CPSC 467: Cryptography and Security

Michael J. Fischer

Lecture 8
September 24, 2020

Symmetric Cryptosystem Components

Building blocks

Symmetric (one-key) ciphers combine simple ideas, some of which we've already seen:

- Substitution
- Transposition
- Composition
- Subkey generation
- Chaining

Substitution: Replacing one letter by another

The methods discussed so far are based on letter substitution.

The Caesar cipher *shifts* the alphabet cyclically.
This yields 26 possible permutations of the alphabet.

In general, one can use any permutation of the alphabet, as long as we have a way of computing the permutation and its inverse.
This gives us 26! possible permutations.

Often, permutations are specified by a table called an *S-box*.

Transposition: Rearranging letters

Another technique is to rearrange the letters of the plaintext.

`this message is encoded with a transposition cipher`

1. Pick a number: 9.
2. Write the message in a 9-column matrix (ignoring spaces):

```
thi sme ssa
gei sen cod
edw ith atr
ans pos iti
onc iph er
```

3. Read it out by columns[1]

`tgeao hednn iiwsc ssipi metop enhsh scaie sottr adri`

[1] Spaces are not part of ciphertext.

Composition: Building new ciphers from old

Let (E', D') and (E'', D'') be ciphers.
Their *composition* is the cipher (E, D) with keys of the form $k = (k'', k')$, where

$$E_{(k'',k')}(m) = E''_{k''}(E'_{k'}(m))$$

$$D_{(k'',k')}(c) = D'_{k'}(D''_{k''}(c)).$$

Can express this using *functional composition*.
$h = f \circ g$ is the function such that $h(x) = f(g(x))$.

Using this notation, we can write $E_{(k'',k')} = E''_{k''} \circ E'_{k'}$ and $D_{(k'',k')} = D'_{k'} \circ D''_{k''}$.

Subkey generation

When ciphers are composed, each component cipher needs a key called a *subkey*. Together, those subkeys can get rather large and unwieldy.

For practical reasons, the subkeys are themselves often generated by a deterministic process dependent on a *master key*, which is the user key of the resulting cryptosystem.

This is similar to the process of generating the stream of subkeys needed by a polyalphabetic substitution cipher such as Vigenère or the Enigma machines when encrypting multiletter messages.

Data Encryption Standard (DES)

Data encryption standard (DES)

The Data Encryption Standard is a block cipher that operates on 64-bit blocks and uses a 56-bit key.

It became an official Federal Information Processing Standard (FIPS) in 1976. It was officially withdrawn as a standard in 2005 after it became widely acknowledged that the key length was too short and it was subject to brute force attack.

Nevertheless, triple DES (with a 112-bit key) is approved through the year 2030 for sensitive government information.

The Advanced Encryption Standard (AES), based on the Rijndael algorithm, became an official standard in 2001. AES supports key sizes of 128, 192, and 256 bits and works on 128-bit blocks.

Feistel networks

DES is based on a *Feistel network*.

This is a general method for building an invertible function from any function f that scrambles bits.

It consists of some number of stages.

- Each stage i maps a pair of n-bit words (L_i, R_i) to a new pair (L_{i+1}, R_{i+1}). ($n = 32$ in case of DES.)
- By applying the stages in sequence, a t-stage network maps (L_0, R_0) to (L_t, R_t).
- (L_0, R_0) is the plaintext, and (L_t, R_t) is the corresponding ciphertext.

