The AES Contest (Cont'd)**
  - In August 1998, fifteen algorithms were chosen from among those submitted
  - In August 1999, the field of candidates was narrowed to five finalists.
  - The five then underwent extensive public and private scrutiny.
  - The final selection was made on the basis not only of security but also of cost or efficiency of operation and ease of implementation in software.

- **The AES Contest (Cont'd)**
  - The winning algorithm, submitted by two Dutch cryptographers, was Rijndael
  - The AES was adopted for use by the U.S. government in December 2001 and became Federal Information Processing Standard 197

- **Overview of Rijndael**
  - **Rijndael** is a fast algorithm that can be implemented easily on simple processors.
  - Primarily uses **substitution**; **transposition**; and the **shift**, **exclusive OR**, and **addition** operations.
  - Like DES, AES uses repeat cycles - there are 10, 12, or 14 cycles for keys of 128, 192, and 256 bits, respectively.
  - In Rijndael, the **cycles** are called "**rounds.**"

- **Structure of the AES (Cont'd)**
  - It is convenient to think of a 128-bit block of AES as a 4 x 4 matrix, called the "state." We present the state here as the matrix *s*[0,0]..*s*[3,3].

b_0	b_4	b_8	b_{12}		$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
b_1	b_5	b_9	b_{13}		$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
b_2	b_6	b_{10}	b_{14}		$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
b_3	b_7	b_{11}	b_{15}		$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

- **Overview of Rijndael (Cont'd)**
  - Each cycle consists of four steps.
    - ***Byte substitution*** - substituting each byte of a 128-bit block according to a substitution table. This is a straight diffusion operation.
    - ***Shift row:*** A transposition step.
      - For 128- and 192-bit block sizes, row *n* is shifted left circular (*n* - 1) bytes; for 256-bit blocks, row 2 is shifted 1 byte and rows 3 and 4 are shifted 3 and 4 bytes, respectively.
      - This is a straight confusion operation.

- **Overview of Rijndael (Cont'd)**
  - Each cycle consists of four steps.
    - *Mix column:*
      - This step involves shifting left and exclusive-ORing bits with themselves.
      - These operations provide both confusion and diffusion.
    - *Add subkey:*
      - Here, a portion of the key unique to this cycle is exclusive-ORed with the cycle result.
      - This operation provides confusion and incorporates the key.

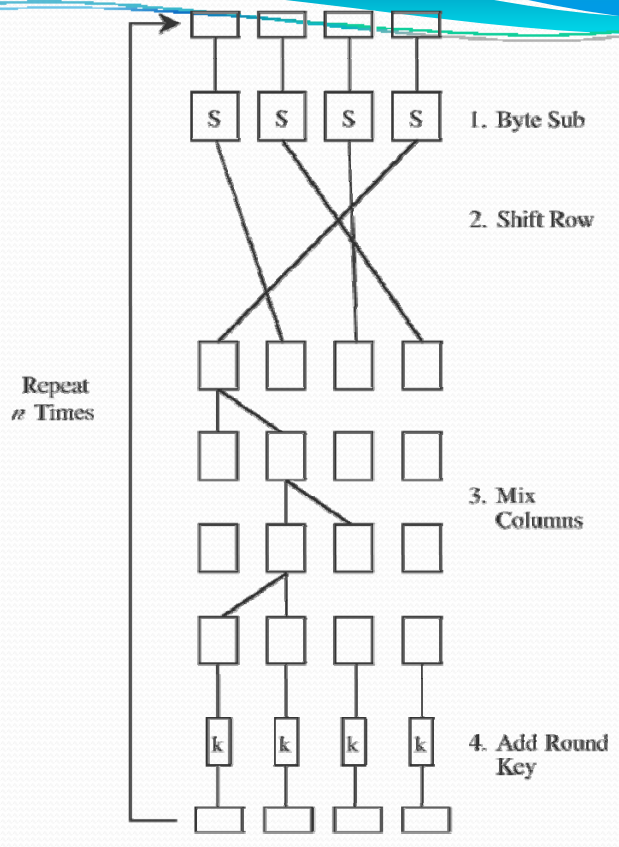


Figure 2-9 AES Algorithm.

- **Strength of the Algorithm**
  - The Rijndael algorithm is quite new
  - However, between its submission as a candidate for AES in 1997 and its selection in 2001, it underwent extensive cryptanalysis by both government and independent cryptographers.
  - Its Dutch inventors have no relationship to the NSA or any other part of the U.S. government, so there is no suspicion that the government somehow weakened the algorithm or added a trapdoor.

