Entropy, $H(X)$, is defined as the average amount of information per message. It can be considered a measure of how much *choice* is involved in the selection of a message $X$. It is expressed by the following summation over all possible messages:

$$H(X) = -\sum_X P(X) \log_2 P(X) = \sum_X P(X) \log_2 \frac{1}{P(X)} \tag{14.5}$$

When the logarithm is taken to the base 2, as shown, $H(X)$ is the *expected number of bits* in an *optimally encoded* message $X$. This is not quite the measure that a cryptanalyst desires. He will have intercepted some ciphertext and will want to know how confidently he can predict a message (or key) given that this particular ciphertext was sent. *Equivocation,* defined as the conditional entropy of $X$ given $Y$, is a more useful measure for the cryptanalyst in attempting to break the cipher and is given by

$$H(X|Y) = -\sum_{X,Y} P(X,Y) \log_2 P(X|Y) \tag{14.6}$$

$$= \sum_Y P(Y) \sum_X P(X|Y) \log_2 \frac{1}{P(X|Y)}$$

Equivocation can be thought of as the uncertainty that message $X$ was sent, having received $Y$. The cryptanalyst would like $H(X|Y)$ to approach zero as the amount of intercepted ciphertext, $Y$, increases.

**Example 14.3 Entropy and Equivocation**

Consider a sample message set consisting of eight equally likely messages $\{X\} = X_1, X_2, \ldots, X8$.

(a) Find the entropy associated with a message from the set $\{X\}$.
(b) Given another equally likely message set $\{Y\} = Y_1, Y_2$. Consider that the occurrence of each message $Y$ narrows the possible choices of X in the following way:

If $Y_1$ is present: only $X_1, X_2, X_3$, or $X_4$ is possible

If $Y_2$ is present: only $X_5, X_6, X_7$, or $X_8$ is possible

Find the equivocation of message $X$ conditioned on message $Y$.

*Solution*

(a) $P(X) = \frac{1}{8}$
$H(X) = 8[(\frac{1}{8}) \log_2 8] = 3$ bits/message

(b) $P(Y) = \frac{1}{2}$. For each $Y$, $P(X|Y) = \frac{1}{4}$ for four of the $X$'s and $P(X|Y) = 0$ for the remaining four $X$'s. Using Equation (14.6), we obtain

$$H(X|Y) = 2[(\tfrac{1}{2})4(\tfrac{1}{4} \log_2 4)] = 2 \text{ bits/message}$$

We see that knowledge of $Y$ has reduced the uncertainty of $X$ from 3 bits/message to 2 bits/message.

### 14.2.3 Rate of a Language and Redundancy

The *true rate* of a language is defined as the average number of *information bits* contained in each character and is expressed for messages of length $N$ by

$$r = \frac{H(X)}{N} \tag{14.7}$$

where $H(X)$ is the message entropy, or the number of bits in the *optimally encoded* message. For large $N$, estimates of $r$ for written English range between 1.0 and 1.5 bits/character [4]. The *absolute rate* or maximum entropy of a language is defined as the maximum number of information bits contained in each character assuming that all possible sequences of characters are equally likely. The absolute rate is given by

$$r' = \log_2 L \tag{14.8}$$

where $L$ is the number of characters in the language. For the English alphabet $r' = \log_2 26 = 4.7$ bits/character. The true rate of English is, or course, much less than its absolute rate since, like most languages, English is highly redundant and structured.

The *redundancy* of a language is defined in terms of its true rate and absolute rate as

$$D = r' - r \tag{14.9}$$

For the English language with $r' = 4.7$ bits/character and $r = 1.5$ bits/character, $D = 3.2$, and the ratio $D/r' = 0.68$ is a measure of the redundancy in the language.

### 14.2.4 Unicity Distance and Ideal Secrecy

We stated earlier that perfect secrecy requires an infinite amount of key if we allow messages of unlimited length. With a finite key size, the equivocation of the key $H(K|C)$ generally approaches zero, implying that the key can be uniquely determined and the cipher system can be broken. The *unicity distance* is defined as the smallest amount of ciphertext, $N$, such that they key equivocation $H(K|C)$ is close to zero. Therefore, the unicity distance is the amount of ciphertext needed to uniquely determine the key and thus break the cipher system. Shannon [5] described an *ideal secrecy* system as one in which $H(K|C)$ does not approach zero as the amount of ciphertext approaches infinity; that is, no matter how much ciphertext is intercepted, the key cannot be determined. The term "ideal secrecy" describes a system that does not achieve perfect secrecy but is nonetheless unbreakable (unconditionally secure) because it does not reveal enough information to determine the key.

