[CS304] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy Winter 2022-2023
Scribed by: Chitranshi Srivastava (202051055) Lecture 5 and 6 (Week #3)

In the previous week, we discussed about block ciphers and stream ciphers. We also learnt about One Time Padding(OTP) which we will continue from this session.

1 One Time Padding (OTP)

One Time Padding (OTP) provides perfect secrecy under some conditions.
Encryption:

$Enc(P,K) = P \oplus K = C$
where,
P → Plain Text
K → Secret Key
$\oplus$ → xor operation

Decryption:

$Dec(C,K) = C \oplus K = P$
To ensure that OTP provides perfect secrecy, that is,

$$P(message|Ciphertext) = P(message)$$

certain conditions have to be fulfilled. These are as follows : The conditions under which OTP provides perfect secrecy are as follows:

1. The secret key K cannot be used to encrypt two messages, that is, the key can not be reused.
2. The length of key must be greater than or equal to the length of message.
   $length(K) \geq length(P)$
3. Key K is uniformly selected from the key space.

Example :
Let us consider OTP on one bit.
$message, M \in \{0,1\} and key, K \in \{0,1\}$

$$Pr[M = 0] = p$$
$$Pr[M = 1] = 1 - p$$

Since we have assumed that key is uniformly selected,

$$\text{Pr[K = 0]} = \tfrac{1}{2}$$

$$\text{Pr[K = 1]} = \tfrac{1}{2}$$

Encryption :

$$C = M \oplus K$$

Cipher text can be 0 or 1. Let us find the probability.

$$\text{Pr[C = 0] = Pr[M = 0, K = 0] + Pr[M = 1, K = 1]}$$

$$Pr[C = 0] = Pr[M = 0] \cdot Pr[K = 0] + Pr[M = 1] \cdot Pr[K = 1]$$

$$\text{Pr[C = 0] = p}\times \tfrac{1}{2} + \text{(1-p)}\times \tfrac{1}{2}$$

$$\mathbf{Pr[C = 0] = \tfrac{1}{2}}$$

$$\text{Pr[C = 1] = 1 - Pr[C = 0]}$$

$$\mathbf{Pr[C = 1] = \tfrac{1}{2}}$$

Let us prove that OTP with the three conditions stated above provides perfect secrecy.

$$Pr[M = 0|C = 0] = \tfrac{Pr[M=0,C=0]}{Pr[C=0]}$$

$$Pr[M = 0|C = 0] = \tfrac{Pr[C=0|M=0]\times Pr[M=0]}{Pr[C=0]}$$

On observing the initial conditions, we see that given that M = 0, it is only possible to get C = 0 iff K = 0. That is, it depends on the probability of K.

$$Pr[M = 0|C = 0] = \tfrac{Pr[K=0]\times Pr[M=0]}{\frac{1}{2}}$$

$$\mathbf{Pr[M = 0 — C = 0] = Pr[M = 0]}$$

Since the above condition is proved, we can state that OTP under conditions provides perfect secrecy. This observation can be generalized for any n number of bits.
Now let us see how perfect secrecy is not satisfied when the conditions are not fulfilled.

- **When keys for two messages are same :**

  $$M_1 \oplus K = C_1$$
  $$M_2 \oplus K = C_2$$

  Let us take XOR of both cipher texts

  $$C_1 \oplus C_2 = (M_1 \oplus K) \oplus (M_2 \oplus K)$$
  $$C_1 \oplus C_2 = M_1 \oplus M_2$$

  We can see that using the cipher text, we can get information about the difference between the two plain texts. Hence, perfect secrecy is not satisfied.

- **When len(K) < len(P) :**

  Let us consider len(K) = 16 bits and len(P) = 32 bits

$$C = P \oplus K$$

$$C = P \oplus 0000000000000000k_1...k_{16}$$

$$C = p_1p_2....p_{16}c_{17}...c_{32}$$

We can see from above that the first 16 bits of plain text will be revealed as zeros will be appended before K to find XOR with P(32 bits) and $A \oplus 0 = A$. Hence, perfect secrecy is not satisfied.

If we try to optimize this process by repeating bits of K to avoid taking xor with 0s, even then some information is revealed. Proof is given below. Here, t < n

$$P = p_1p_2......p_n$$
$$K = k_1k_2...k_t$$

Let us repeat bits of k after t length.

$$C = (p_1 \oplus k_1)(p_2 \oplus k_2)(p_t \oplus k_t)(p_{t+1} \oplus k_1)...(p_n \oplus k_{n-t})$$
$$\text{Therefore,}$$
$$c_1 \oplus c_{t+1} = (p_1 \oplus k_1) \oplus (p_{t+1} \oplus p_1)$$
$$c_1 \oplus c_{t+1} = p_1 \oplus p_{t+1}$$

This again reveals information about the plain text. Therefore, perfect secrecy is not satisfied.