- **Comparison of DES and AES**

	DES	AES
Date	1976	1999
Block size	64 bits	128 bits
Key length	56 bits (effective length)	128, 192, 256 (and possibly more) bits
Encryption primitives	Substitution, permutation	Substitution, shift, bit mixing
Cryptographic primitives	Confusion, diffusion	Confusion, diffusion
Design	Open	Open
Design rationale	Closed	Open
Selection process	Secret	Secret, but accepted open public comment
Source	IBM, enhanced by NSA	Independent Dutch cryptographers

2.7. Public Key Encryption

- In 1976, Diffie and Hellman proposed a new kind of encryption system.
- With a public key encryption system, each user has a key that does not have to be kept secret.
- Although counterintuitive, in fact the public nature of the key does not compromise the secrecy of the system.
- Instead, the basis for public key encryption is to allow the key to be divulged but to keep the decryption technique secret.
- Public key cryptosystems accomplish this goal by using two keys: one to encrypt and the other to decrypt

- **Motivation**
  - In general, an *n*-user system requires $n * (n - 1)/2$ keys, and each user must track and remember a key for each other user with which he or she wants to communicate.
  - As the number of users grows, the number of keys increases very rapidly

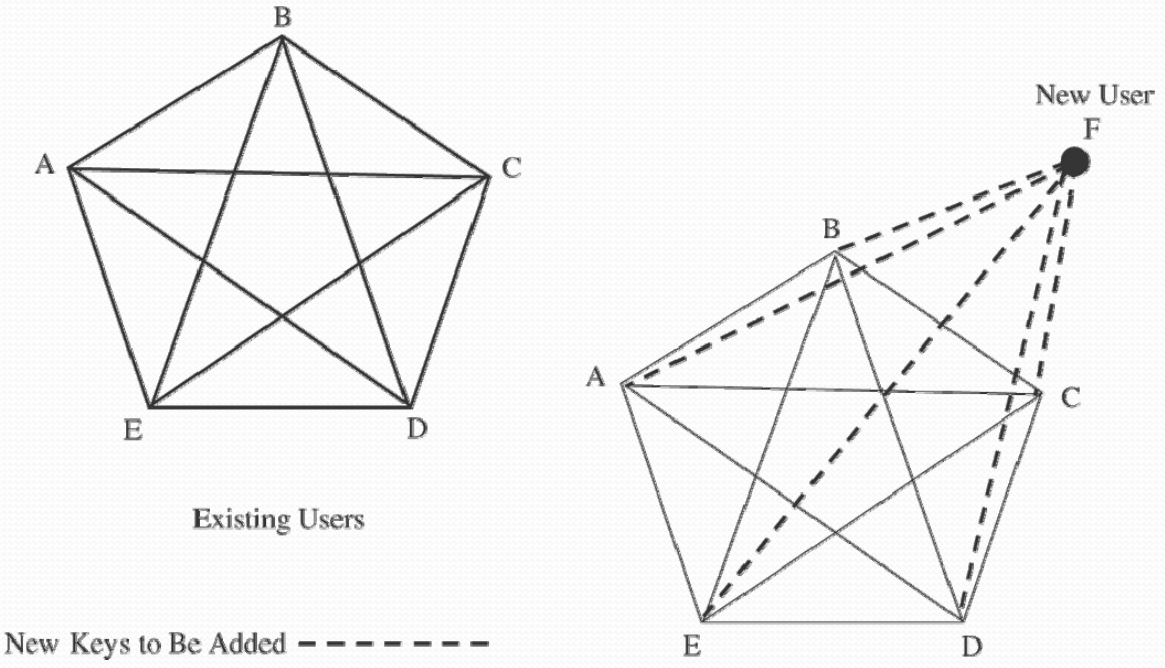


Figure 2-10 Key Proliferation.

- **Characteristics**
  - In a **public key** or **asymmetric encryption system**, each user has two keys: a public key and a private key.
  - The user may publish the public key freely because each key does only half of the encryption and decryption process.
  - The keys operate as inverses, meaning that one key undoes the encryption provided by the other key.

- **Characteristics (Cont'd)**
  - Let k_{PRIV} be a user's private key, and let k_{PUB} be the corresponding public key. Then, encrypted plaintext using the public key is decrypted by application of the private key; we write the relationship as

$$P = D(k_{PRIV}, E(k_{PUB}, P))$$

- **Characteristics (Cont'd)**
  - That is, a user can decode with a private key what someone else has encrypted with the corresponding public key. Furthermore, with some public key encryption algorithms, including RSA, we have this relationship:

$$P = D(k_{PUB}, E(k_{PRIV}, P))$$

- Characteristics (Cont'd)

	Secret Key (Symmetric)	Public Key (Asymmetric)
Number of keys	1	2
Protection of key	Must be kept secret	One key must be kept secret; the other can be freely exposed
Best uses	Cryptographic workhorse; secrecy and integrity of datasingle characters to blocks of data, messages, files	Key exchange, authentication
Key distribution	Must be out-of-band	Public key can be used to distribute other keys
Speed	Fast	Slow; typically, 10,000 times slower than secret key

- Rivest Shamir Adelman Encryption
  - The algorithm was introduced in 1978 and to date remains secure.
  - RSA has been the subject of extensive cryptanalysis, and no serious flaws have yet been found.
  - RSA relies on an area of mathematics known as number theory, in which mathematicians study properties of numbers such as their prime factors.