A Feistel network

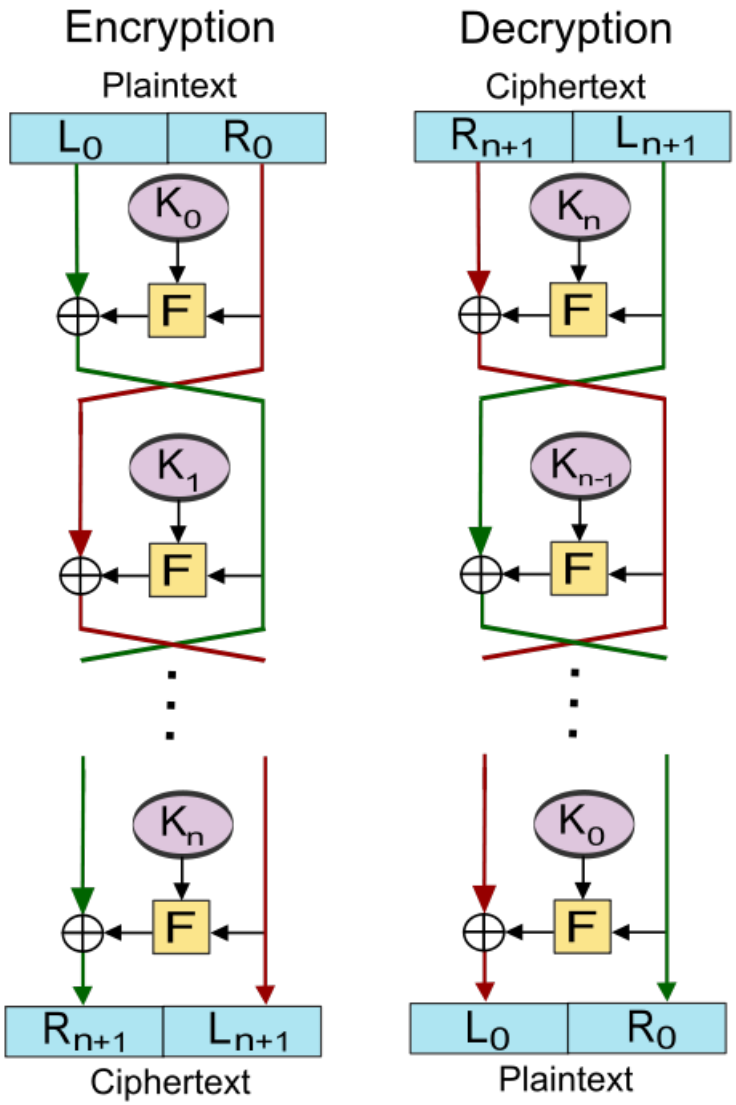

Image credit: Wikipedia: Feistel cipher,
http://en.wikipedia.org/wiki/Feistel_cipher

One stage

Each stage works as follows:

$$L_{i+1} = R_i \tag{1}$$

$$R_{i+1} = L_i \oplus f(R_i, K_i) \tag{2}$$

Here, K_i is a *subkey*, which is generally derived in some systematic way from the master key k, and f is the scrambling function (shown as F in the diagram).

The inversion problem is to find (L_i, R_i) given (L_{i+1}, R_{i+1}). Equation 1 gives us R_i. Knowing R_i and K_i, we can compute $f(R_i, K_i)$. We can then solve equation 2 to get

$$L_i = R_{i+1} \oplus f(R_i, K_i)$$

Properties of Feistel networks

The *security* of a Feistel-based code lies in the construction of the scrambling function f and in the method for producing the subkeys K_i.

The *invertibility* follows just from properties of $\oplus$ (exclusive-or).

DES Feistel network

DES uses a 16 stage Feistel network.

The pair $L_0 R_0$ is constructed from a 64-bit message by a fixed initial permutation IP.

The ciphertext output is obtained by applying IP^{-1} to $R_{16} L_{16}$.

The scrambling function $f(R_i, K_i)$ operates on a 32-bit data block and a 48-bit key block. Thus, $48 \times 16 = 768$ key bits are used.

The key bits are all derived in a systematic way from the 56-bit primary key and are far from independent of each other.

S-boxes

The scrambling function $f(R_i, K_i)$ is the heart of DES.

At the heart of the scrambling function are eight "S-boxes" that compute Boolean functions with 6 binary inputs

$$c_0, x_1, x_2, x_3, x_4, c_1$$

and 4 binary outputs y_1, y_2, y_3, y_4.

Each computes some fixed function in $\{0,1\}^6 \rightarrow \{0,1\}^4$.

The eight S-boxes are all different and are specified by tables.

Special properties of S-boxes

For fixed values of (c_0, c_1), the resulting function on inputs $x_1, \ldots, x_4$ is a permutation from $\{0,1\}^4 \rightarrow \{0,1\}^4$.

Hence, can regard an S-box as performing a substitution on four-bit "characters", where the substitution performed depends both on the structure of the particular S-box and on the values of its "control inputs" c_0 and c_1.

Thus, DES's 8 S-boxes are capable of performing 32 different substitutions on 4-bit fields.