Hence, we can conclude that the three conditions stated above must be satisfied to attain perfect secrecy in One Time Padding(OTP).

2 Data Encryption Standard (DES)

DES is a block cipher designed by IBM. It is based on Feistel Networks. It has the following parameters:

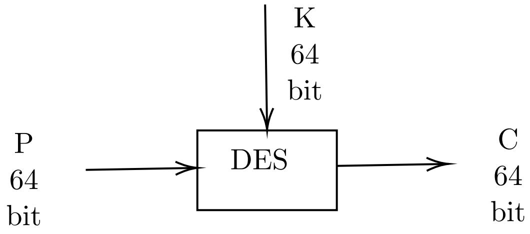
- Block Size = 64 bits
- Number of Rounds = 16
- Secret Key Size = 64 bits (8 parity check bits)

Initially, the design was kept secret and was used for personal communication only. But when the design came into the public domain, the cipher was broken immediately. Several weaknesses revealed the secret key in a short time.

The actual length of secret key is 56 bits. The rest 8 bits are parity bits to ensure the correctness of the main 56 bits.

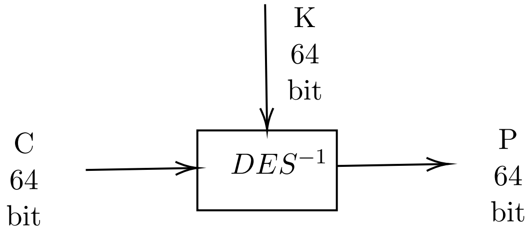
In sets of 8 bits, the last bit is the parity bit. To find the secret key of DES exhaustively, 2^{56} keys will be checked in the worst case.

Encryption:



It takes 64 bit message and a 64 bit key and generates 64 bit cipher text.

Decryption:



The secret key is 64 bits long with every 8th bit (from MSB side) as a parity bit. For Example,

Key: 01101010 11010001 11010111

The red coloured bits are parity bits. These are calculated by taking xor of the 7 bits prior to each parity bit. Now, if there is an odd number of bits altered in the 7 bits, it can be identified using the parity bit.
Note:However, parity bits will not help in identifying an alteration in even number of bits.

The first step of DES is to convert the 65-bit secret key to 56-bit by removing the parity bits. Hence, DES should provide 56-bit security.
In DES, there are 16 rounds. DES uses the same round function in every round. For each round, there is a unique round key K_i which is generated by the key scheduling algorithm. The key scheduling algorithm takes secret key as input and generate the round keys $K_1, K_2, ..., K_{16}$, that are of 48 bits.

Round Function :

$$f : \{0,1\}^{32} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$$

Encryption Algorithm :

- We pass the 64-bit input message to the IP which permutes the bits.
- The output from IP is then divided into 32-bit left and right halves, L_0 and R_0
- From i = 1 to 16, we execute 16 rounds of fiestal network where,

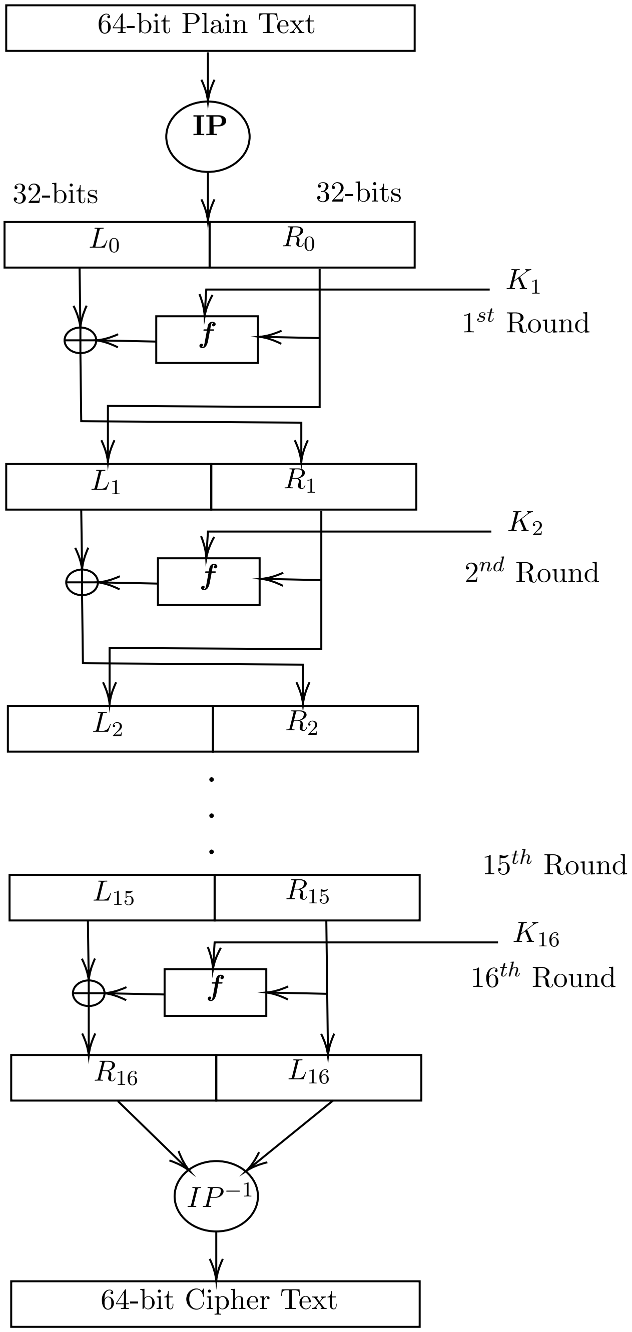
  $$L_i = R_{i-1} \; R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