Most cipher systems are too complex to determine the probabilities required to derive the unicity distance. However, it is sometimes possible to approximate unicity distance, as shown by Shannon [5] and Hellman [6]. Following Hellman, assume that each plaintext and ciphertext message comes from a finite alphabet of $L$ symbols.

Thus there are $2^{r'N}$ possible mes ges of length, $N$, where $r'$ is the absolute rate of the language. We can consider the total message space partitioned into two classes, meaningful messages, $M_1$, and meaningless messages $M_2$. We then have

$$\text{number of meaningful messages} = 2^{rN} \tag{14.10}$$

$$\text{number of meaningless messages} = 2^{r'N} - 2^{rN} \tag{14.11}$$

where $r$ is the true rate of the language, and where the a priori probabilities of the message classes are

$$P(M_1) = \frac{1}{2^{rN}} = 2^{-rN} \qquad M_1 \text{ meaningful} \tag{14.12}$$

$$P(M_2) = 0 \qquad M_2 \text{ meaningless} \tag{14.13}$$

Let us assume that there are $2^{H(K)}$ possible keys (size of the key alphabet), where $H(K)$ is the entropy of the key (number of bits in the key). Assume that all keys are equally likely; that is,

$$P(K) = \frac{1}{2^{H(K)}} = 2^{-H(K)} \tag{14.14}$$

The derivation of the unicity distance is based on a *random cipher* model, which states that for each key $K$ and ciphertext $C$, the decryption operation $D_K(C)$ yields an independent random variable distributed over all the possible $2^{r'N}$ messages (both meaningful and meaningless). Therefore, for a given $K$ and $C$, the $D_K(C)$ operation can produce any one of the plaintext messages with equal probability.

Given an encryption described by $C_i = E_{K_i}(M_i)$, a *false solution* $F$ arises whenever encryption under another key $K_j$ could also produce $C_i$ either from the message $M_i$ or from some other message $M_j$; that is,

$$C_i = E_{K_i}(M_i) = E_{K_j}(M_i) = E_{K_j}(M_j) \tag{14.15}$$

A cryptanalyst intercepting $C_i$ would not be able to pick the correct key and hence could not break the cipher system. We are not concerned with the decryption operations that produce *meaningless* messages because these are easily rejected.

For every correct solution to a particular ciphertext there are $2^{H(K)} - 1$ incorrect keys, each of which has the same probability $P(F)$ of yielding a false solution. Because each meaningful plaintext message is assumed equally likely, the probability of a false solution, is the same as the probability of getting a meaningful message, namely,

$$P(F) = \frac{2^{rN}}{2^{r'N}} = 2^{(r-r')N} = 2^{-DN} \tag{14.16}$$

where $D = r' - r$ is the redundancy of the language. The expected number of false solutions $\bar{F}$ is then

$$\bar{F} = [2^{H(K)} - 1]P(F) = [2^{H(K)} - 1]2^{-DN} \approx 2^{H(K)-DN} \tag{14.17}$$

Thus there are $2^{r'N}$ possible mes ges of length, $N$, where $r'$ is the absolute rate of the language. We can consider the total message space partitioned into two classes, meaningful messages, $M_1$, and meaningless messages $M_2$. We then have

$$\text{number of meaningful messages} = 2^{rN} \tag{14.10}$$

$$\text{number of meaningless messages} = 2^{r'N} - 2^{rN} \tag{14.11}$$

where $r$ is the true rate of the language, and where the a priori probabilities of the message classes are

$$P(M_1) = \frac{1}{2^{rN}} = 2^{-rN} \qquad M_1 \text{ meaningful} \tag{14.12}$$

$$P(M_2) = 0 \qquad M_2 \text{ meaningless} \tag{14.13}$$

Let us assume that there are $2^{H(K)}$ possible keys (size of the key alphabet), where $H(K)$ is the entropy of the key (number of bits in the key). Assume that all keys are equally likely; that is,

$$P(K) = \frac{1}{2^{H(K)}} = 2^{-H(K)} \tag{14.14}$$

The derivation of the unicity distance is based on a *random cipher* model, which states that for each key $K$ and ciphertext $C$, the decryption operation $D_K(C)$ yields an independent random variable distributed over all the possible $2^{r'N}$ messages (both meaningful and meaningless). Therefore, for a given $K$ and $C$, the $D_K(C)$ operation can produce any one of the plaintext messages with equal probability.