DES scrambling network

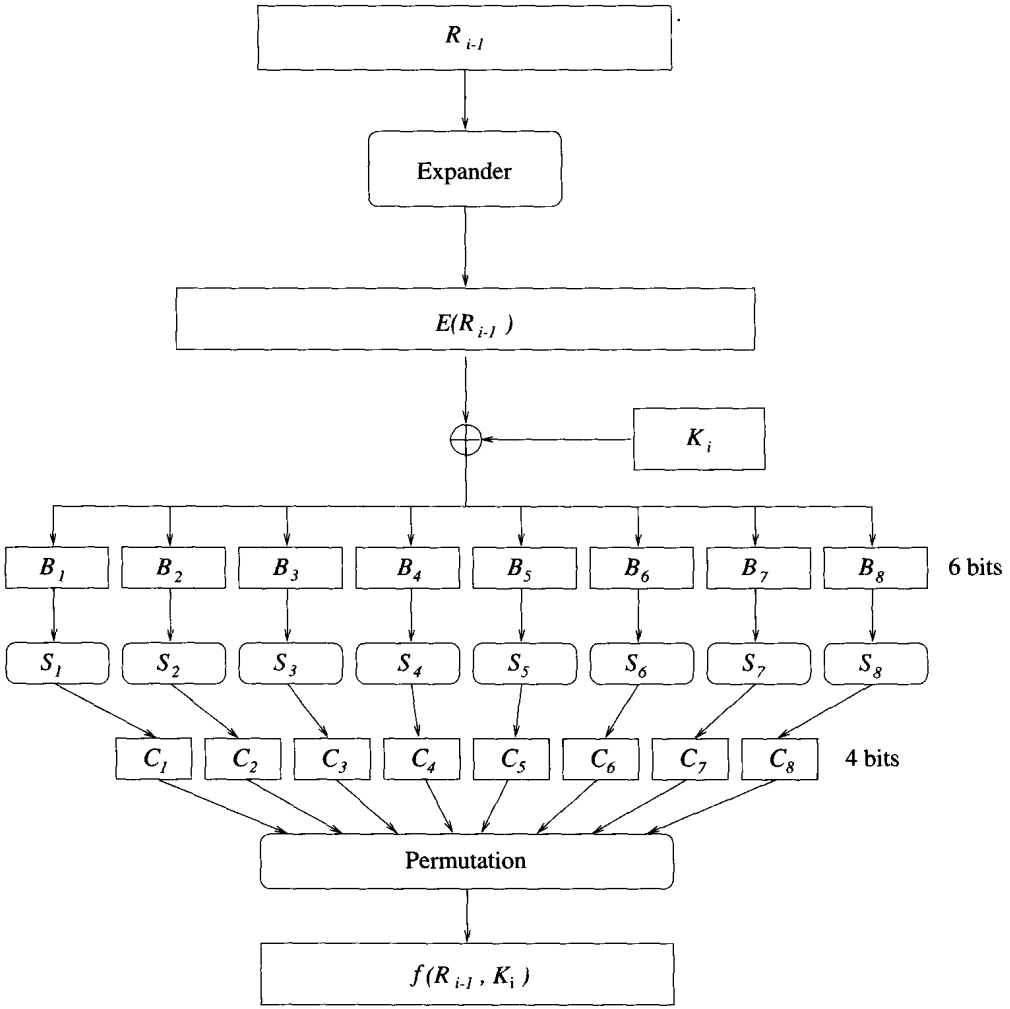

Figure 4.5: The DES Function $f(R_{i-1}, K_i)$.

Connecting the boxes

The S-boxes together have a total of 48 input lines.

Each of these lines is the output of a corresponding $\oplus$-gate.

- One input of each of these $\oplus$-gates is connected to a corresponding bit of the 48-bit subkey K_i. (This is the only place that the key enters into DES.)
- The other input of each $\oplus$-gate is connected to one of the 32 bits of the first argument of f.

Since there are 48 $\oplus$-gates and only 32 bits in the first argument to f, some of those bits get used more than once.

The mapping of input bits to $\oplus$-gates is called the *expansion permutation* E.

Expansion permutation

The expansion permutation connects input bits to $\oplus$ gates. We identify the $\oplus$ gates by the S-box inputs to which they connect.

- Input bits $32, 1, 2, 3, 4, 5$ connect to the six $\oplus$ gates that go input wires $c_0, x_1, x_2, x_3, x_4, c_1$ on S-box 1.
- Bits $4, 5, 6, 7, 8, 9$ are connect to the six $\oplus$ gates that go input wires $c_0, x_1, x_2, x_3, x_4, c_1$ on S-box 2.
- The same pattern continues for the remaining S-boxes.

Thus, input bits $1, 4, 5, 8, 9, \ldots 28, 29, 32$ are each used twice, and the remaining input bits are each used once.

Connecting the outputs

The 32 bits of output from the S-boxes are passed through a fixed permutation P (transposition) that spreads out the output bits.