- **Rivest Shamir Adelman Encryption (Cont'd)**
  - The two keys used in RSA, *d* and *e*, are used for decryption and encryption.
  - They are actually interchangeable:
    - Either can be chosen as the public key, but one having been chosen,
    - the other one must be kept private.
    - For simplicity, we call the encryption key *e* and the decryption key *d*.
    - Also, because of the nature of the RSA algorithm, the keys can be applied in either order:

$$P = E(D(P)) = D(E(P))$$

- **Rivest Shamir Adelman Encryption (Cont'd)**
  - Any plaintext block *P* is encrypted as P^e mod *n*.
  - Because the exponentiation is performed mod *n*, factoring P^e to uncover the encrypted plaintext is difficult.
  - However, the decrypting key *d* is carefully chosen so that $(P^e)^d$ mod $n = P$.
  - Thus, the legitimate receiver who knows *d* simply computes $(P^e)^d$ mod $n = P$ and recovers *P* without having to factor P^e.

- **Detailed Description of the Encryption Algorithm**
  - The RSA algorithm uses two keys, d and e, which work in pairs, for decryption and encryption, respectively. A plaintext message P is encrypted to ciphertext C by

$$C = P^e \bmod n$$

  - The plaintext is recovered by

$$P = C^d \bmod n$$

  - Because of symmetry in modular arithmetic, encryption and decryption are mutual inverses and commutative. Therefore,

$$P = C^d \bmod n = (P^e)^d \bmod n = (P^d)^e \bmod n$$

- **Key Choice**
  - The encryption key consists of the pair of integers (e, n), and the decryption key is (d, n).
  - The starting point in finding keys for this algorithm is selection of a value for n.
  - The value of n should be quite large, a product of two primes p and q.

- **Key Choice (Cont'd)**
  - Both *p* and *q* should be large themselves.
  - Typically, *p* and *q* are nearly 100 digits each, so*n* is approximately 200 decimal digits (about 512 bits) long; depending on the application, 768, 1024, or more bits may be more appropriate.
  - A large value of *n* effectively inhibits factoring *n* to infer *p* and *q*.

- **Key Choice (Cont'd)**
  - Next, a relatively large integer *e* is chosen so that *e* is relatively prime to (*p* - 1) * (*q* - 1). (Recall that "relatively prime" means that *e* has no factors in common with (*p* - 1) * (*q* - 1).)
  - An easy way to guarantee that *e* is relatively prime to (*p* - 1) * (*q* - 1) is to choose *e* as a prime that is larger than both (*p* - 1) and (*q* - 1).
  - Finally, select *d* such that

$$e * d = 1 \bmod (P - 1) * (q - 1)$$

- **Mathematical Foundations of the RSA Algorithm**
  - The **Euler totient function** $\varphi(n)$ is the number of positive integers less than n that are relatively prime to n. If p is prime, then

$$\varphi(p) = p - 1$$

  - Furthermore, if $n = p * q$, where p and q are both prime, then

$$\varphi(n) = \varphi(p) * \varphi(q) = (p - 1) * (q - 1)$$

- **Mathematical Foundations of the RSA Algorithm (Cont'd)**
  - Euler and Fermat proved that

$$x^{\varphi(n)} \equiv 1 \bmod n$$

for any integer x if n and x are relatively prime.

- **Mathematical Foundations of the RSA Algorithm (Cont'd)**
  - Suppose we encrypt a plaintext message P by the RSA algorithm so that $E(P) = P^e$. We need to be sure we can recover the message.
  - The value e is selected so that we can easily find its inverse d. Because e and d are inverses mod $\varphi(n)$,

$$e * d \equiv 1 \bmod \varphi(n)$$

or

$$e * d = k * \varphi(n) + 1 \quad (*)$$

for some integer k.

- **Mathematical Foundations of the RSA Algorithm (Cont'd)**
  - Because of the Euler-Fermat result, assuming P and p are relatively prime,

$$P^{p-1} \equiv 1 \bmod p$$

and, since $(p-1)$ is a factor of $\varphi(n)$,

$$p^{k*\varphi(n)} \equiv 1 \bmod p$$

  - Multiplying by P produces

$$p^{k*\varphi(n)+1} \equiv p \bmod p$$

- **Mathematical Foundations of the RSA Algorithm (Cont'd)**
  - The same argument holds for q, so

    $$p^{k*\varphi(n)+1} \equiv p \bmod q$$

  - Combining these last two results with (*) produces

    $(P^e)^d = P^{e*d} = P^{k*\varphi(n)+1} = P \bmod p = P \bmod q$

    so that

    $$(P^e)^d \equiv p \bmod n$$

- and e and d are inverse operations.

- **Example**
  - Let $p = 11$ and $q = 13$,

    so that $n = p * q = 143$

    and $\varphi(n) = (p - 1) * (q - 1) = 10 * 12 = 120$.
  - Next, an integer e is needed,

    and e must be relatively prime to $(p - 1) * (q - 1)$.

    Choose $e = 11$.