Given an encryption described by $C_i = E_{K_i}(M_i)$, a *false solution* $F$ arises whenever encryption under another key $K_j$ could also produce $C_i$ either from the message $M_i$ or from some other message $M_j$; that is,

$$C_i = E_{K_i}(M_i) = E_{K_j}(M_i) = E_{K_j}(M_j) \tag{14.15}$$

A cryptanalyst intercepting $C_i$ would not be able to pick the correct key and hence could not break the cipher system. We are not concerned with the decryption operations that produce *meaningless* messages because these are easily rejected.

For every correct solution to a particular ciphertext there are $2^{H(K)} - 1$ incorrect keys, each of which has the same probability $P(F)$ of yielding a false solution. Because each meaningful plaintext message is assumed equally likely, the probability of a false solution, is the same as the probability of getting a meaningful message, namely,

$$P(F) = \frac{2^{rN}}{2^{r'N}} = 2^{(r-r')N} = 2^{-DN} \tag{14.16}$$

where $D = r' - r$ is the redundancy of the language. The expected number of false solutions $\bar{F}$ is then

$$\bar{F} = [2^{H(K)} - 1]P(F) = [2^{H(K)} - 1]2^{-DN} \approx 2^{H(K)-DN} \tag{14.17}$$

Because of the rapid decrease of $\bar{F}$ with increasing $N$,

$$\log_2 \bar{F} = H(K) - DN = 0 \tag{14.18}$$

is defined as the point where the number of false solutions is sufficiently small so that the cipher can be broken. The resulting unicity distance is therefore

$$N = \frac{H(K)}{D} \tag{14.19}$$

We can see from Equation (14.17) that if $H(K)$ is much larger than $DN$, there will be a large number of meaningful decryptions, and thus a small likelihood of a cryptanalyst distinguishing which meaningful message is the correct message. In a loose sense, $DN$ represents the number of equations available for solving for the key, and $H(K)$ the number of unknowns. When the number of equations is smaller than the number of unknown key bits, a unique solution is not possible and the system is said to be unbreakable. When the number of equations is larger than the number of unknowns, a unique solution is possible and the system can no longer be characterized as unbreakable (although it may still be computationally secure).

It is the predominance of meaningless decryptions that enables cryptograms to be broken. Equation (14.19) indicates the value of using *data compression* techniques prior to encryption. Data compression removes redundancy, thereby increasing the unicity distance. Perfect data compression would result in $D = 0$ and $N = \infty$ for any key size.

**Example 14.4 Unicity Distance**

Calculate the unicity distance for a written English encryption system, where the key is given by the sequence $k_1, k_2, \ldots, k_{29}$, where each $k_i$ is a random integer in the range (1, 25) dictating the shift number (Figure 14.3) for the $i$th character. Assume that each of the possible key sequences is equally likely.

*Solution*

There are $(25)^{29}$ possible key sequences, each of which is equally likely. Therefore, using Equations (14.5), (14.8), and (14.19) we have

$$\text{Key entropy:} \quad H(K) = \log_2 (25)^{29} = 135 \text{ bits}$$

$$\text{Absolute rate for English:} \quad r' = \log_2 26 = 4.7 \text{ bits/character}$$

$$\text{Assumed true rate for English:} \quad r = 1.5 \text{ bits/character}$$

$$\text{Redundancy:} \quad D = r' - r = 3.2 \text{ bits/character}$$

$$N = \frac{H(K)}{D} = \frac{135}{3.2} \approx 43 \text{ characters}$$

In Example 14.2, perfect secrecy was illustrated using the same type of key sequence described here, with a 29-character message. In this example we see that if the available ciphertext is 43 characters long (which implies that some portion of the key sequence must be used twice), a unique solution may be possible. However, there is

no indication as to the computational difficulty in finding the solution. Even though we have estimated the theoretical amount of ciphertext required to break the cipher, it might be computationally infeasible to accomplish this.