  where

  $$f(R_{i-1}, K_i) = P(S(E(R_{i-1} \oplus K_i)))$$

- We then exchange L_{16} and R_{16}
- We then permute the final expression using IP^{-1}

A flowchart for encrypting a block of message is given below.



Here, IP is the Initial Permutation.

Similar to Feistel Network, for each round L_{i+1} and R_{i+1} will be:

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

where $i \in \{0, 1, ..., 15\}$. In the last round, i.e. 16^{th} round, the position of L_{16} and R_{16} are swapped. After this round, the inverse of Initial Permutation (IP^{-1}) is applied to the 64-bits and we get the cipher text.
We need to address the following now:

- Initial Permutation and its inverse
- The round function f
- How are the round keys $K_1, K_2, ..., K_{16}$ generated.

2.1 Initial Permutation

It is a bijection from 64-bit to 64-bit. The 64-bit message is permuted using the IP and then further encryption is done. Initial Permutation is defined as:

$$\text{IP} = \begin{bmatrix} 58 & 50 & 42 & 34 & 26 & 18 & 10 & 2 \\ 60 & 52 & 44 & 36 & 28 & 20 & 12 & 4 \\ 62 & 54 & 46 & 38 & 30 & 22 & 14 & 6 \\ 64 & 56 & 48 & 40 & 32 & 24 & 16 & 8 \\ 57 & 49 & 41 & 33 & 25 & 17 & 9 & 1 \\ 59 & 51 & 43 & 35 & 27 & 19 & 11 & 3 \\ 61 & 53 & 45 & 37 & 29 & 21 & 13 & 5 \\ 63 & 55 & 47 & 39 & 31 & 23 & 25 & 7 \end{bmatrix}$$

The permutation can be simply visualised as:

$$IP(m_1 m_2 ... m_7 m_8 m_9 .... m_{64}) = m_{58} m_{50} ... m_{10} m_2 m_{60} .... m_7$$

We can easily compute its inverse and it will be equal to:

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

2.2 Round Function of DES

$$f : \{0,1\}^{32} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$$
$$f(R_i, K_i) = X_i$$
$$\text{where,}$$
$$R_i \text{ is 32-bit}$$
$$K_i \text{ is 48-bit}$$
$$X_i \text{ is 32-bit.}$$

The round function for DES is defined as:

$$f(R_i, K_i) = P(S(E(R_i) \oplus K_i))$$

where,
Expansion Function E : $\{0,1\}^{32} \rightarrow \{0,1\}^{48}$
Substitution Box S: $\{0,1\}^{48} \rightarrow \{0,1\}^{32}$
Permutation Box P: $\{0,1\}^{32} \rightarrow \{0,1\}^{32}$
Hence,

$$\begin{gathered}
\text{length of } R_i = 32\text{-bits} \\
\text{length of } E(R_i) = 48\text{-bits} = \text{length of } K_i \\
\text{length of } E(R_i) \oplus K_i = 48\text{-bits} \\
\text{length of } S(E(R_i) \oplus K_i) = 32\text{-bits} \\
\text{length of } P(S(E(R_i) \oplus K_i)) = 32\text{-bits}
\end{gathered}$$