- **Example (Cont'd)**
  - The inverse of 11 mod 120 is also 11,
    since 11 * 11 = 121 = 1 mod 120.
  - Thus, both encryption and decryption keys are the same:
    $e = d = 11$.
    (For the example, $e = d$ is not a problem, but in a real application you would want to choose values where e is not equal to d.)

- **Example (Cont'd)**
  - Let P be a "message" to be encrypted.
  - For this example we use $P = 7$.
    The message is encrypted as follows:
    7^{11} mod 143 = 106, so that $E(7) = 106$.

- **Example (Cont'd)**
  - This result can be computed fairly easily with the use of a common pocket calculator. $7^{11} = 7^9 * 7^2$. Then $7^9 = 40\ 353\ 607$, but we do not have to work with figures that large.
  - Because of the reducibility rule, $a * b \bmod n = (a \bmod n) * (b \bmod n) \bmod n$. Since we will reduce our final result mod 143, we can reduce any term, such as 7^9, which is 8 mod 143. Then, $8 * 7^2 \bmod 143 = 392 \bmod 143 = 106$.
  - This answer is correct since $D(106) = 106^{11} \bmod 143 = 7$.

- **Use of the Algorithm**
  - The user of the RSA algorithm chooses primes p and q, from which the value $n = p * q$ is obtained.
  - Next e is chosen to be relatively prime to $(p - 1) * (q - 1)$; e is usually a prime larger than $(p - 1)$ or $(q - 1)$.
  - Finally, d is computed as the inverse of e mod $(\varphi(n))$.

- **Use of the Algorithm (Cont'd)**
  - The user distributes *e* and *n* and keeps *d* secret; *p*, *q*, and $\varphi(n)$ may be discarded (but not revealed) at this point.
  - Notice that even though *n* is known to be the product of two primes, if they are relatively large (such as 100 digits long), it will not be feasible to determine the primes *p* and *q* or the private key *d* from *e*.
  - Therefore, this scheme provides adequate security for *d*.

- **Cryptanalysis of the RSA Method**
  - The RSA method has been scrutinized intensely by professionals in computer security and cryptanalysis.
  - Several minor problems have been identified with it, but there have been no serious flaws.

2.8. The Uses of Encryption

- Four applications of encryption:
  1. cryptographic hash functions
  2. key exchange
  3. digital signatures
  4. certificates.

- **Cryptographic Hash Functions**
  - One technique for providing the seal is to compute a cryptographic function, sometimes called a **hash** or **checksum** or **message digest** of the file

- **Cryptographic Hash Functions**
  - The hash function has special characteristics.
    - For instance, some encryptions depend on a function that is easy to understand but difficult to compute.
    - For a simple example, consider the cube function, $y = x^3$. It is relatively easy to compute x^3 by hand, with pencil and paper, or with a calculator. But the inverse function, is much more difficult to compute.
    - Functions like these, which are much easier to compute than their inverses, are called **one-way functions**.
    - Any change to even a single bit will alter the checksum result

- **Cryptographic Hash Functions (Cont'd)**
  - A cryptographic function, such as the DES or AES, is especially appropriate for sealing values, since an outsider will not know the key and thus will not be able to modify the stored value to match with data being modified.
  - In block encryption schemes, **chaining** means linking each block to the previous block's value

- **Cryptographic Hash Functions (Cont'd)**
  - The most widely used cryptographic hash functions are **MD4, MD5** (where MD stands for Message Digest), and **SHA/SHS** (Secure Hash Algorithm or Standard).
  - The MD4/5 algorithms were invented by Ron Rivest and RSA Laboratories. MD5 is an improved version of MD4. Both condense a message of any size to a 128-bit digest.
  - SHA/SHS is similar to both MD4 and MD5; it produces a 160-bit digest

- **Key Exchange**
  - The problem of two previously unknown parties exchanging cryptographic keys is both hard and important. Indeed, the problem is almost circular:

    To establish an encrypted session, you need an encrypted means to exchange keys.

- **Key Exchange (Cont'd)**
  - Let *S* send $E(k_{PUB\text{-}R}, K)$ to *R*.

    Then, only *R* can decrypt *K*.

    Unfortunately, *R* has no assurance that *K* came from *S*.
  - $E(k_{PUB\text{-}R}, E(k_{PRIV\text{-}S}, K))$