## 14.3 PRACTICAL SECURITY

For ciphertext sequences greater than the unicity distance any system can be solved, in principle, merely by trying each possible key until the unique solution is obtained. This is completely impractical, however, except when the key is extremely small. For example, for a key configured as a permutation of the alphabet, there are $26! \approx 4 \times 10^{26}$ possibilities (considered small in the cryptographic context). In an exhaustive search, one might expect to reach the right key at about halfway through the search. If we assume that each trial requires a computation time of 1 μs, the total search time exceeds $10^{12}$ years. Hence techniques other than a brute-force search (e.g., statistical analysis) must be employed if a cryptanalyst is to have any hope of success.

### 14.3.1 Confusion and Diffusion

A statistical analysis using the frequency of occurrence of individual characters and character combinations can be used to solve many cipher systems. Shannon [5] suggested two encryption concepts for frustrating the statistical endeavors of the cryptanalyst. He termed these encryption transformations confusion and diffusion. *Confusion* involves substitutions that render the final relationship between the key and ciphertext as complex as possible. This makes it difficult to utilize a statistical analysis to narrow the search to a particular subset of the key variable space. Confusion ensures that the majority of the key is needed to decrypt even very short sequences of ciphertext. *Diffusion* involves transformations that smooth out the statistical differences between characters and between character combinations. An example of diffusion with a 26-letter alphabet is to transform a message sequence $M = M_0, M_1, \ldots$ into a new message sequence $Y = Y_0, Y_1, \ldots$ according to the relationship

$$Y_n = \sum_{i=0}^{s-1} M_{n+i} \quad \text{modulo-26} \tag{14.20}$$

where each character in the sequence is regarded as an integer modulo-26, $s$ is some chosen integer, and $n = 1, 2, \ldots$. The new message, $Y$, will have the same redundancy as the original message, $M$, but the letter frequencies of $Y$ will be more uniform than in $M$. The effect is that the cryptanalyst needs to intercept a longer sequence of ciphertext before any statistical analysis can be useful.

### 14.3.2 Substitution

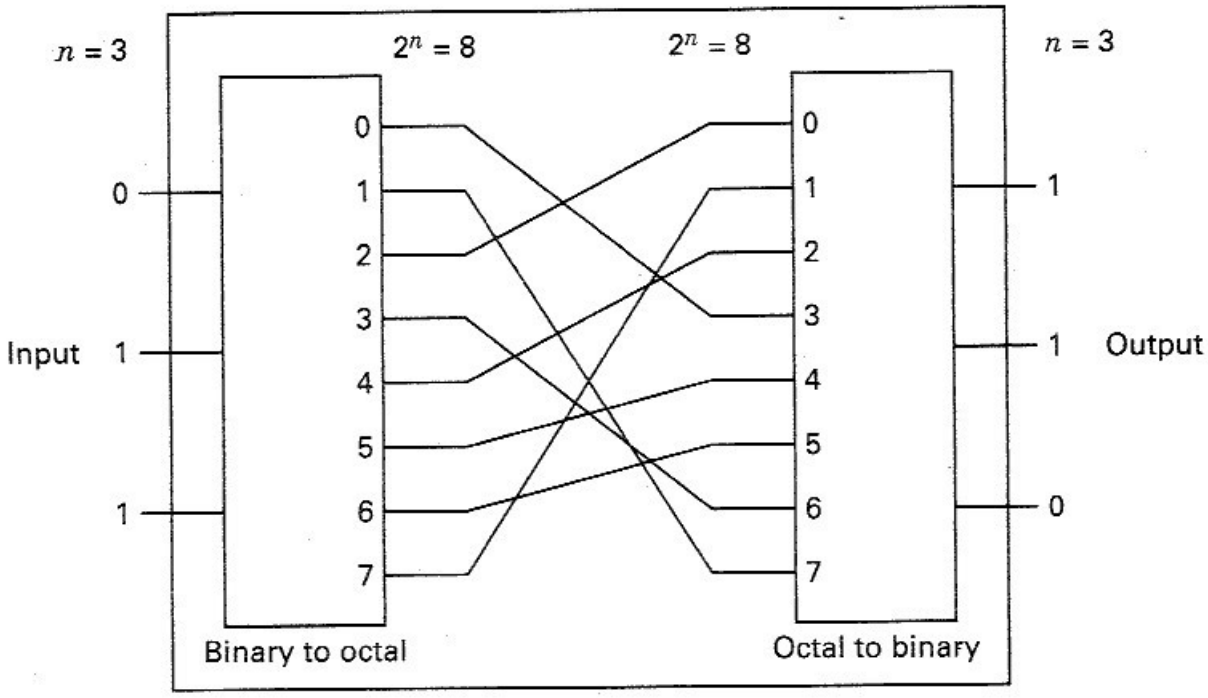
Substitution encryption techniques, such as the Caesar cipher and the Trithemius progressive key cipher, are widely used in puzzles. Such simple substitution ciphers offer little encryption protection. For a substitution technique to fulfill Shannon's