The outputs of a single S-box at one stage of DES become inputs to several different S-boxes at the next stage.

This helps provide the desirable "avalanche" effect, where a change in one input bit spreads out through the network and causes many output bits to change.

Obtaining the subkey

The scrambling function operates on a 32-bit data block and a 48-bit key block (called a *subkey*).

The 56-bit master key k is split into two 28-bit pieces C and D. At each stage, C and D are rotated by one or two bit positions. Subkey K_i is then obtained by applying a fixed permutation (transposition) to CD.

Security considerations

DES is vulnerable to a brute force attack because of its small key size.

However, it has turned out to be remarkably resistant to recently-discovered (in the open world) sophisticated attacks.

Differential cryptanalysis: Can break DES using "only" 2^{47} chosen ciphertext pairs.

Linear cryptanalysis: Can break DES using 2^{43} chosen plaintext pairs.

Neither attack is feasible in practice.

Multiple Encryption

Composition of cryptosystems

Encrypting a message multiple times with the same or different ciphers and keys seems to make the cipher stronger, but that's not always the case. The security of the composition can be difficult to analyze.

For example, with the one-time pad, the encryption and decryption functions E_k and D_k are the same. The composition $E_k \circ E_k$ is the identity function!

Composition within practical cryptosystems

Practical symmetric cryptosystems such as DES and AES are built as a composition of simpler systems.

Each component offers little security by itself, but when composed, the layers obscure the message to the point that it is difficult for an adversary to recover.

The trick is to find ciphers that successfully hide useful information from a would-be attacker when used in concert.

Double Encryption

Double encryption is when a cryptosystem is composed with itself. Each message is encrypted twice using two different keys k' and k'', so $E^2_{(k'',k')} = E_{k''} \circ E_{k'}$ and $D^2_{(k'',k')} = D_{k'} \circ D_{k''}$.

(E, D) is the *underlying* or *base* cryptosystem and (E^2, D^2) is the *doubled* cryptosystem. $\mathcal{M}$ and $\mathcal{C}$ are unchanged, but $\mathcal{K}^2 = \mathcal{K} \times \mathcal{K}$.

The size of the keyspace is squared, resulting in an apparent doubling of the effective key length and making a brute force attack much more costly.

However, *it does not always increase the security* of a cryptosystem as much as one might naïvely think, for other attacks may become possible.

Example: Double Caesar

Consider Double Caesar, the Caesar cipher composed with itself.

It has $26^2 = 676$ possible key pairs (k'', k'). One might hope that double Caesar is more resistant to a brute force attack.

Unfortunately, still only 26 possible distinct encryption functions and only 26 possible decryptions of each ciphertext.

This is because $E^2_{(k'',k')} = E_k$ for $k = (k' + k'') \bmod 26$.

Any attack on the Caesar cipher will work equally well on the Double Caesar cipher. To the attacker, there is no difference between the two systems. Eve neither knows nor cares how Alice actually computed the ciphertext; all that matters are the probabilistic relationships between plaintexts and ciphertexts.

Group property

Let (E, D) be a cryptosystem for which $\mathcal{M} = \mathcal{C}$.
Each E_k is then a *permutation* on $\mathcal{M}$.[2]
The set of all permutations on $\mathcal{M}$ forms a *group*.[3]

Definition

(E, D) is said to have the *group property* if the set of possible encryption functions $\mathcal{E} = \{E_k \mid k \in \mathcal{K}\}$ is closed under functional composition $\circ$.

That is, if $k', k'' \in \mathcal{K}$, then there exists $k \in \mathcal{K}$ such that

$$E_k = E_{k''} \circ E_{k'}.$$

[2] A *permutation* is one-to-one and onto function.

[3] A *group* has an associative binary operation with an identity element, and each element has an inverse.

Cryptosystems with group property

We've seen that the Caesar cipher has the group property.

When $\mathcal{E}$ is closed under composition, then $(\mathcal{E}, \circ)$ is a subgroup of all permutations on $\mathcal{M}$. In this case, double encryption adds no security against a brute force attack.

Even though the key length has doubled, the number of distinct encryption functions has not increased, and the double encryption system will fall to a brute force attack on the original cryptosystem.

Triple DES

Triple DES uses two 56-bit DES keys k_1 and k_2 for a total key length of 112 bits.

The encryption of m is $c = E_{k_1}(D_{k_2}(E_{k_1}(m)))$.

The decryption of c is $m = D_{k_1}(E_{k_2}(D_{k_1}(m)))$.

Triple DES is still supported by some applications.