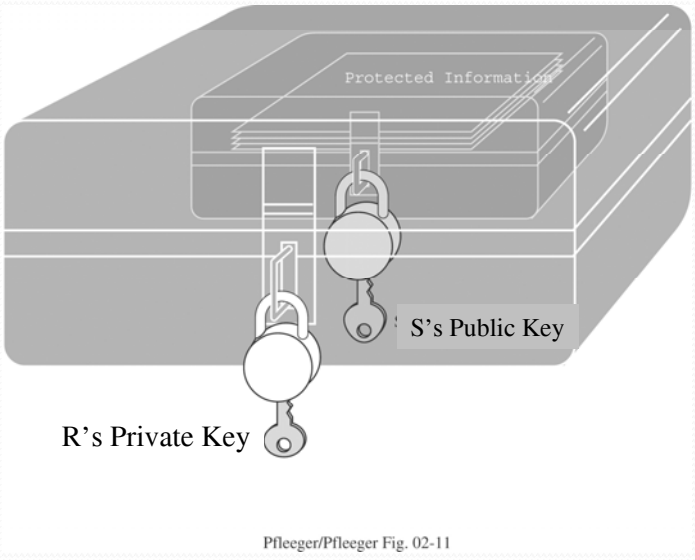
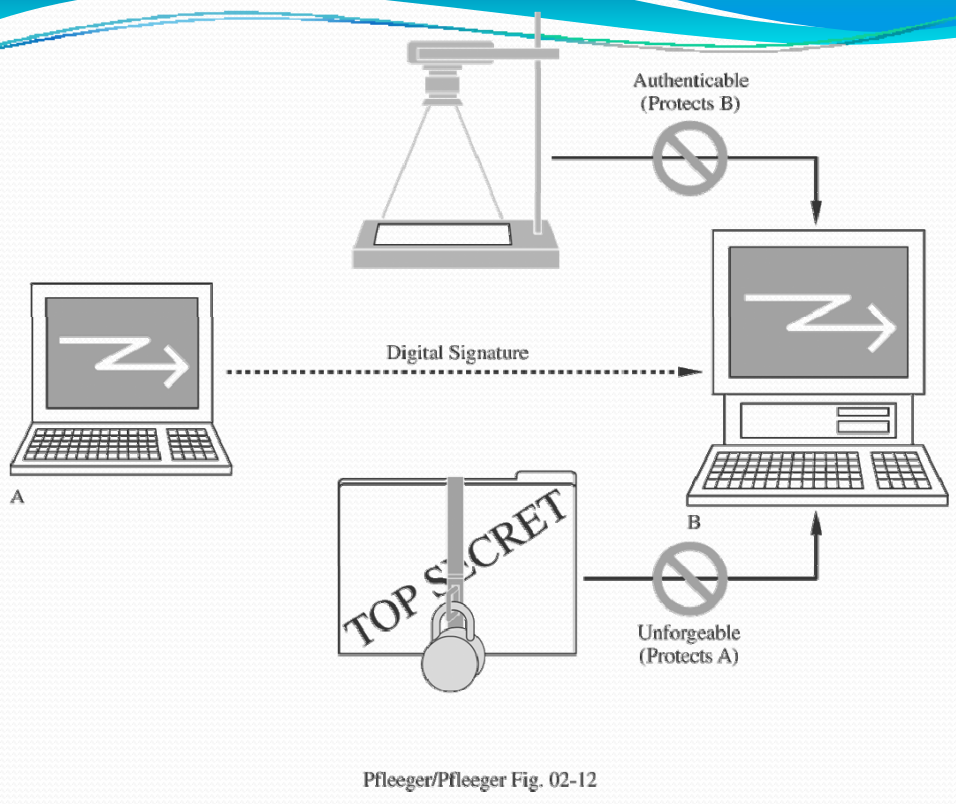


Figure 2-11 The Idea Behind Key Exchange.

- **Digital Signatures**
  - A **digital signature** is a protocol that produces the same effect as a real signature:
    - It is a mark that only the sender can make, but other people can easily recognize as belonging to the sender.
  - Just like a real signature, a digital signature is used to confirm agreement to a message.

- **Digital Signatures (Cont'd)**
  - **Properties**
    - *It must be* ***unforgeable****.*
      - If person *P* signs message *M* with signature *S*(*P*,*M*), it is impossible for anyone else to produce the pair [*M*, *S*(*P*,*M*)].
    - *It must be* ***authentic****.*
      - If a person *R* receives the pair [*M*, *S*(*P*,*M*)] purportedly from *P*, *R* can check that the signature is really from *P*.
      - Only *P* could have created this signature, and the signature is firmly attached to *M*.

- **Digital Signatures (Cont'd)**
  - Two more properties are desirable for transactions completed with the aid of digital signatures:
    - *It is* ***not alterable****.* After being transmitted, *M* cannot be changed by *S*, *R*, or an interceptor.
    - *It is* ***not reusable****.* A previous message presented again will be instantly detected by *R*.



Pfleeger/Pfleeger Fig. 02-12

Figure 2-12 Requirements for a Digital Signature.

- **Public Key Protocol**
  - Public key encryption systems are ideally suited to digital signatures.
  - For simple notation, let us assume that the public key encryption for user *U* is accessed through $E(M, K_U)$ and that the private key transformation for *U* is written as $D(M,K_U)$.
  - We can think of *E* as the ***privacy*** transformation (since only *U* can decrypt it) and *D* as the ***authenticity*** transformation (since only *U* can produce it).