concept of *confusion*, a more complex relationship is required. Figure 14.6 shows one example of providing greater substitution complexity through the use of a nonlinear transformation. In general, $n$ input bits are first represented as one of $2^n$ different characters (binary-to-octal transformation in the example of Figure 14.6. The set of $2^n$ characters are then permuted so that each character is transposed to one of the others in the set. The character is then converted back to an $n$-bit output.

It can be easily shown that there are $(2^n)!$ different substitution or connection patterns possible. The cryptanalyst's task becomes computationally unfeasible as $n$ gets large, say $n = 128$; then $2^n = 10^{38}$, and $(2^n)!$ is an astronomical number. We recognize that for $n = 128$, this substitution box ($S$-box) transformation is complex (confusion). However, although we can identify the $S$-box with $n = 128$ as ideal, its implementation is not feasible because it would require a unit with $2^n = 10^{38}$ wiring connections.

To verify that the $S$-box example in Figure 14.6 performs a *nonlinear transformation*, we need only use the superposition theorem stated below as a test. Let

$$C = Ta + Tb$$
$$C' = T(a + b) \tag{14.21}$$




| Input | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| Output | 011 | 111 | 000 | 110 | 010 | 100 | 101 | 001 |

**Figure 14.6** Substitution box.

where $a$ and $b$ are input terms, $C$ and $C'$ are output terms, and $T$ is the transformation. Then

$$\text{If } T \text{ is linear: } C = C' \text{ for all inputs}$$

$$\text{If } T \text{ is nonlinear: } C \neq C'$$

Suppose that $a = 001$ and $b = 010$; then, using $T$ as described in Figure 12.6, we obtain

$$C = T(001) \oplus T(010) = 111 \oplus 000 = 111$$

$$C' = T(001 \oplus 010) = T(011) = 110$$

where the symbol $\oplus$ represents modulo-2 addition. Since $C \neq C'$, the $S$-box is nonlinear.

### 14.3.3 Permutation

In permutation (transposition), the positions of the plaintext letters in the message are simply rearranged, rather than being substituted with other letters of the alphabet as in the classic ciphers. For example, the word THINK might appear, after permutation, as the ciphertext HKTNI. Figure 14.7 represents an example of binary data permutation (a linear operation). Here we see that the input data are simply rearranged or permuted (P-box). The technique has one major disadvantage when used alone; it is vulnerable to trick messages. A trick message is