2.2.1 Expansion Function

$$E : \{0,1\}^{32} \rightarrow \{0,1\}^{48}$$

The expansion function for DES is given below:

$$E = \begin{bmatrix} 32 & 1 & 2 & 3 & 4 & 5 \\ 4 & 5 & 6 & 7 & 8 & 9 \\ 8 & 9 & 10 & 11 & 12 & 13 \\ 12 & 13 & 14 & 15 & 16 & 17 \\ 16 & 17 & 18 & 19 & 20 & 21 \\ 20 & 21 & 22 & 23 & 24 & 25 \\ 24 & 25 & 26 & 27 & 28 & 29 \\ 28 & 29 & 30 & 31 & 32 & 1 \end{bmatrix}$$

The bits are repeated for expanding 32-bits to 48-bits. In simple words, for each set of 4 bits, we add the LSB of prev set in the beginning of current set and the MSB of next set at the end of current set. We do in a circular manner for the first and last set.

$$E(x_1x_2....x_{32}) = (x_{32}x_1x_2x_3x_4x_5x_4x_5....x_{32}x_1)$$

2.2.2 Substitution Box

$$\begin{gathered}
S : \{0,1\}^{48} \rightarrow \{0,1\}^{32} \\
S(X) = Y, \text{ where X is 48 and Y is 32 bit long}
\end{gathered}$$

Dividing X into 8 parts each of length 6-bits.

$$X = B_1B_2B_3B_4B_5B_6B_7B_8$$

Corresponding to each B_i there is a substitution box S_i where $i \in \{1,2,..,8\}$.

$$\begin{gathered}
S_i : \{0,1\}^6 \rightarrow \{0,1\}^4 \forall i \in \{1,2,..,8\} \\
S_i(B_i) = C_i \\
\therefore S(X) = (S_1(B_1), S_2(B_2), S_3(B_3), S_4(B_4), S_5(B_5), S_6(B_6), S_7(B_7), S_8(B_8))
\end{gathered}$$

Therefore, length of S(X) is 32 bits. The substitution boxes are given on page 260 of the book

H́andbook of Applied Cryptography.́ Now let us see how to perform the conversion using Substitution box.

$$B_i = b_1b_2b_3b_4b_5b_6 \text{ , where } b_i \in \{0,1\}$$

We can find the row and column of the substitution box using these bits.

$$\text{r(row)} = 2 * b_1 + b_6,$$
$$\text{where r is integer representation of } b_1b_6 \text{ and } 0 \leq r \leq 3$$

$$\text{c(column)} = \text{integer representation of } b_2b_3b_4b_5$$
$$\text{where } 0 \leq c \leq 15$$

$$S_i = \begin{bmatrix} a_{0,0} & \cdots & a_{0,15} \\ \vdots & \ddots & \vdots \\ a_{3,0} & \cdots & a_{3,15} \end{bmatrix} \text{ where } a_{i,j} \in \{0, 1, ...15\}$$

now using this S_i :

$$S_i(B_i) = a_{r,c}$$

2.2.3 Permutation Box

$$P : \{0,1\}^{32} \rightarrow \{0,1\}^{32}$$

It is also defined by a table. The table is given below:

$$P = \begin{bmatrix} 16 & 7 & 20 & 21 \\ 29 & 12 & 28 & 17 \\ 1 & 15 & 23 & 26 \\ 5 & 18 & 31 & 10 \\ 2 & 8 & 24 & 14 \\ 32 & 27 & 3 & 9 \\ 19 & 13 & 30 & 6 \\ 22 & 11 & 4 & 25 \end{bmatrix}$$

Using this, we can see the permutation as:

$$P(x_1x_2x_3x_4x_5...x_{32}) = x_{16}x_7x_{20}x_{21}x_{29}...x_{25}$$

2.3 Key Scheduling Algorithm

Input : 64-bit secret key
Output : 16 round keys where len(k_i) = 48 bits

Algorithm :

- Define $v_i, 1 \leq i \leq 16$, where $v_i = 1$ if $i \in \{1, 2, 9, 16\}$, else $v_i = 2$.

- Discard 8 parity check bits from K. The 56 bit key is $\tilde{\text{K}}$.