- **Public Key Protocol (Cont'd)**
  - Some asymmetric algorithms such as RSA, *D* and *E* are commutative, and either one can be applied to any message. Thus,

$$D(E(M,),) = M = E(D(M,),)$$

- **Public Key Protocol (Cont'd)**
  - If *S* wishes to send *M* to *R*, *S* uses the authenticity transformation to produce $D(M, K_S)$.
  - *S* then sends $D(M, K_S)$ to *R*. *R* decodes the message with the public key transformation of *S*, computing $E(D(M,K_S), K_S) = M$.
  - Since only *S* can create a message that makes sense under $E(,K_S)$, the message must genuinely have come from *S*.
  - This test satisfies the **authenticity requirement**.

- **Public Key Protocol (Cont'd)**
  - *R* will save $D(M,K_S)$.
  - If *S* should later allege that the message is a forgery (not really from *S*), *R* can simply show *M* and $D(M,K_S)$.
  - Anyone can verify that since $D(M,K_S)$ is transformed to *M* with the public key transformation of *S* but only *S* could have produced $D(M,K_S)$ then $D(M,K_S)$ must be from *S*.
  - This test satisfies the **unforgeable requirement**.

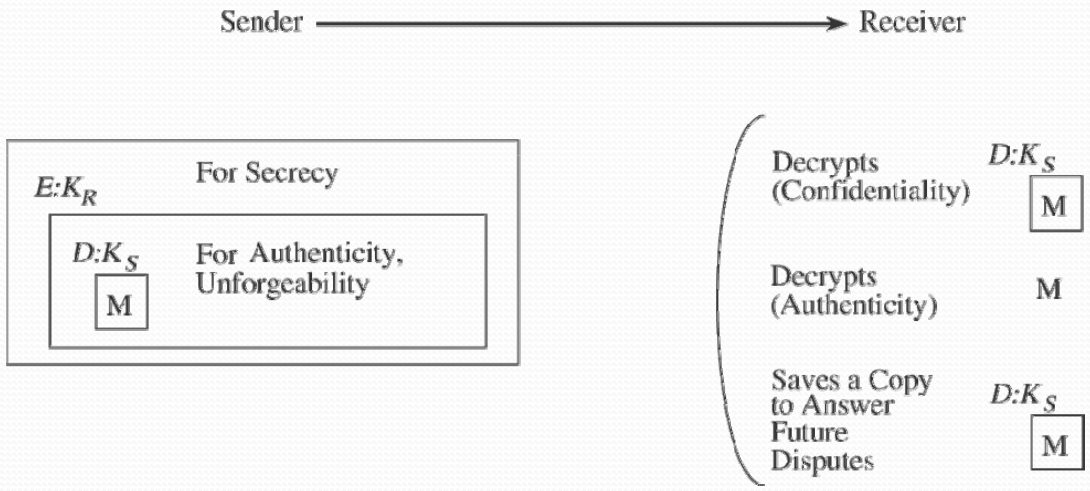
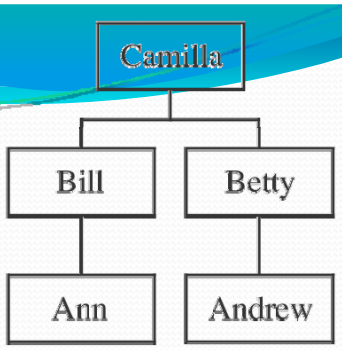


Figure 2-13 Use of Two Keys in Asymmetric Digital Signature.

- **Certificates**
  - For electronic communication to succeed, we must develop ways for two parties to establish trust without having met
  - The concept of "vouching for" by a third party can be a basis for trust in commercial settings where two parties do not know each other.



- Certificates (Cont'd)
  - Trust Through a Common Respected Individual
    - The chain of verification might be something like this:
      - Ann asks Bill who Andrew is.
      - Bill either asks Betty if he knows her directly or if not, asks Camilla.
      - Camilla asks Betty.
      - Betty replies that Andrew works for her.
      - Camilla tells Bill.
      - Bill tells Ann.

- Certificates (Cont'd)
  - Trust Through a Common Respected Individual (Cont'd)
    - We can use a similar process for cryptographic key exchange
    - If Andrew and Ann want to communicate, Andrew can give his public key to Betty, who passes it to Camilla or directly to Bill, who gives it to Ann
    - This protocol is a way of obtaining authenticated public keys, a binding of a key, and a reliable identity.