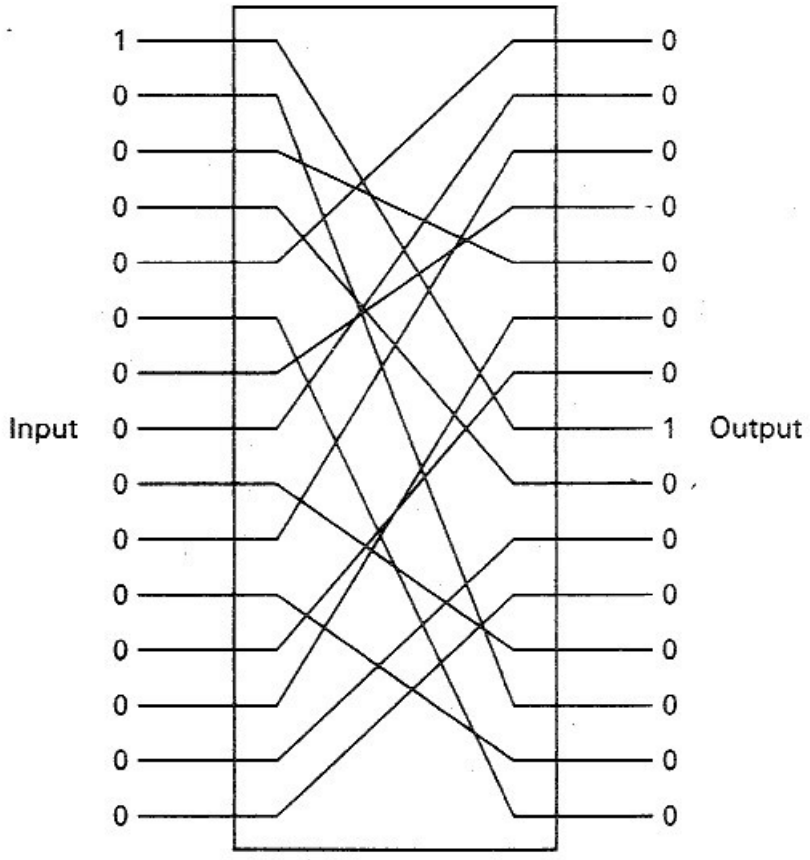



**Figure 14.7** Permutation box.

illustrated in Figure 14.7. A single 1 at the input and all the rest 0 quickly reveals one of the internal connections. If the cryptanalyst can subject the system to a plaintext attack, he will transmit a sequence of such trick messages, moving the single 1 one position for each transmission. In this way, each of the connections from input to output is revealed. This is an example of why a system's security should not depend on its architecture.

### 14.3.4 Product Cipher System

For transformation involving reasonable numbers of $n$-message symbols, both of the foregoing cipher systems (the $S$-box and the $P$-box) are by themselves wanting. Shannon [5] suggested using a *product cipher* or a combination of $S$-box and $P$-box transformations, which together could yield a cipher system more powerful than either one alone. This approach of alternately applying substitution and permutation transformations has been used by IBM in the LUCIFER system [7, 8], and has become the basis for the national Data Encryption Standard (DES) [9]. Figure 14.8 illustrates such a combination of $P$-boxes and $S$-boxes. Decryption is accomplished by running the data backward, using the inverse of each $S$-box. The system as pictured in Figure 14.8 is difficult to implement since each $S$-box is different, a randomly generated key is not usable, and the system does not lend itself to repeated use of the same circuitry. To avoid these difficulties, the LUCIFER system [8] used two different types of $S$-boxes, $S_1$ and $S_0$, which could be publicly revealed. Figure 14.9 illustrates such a system. The input data are transformed by the sequence of $S$-boxes and $P$-boxes under the dictates of a key. The 25-bit key in this example designates, with a binary one or zero, the choice ($S_1$ or $S_0$) of each of the 25 $S$-boxes