- T = PC1($\tilde{\text{K}}$), where PC1 is a permuation defined as:

$$PC1 : \{0,1\}^{56} \rightarrow \{0,1\}^{56}$$

- $(C_0, D_0) = T$, where C_0 is most significant 28 bits of T and D_0 is least significant 28 bits of T.
- for $i = 1$ to 16:

$$C_i = (C_{i-1} \hookleftarrow v_i)$$
$$D_i = (D_{i-1} \hookleftarrow v_i)$$

where $\hookleftarrow$ is left circular shift

For example: $x_1x_2x_3....x_28 \hookleftarrow 2 = x_3x_4x_5....x_1x_2$ $K_i = PC2(C_i, D_i)$

where, PC2 is a substitution defined as:

$$PC2 : \{0,1\}^{56} \rightarrow \{0,1\}^{48}$$

2.3.1 PC1(Permuted Choice 1)

It permutes the 56 bits of secret key before generating round keys.

$$PC1 : \{0,1\}^{56} \rightarrow \{0,1\}^{56}$$

For C_i:

$$\text{PC1} = \begin{bmatrix} 57 & 49 & 41 & 33 & 25 & 17 & 9 \\ 1 & 58 & 50 & 42 & 34 & 26 & 18 \\ 10 & 2 & 59 & 51 & 43 & 35 & 27 \\ 19 & 11 & 3 & 60 & 52 & 44 & 36 \end{bmatrix}$$

For D_i:

$$\text{PC1} = \begin{bmatrix} 63 & 55 & 47 & 39 & 31 & 23 & 15 \\ 7 & 62 & 54 & 46 & 38 & 30 & 22 \\ 14 & 6 & 61 & 53 & 45 & 37 & 29 \\ 21 & 13 & 5 & 28 & 20 & 12 & 4 \end{bmatrix}$$

$$PC1(k_1k_2 \ldots k_7k_9 \ldots k_{63}) = (k_{57}k_{49}k_{41}k_{33} \ldots k_9k_1k_{58} \ldots k_{63}k_{55} \ldots k_4)$$

2.3.2 PC2(Permuted Choice 2)

It is used to select 48 bits from the concatenation $b_1b_2 \ldots b_{56}$ of C_i and D_i

$$\text{PC2} = \begin{bmatrix} 14 & 17 & 11 & 24 & 1 & 5 \\ 3 & 28 & 15 & 6 & 21 & 10 \\ 23 & 19 & 12 & 4 & 26 & 8 \\ 16 & 7 & 27 & 20 & 13 & 2 \\ 41 & 52 & 31 & 37 & 47 & 55 \\ 30 & 40 & 51 & 45 & 33 & 48 \\ 44 & 49 & 39 & 56 & 34 & 53 \\ 46 & 42 & 50 & 36 & 29 & 32 \end{bmatrix}$$

$$K_i = b_{14}b_{17}b_{11} \ldots b_32$$

2.4 Complementarity Property of DES

We know that in expansion function, we simply use the bits and map them using the expansion table. Hence, taking complement after mapping is same as mapping the complemented bits. Therefore,

$$E(\overline{R_0}) \oplus \overline{K} = \overline{E(R_0)} \oplus \overline{K}$$

Therefore,

$$P(S(E(\overline{R_0}) \oplus \overline{K})) = P(S(\overline{E(R_0)} \oplus \overline{K}))$$

$$\Rightarrow f(R_0, K) = f(\overline{R_0}, \overline{K})$$

Now let us see for message M, key K :

$$\begin{gathered} M = L_0 \parallel R_0 \\ L_1 = R_0 \\ R_1 = L_0 \oplus f(R_0, K) \\ C = L1 \parallel R_1 \end{gathered}$$

For message $\overline{M}, key \overline{K}$, where $\overline{M}$ and $\overline{K}$ are complements of M and K respectively:

$$\begin{gathered} \overline{M} = \overline{L_0} \parallel \overline{R_0} \\ \tilde{L_1} = \overline{R_0} \\ \tilde{R_1} = \overline{L_0} \oplus f(\overline{R_0}, \overline{K}) \end{gathered}$$

But we proved above that: $f(R_0, K) = f(\overline{R_0}, \overline{K})$.
Hence,

$$\tilde{R_1} = \overline{L_0} \oplus f(R_0, K)$$

We also know that $\overline{A} \oplus B = \overline{A \oplus B}$. So,

$$\begin{gathered} \tilde{R_1} = \overline{L_0 \oplus f(R_0, K)} \\ \tilde{R_1} = \overline{R1} \end{gathered}$$

We can conclude :

$$\tilde{L_1} \parallel \tilde{R_1} = \overline{L_1} \parallel \overline{R_1}$$

$$\Rightarrow \tilde{C} = \overline{C}$$

The proof that we have seen above is for round. But if we consider the whole process of DES, if the secret key K is complemented, the round keys K_i will also be complemented because they are the permutations of secret key K. For the next round, the input will be $\overline{C_1}$ and $\overline{K_2}$. So it will generate output as $\overline{C_2}$. Similarly, after all 16 rounds also, this complementarity property will hold. And in the final step, IP^{-1} also performs permutation, which again maintains complementarity.