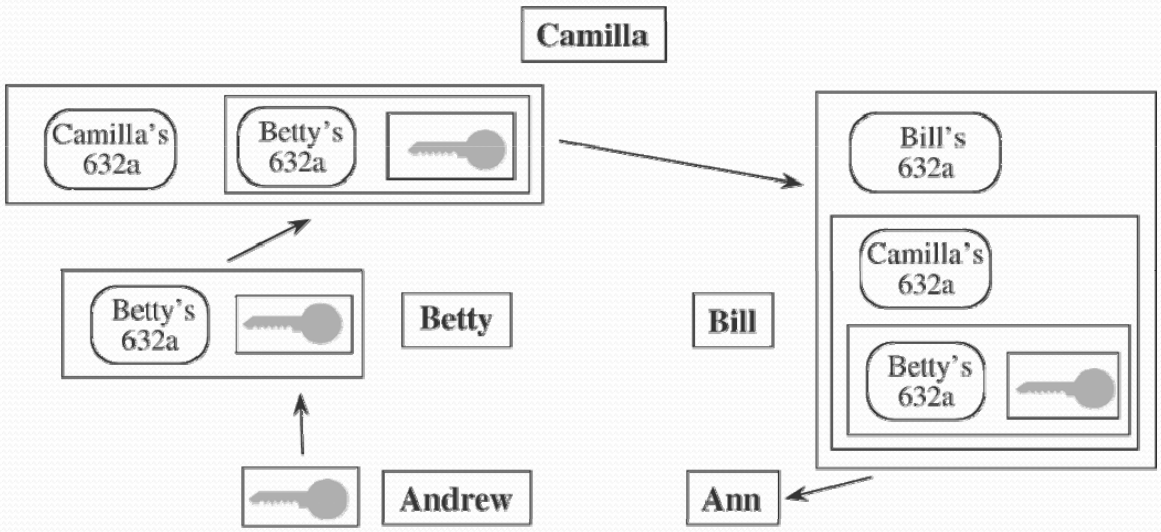


Figure 2-15 Andrew Passes a Key to Ann.

- Certificates (Cont'd)
  - Certificates to Authenticate an Identity
    - A public key and user's identity are bound together in a **certificate**, which is then signed by someone called a **certificate authority**, certifying the accuracy of the binding.

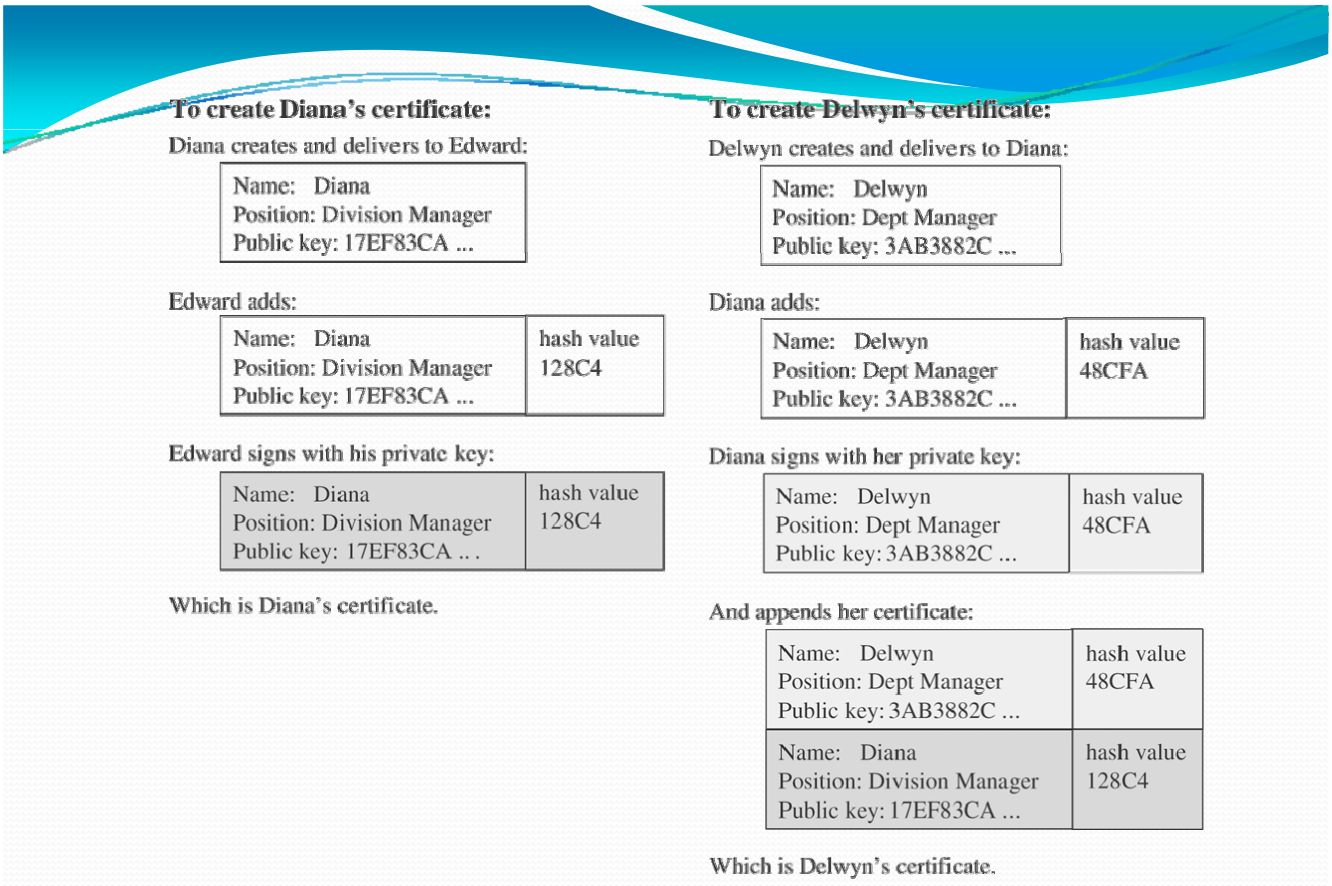


Figure 2-17 Signed Certificates.