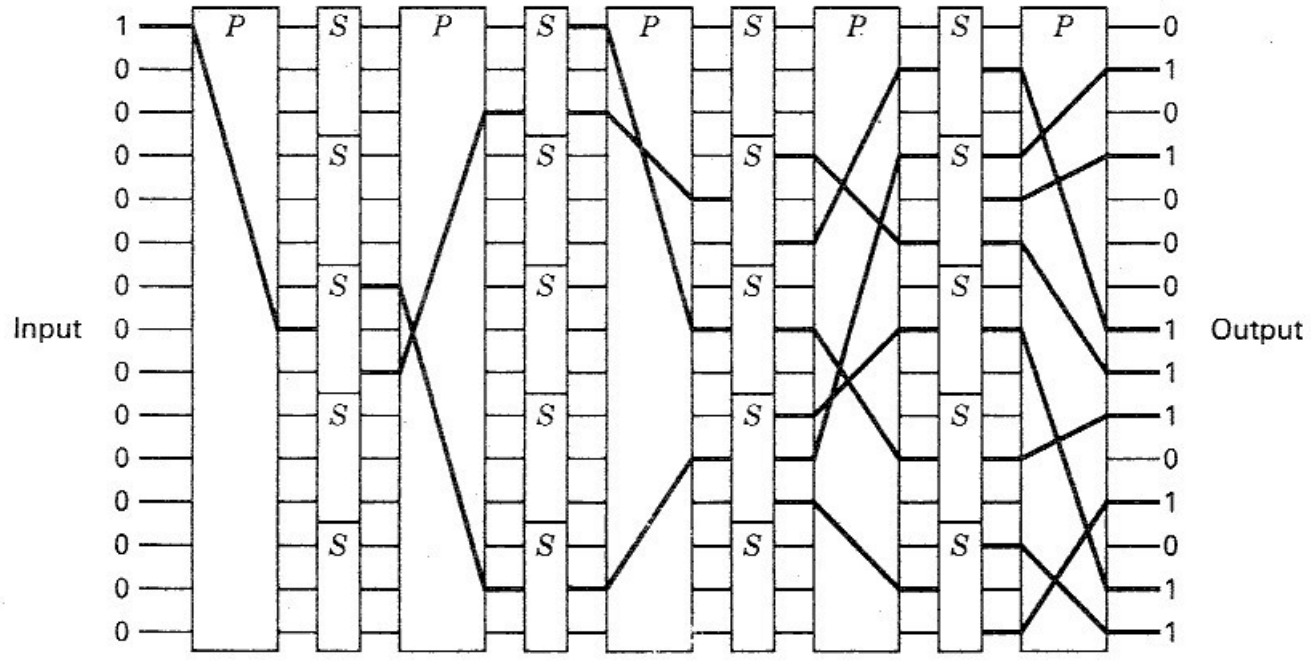



**Figure 14.8** Product cipher system.

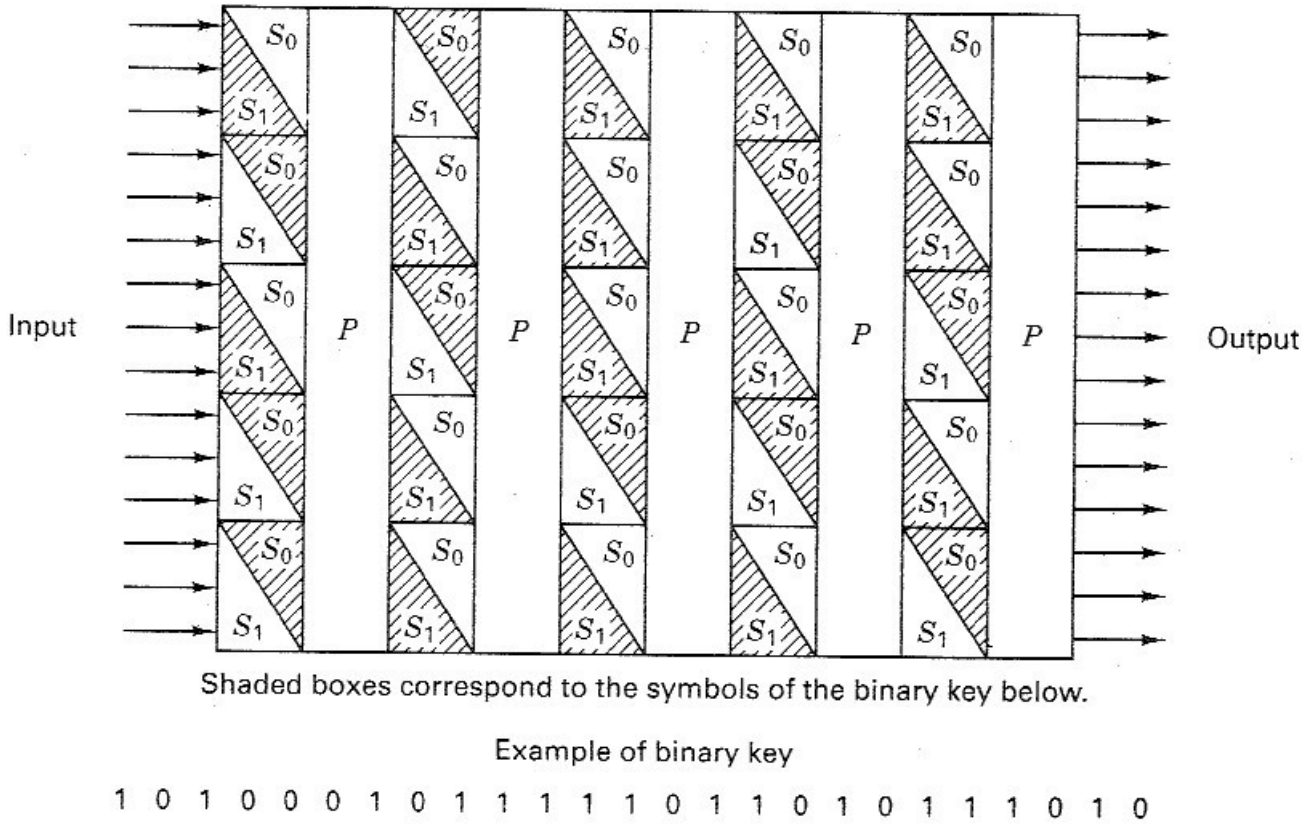



**Figure 14.9** Individual keying capability.

in the block. The details of the encryption devices can be revealed since security of the system is provided by the key.

The iterated structure of the product cipher system in Figure 14.9 is typical of most present-day block ciphers. The messages are partitioned into successive blocks of $n$ bits, each of which is encrypted with the same key. The $n$-bit block represents one of $2^n$ different characters, allowing for $(2^n)!$ different substitution patterns. Consequently, for a reasonable implementation, the substitution part of the encryption scheme is performed in parallel on small segments of the block. An example of this is seen in the next section.

### 14.3.5 The Data Encryption Standard

In 1977, the National Bureau of Standards adopted a modified Lucifer system as the national Data Encryption Standard (DES) [9]. From a system input-output point of view, DES can be regarded as a block encryption system with an alphabet size of $2^{64}$ symbols, as shown in Figure 14.10. An input block of 64 bits, regarded as a plaintext symbol in this alphabet, is replaced with a new ciphertext symbol. Figure 14.11 illustrates the system functions in block diagram form. The encryption algorithm starts with an initial permutation (IP) of the 64 plaintext bits, described in the IP-table (Table 14.1). The IP-table is read from left to right and from top to bottom, so that bits $x_1, x_2, \ldots, x_{64}$ are permuted to $x_{58}, x_{50}, \ldots, x_7$. After this initial permutation, the heart of the encryption algorithm consists of 16 iterations using

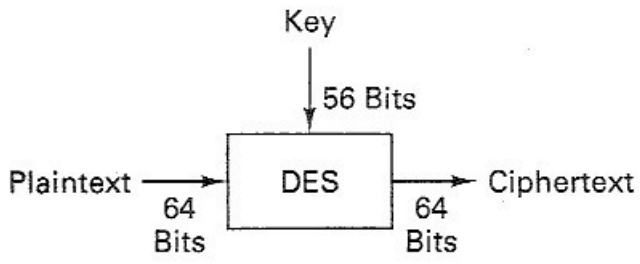



**Figure 14.10** Data encryption standard (DES) viewed as a block encryption system.

the standard building block (SBB) shown in Figure 14.12. The standard building block uses 48 bits of key to transform the 64 input data bits into 64 output data bits, designated as 32 left-half bits and 32 right-half bits. The output of each building block becomes the input to the next building block. The input right-half 32 bits ($R_{i-1}$) are copied unchanged to become the output left-half 32 bits ($L_i$). The $R_{i-1}$ bits are also *extended* and transformed into 48 bits with the $E$-table (Table 14.2), and then modulo-2 summed with the 48 bits of the key. As in the case of the IP-table, the $E$-table is read from left to right and from top to bottom. The table expands bits

$$R_{i-1} = x_1, x_2, \ldots, x_{32}$$

into

$$(R_{i-1})_E = x_{32}, x_1, x_2, \ldots, x_{32}, x_1 \tag{14.22}$$

Notice that the bits listed in the first and last columns of the $E$-table are those bit positions that are used twice to provide the 32 bit-to-48 bit expansion.

Next, $(R_{i-1})_E$ is modulo-2 summed with the $i$th key selection, explained later, and the result is segmented into eight 6-bit blocks

$$B_1, B_2, \ldots, B_8$$

That is,

$$(R_{i-1})_E \oplus K_i = B_1, B_2, \cdots, B_8 \tag{14.23}$$

Each of the eight 6-bit blocks, $B_j$, is then used as an input to an $S$-box function which returns a 4-bit block, $S_j(B_j)$. Thus the input 48 bits are transformed by the $S$-box to 32 bits. The $S$-box mapping function, $S_j$, is defined in Table 14.3. The transformation of $B_j = b_1, b_2, b_3, b_4, b_5, b_6$ is accomplished as follows. The integer corresponding to bits, $b_1, b_6$ selects a row in the table, and the integer corresponding to bits $b_2\, b_3\, b_4\, b_5$ selects a column in the table. For example, if $b_1 = 110001$, then $S_1$ returns the value in row 3, column 8, which is the integer 5 and is represented by the bit sequence 0101. The resulting 32-bit block out of the $S$-box is then permuted using the $P$-table (Table 14.4). As in the case of the other tables, the $P$-table is read from left to right and from top to bottom, so that bits $x_1, x_2, \ldots, x_{32}$ are permuted to $x_{16}, x_7, \ldots, x_{25}$. The 32-bit output of the $P$-table is modulo-2 summed with the input left-half 32 bits ($L_{i-1}$), forming the output right-half 32 bits ($R_i$).

The algorithm of the standard building block can be represented by