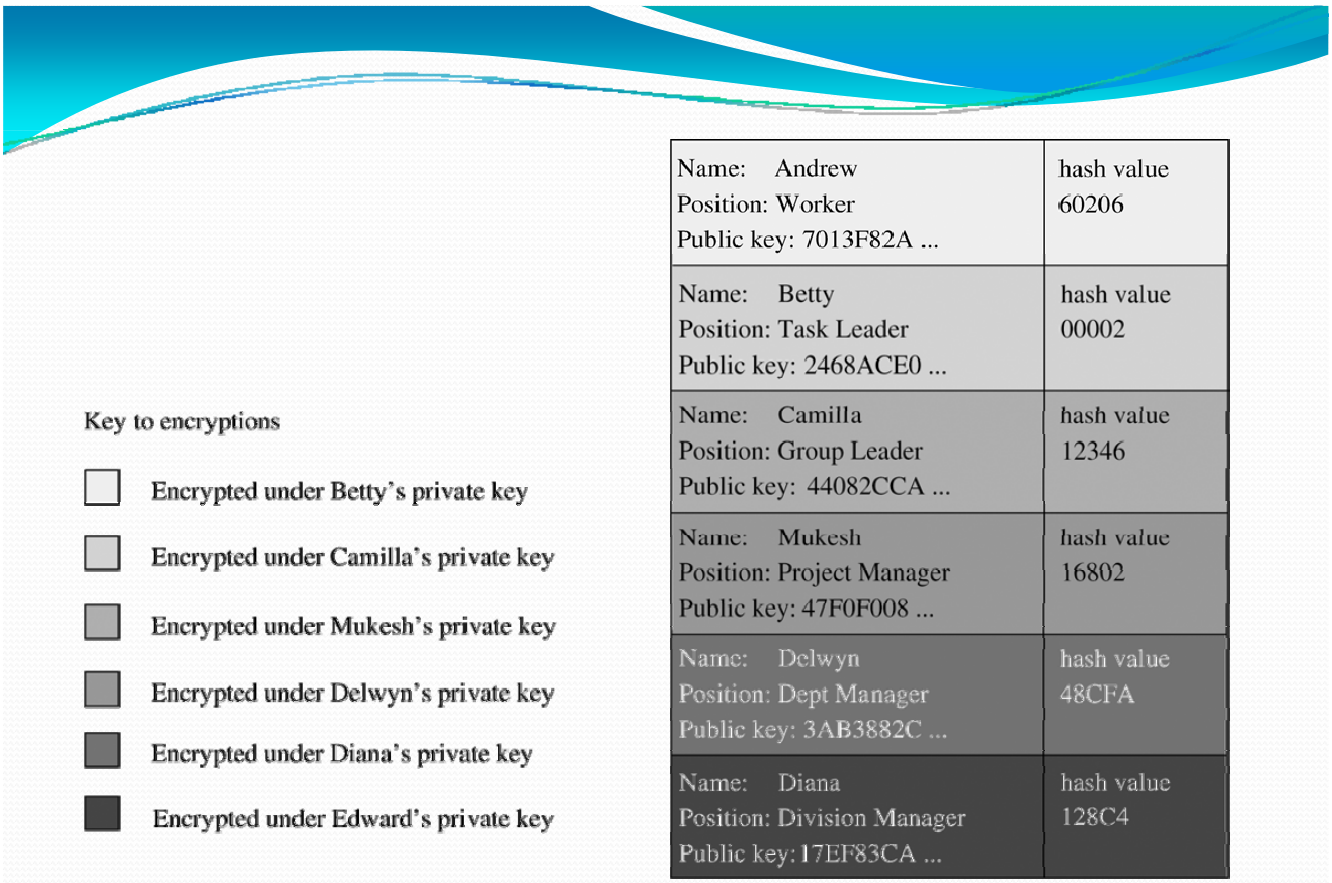


Figure 2-18 Chain of Certificates.

- Certificates (Cont'd)
  - Trust Without a Single Hierarchy
    - it is not necessary to have such a structure or to follow it to use certificate signing for authentication.
    - Anyone who is considered acceptable as an authority can sign a certificate.
    -

2.9. Summary of Encryption

- The basic processes of encryption and cryptanalysis
- Introduce the two basic methods of encipherment substitution and transposition or permutation as well as techniques of cryptanalysis
- "Real" cryptosystems: DES, AES, and RSA
- Several very important and widely used applications of cryptography: hash functions, key exchange protocols, digital signatures, and certificates