[CS304] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy
Scribed by: Akshay (202051018)

Winter 2022-2023
Lecture (Week 3)

1 One Time Padding

One time padding provides perfect secrecy under some conditions.

Encryption:

P : Plaintext
K : Secret key

Enc(P,K) = P⊕K = C

Decryption:
Dec(C,K) = C⊕K = P

$$Pr[message|ciphertext] = Pr[message]$$

Conditions:

- The secret key K cannot be used to encrypt two messages. (No repeatition of K)
- length(K) ≥ length(P)
- K is uniformly selected from the key space.

1.1 OTP on 1 Bit Enc.

message: m ∈ {0,1} K ∈ {0,1}

Pr[m=0] = p Pr[K=0] = $\frac{1}{2}$
Pr[m=1] = 1-p Pr[K=1] = $\frac{1}{2}$

(Key is uniformly distributed, that's why 0 and 1 have equal probability)

Encryption:
C = m⊕K
C = 0 $\implies$ (m=0 and K=0) or (m=1 and K=1)
Pr[C=0] = Pr[m=0,K=0] + Pr[m=1,K=1]
Pr[C=0] = Pr[m=0].P[K=0] + Pr[m=1].Pr[K=1]

$$= (p \times \frac{1}{2}) + (1-p) \times \frac{1}{2}$$
$$= \frac{1}{2}$$

Similarly, Pr[C=1] = $\frac{1}{2}$

This should hold for perfect secrecy:

$$Pr[M = m|C = c] = Pr[M = m]$$

Let's check:

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

Thus it provides perfect secrecy.

1.2 Unfavourable Conditions

1)

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

(This will reveal information on messages)

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

Ciphertext difference will give you message difference. i^{th} bit $= 0 \implies$ both message have same i^{th} bit (revealed some information).

2) len(K) < len(P)
Let len(P) = n, and len(K) = t
One way to use the key to encrypt P is by padding n-t zeroes at starting or ending but that will explicitly reveal some part of message. We must avoid using constants in the key.
What we do is we repeat bits of the same key to fill the gap.

$P = P_1....P_l....P_n$
$\oplus K = K_1....K_lK_1....K_t$

$C = (P_1 \oplus K_1)\ (P_2 \oplus K_2)\ ....\ (P_l \oplus K_l)\ (P_{l+1} \oplus K_1)\ ....\ (P_n \oplus K_t)$

If we xor C_i bit from first l bits with C_{l+i} bit then we will get $(P_i \oplus P_{l+i})$ which is revealing information about the message.

OTP is not usable in real life because we will not be able to reuse the secret key.

2 Data Encryption Standard (DES)

- It is a Block Cipher.
- Designed by IBM. Initially design was kept secret by IBM. When the design was made public it was immediately broken because of certain weaknesses.

1. Block size = 64 bit
2. Number of rounds = 16
3. Secret key size = 64 bit with 8 parity check bits (means there are 8 bits depending on the 56 bits).
4. It is based on Feistel Network.

Encryption:

$$\text{P (64 bit)} \rightarrow \underset{\downarrow \text{K (64 bit)}}{\boxed{\text{DES}}} \rightarrow \text{C of 64 bit}$$

Decryption:

$$\text{C (64 bit)} \rightarrow \underset{\downarrow \text{K (64 bit)}}{\boxed{DES^{-1}}} \rightarrow \text{P of 64 bit}$$

Secret key is 64 bit with 8 parity check bits.
01101010 11010001 01
Parity check bit is placed after every 7 bits. It is equal to the xor of previous seven bits. Parity check bits are used to check if the key is altered. After discarding 8 parity check bits the final secret key will be of 56 bit. DES should provide 56 bit security.

In DES we have 16 round keys $K_1, K_2, ...K_{16}$ (sub keys).
length(K_i) = 48 bits
They (sub keys) are generated using key scheduling algorithm. Key scheduling algorithm will take the secret key as input and output 16 sub keys, G(K).

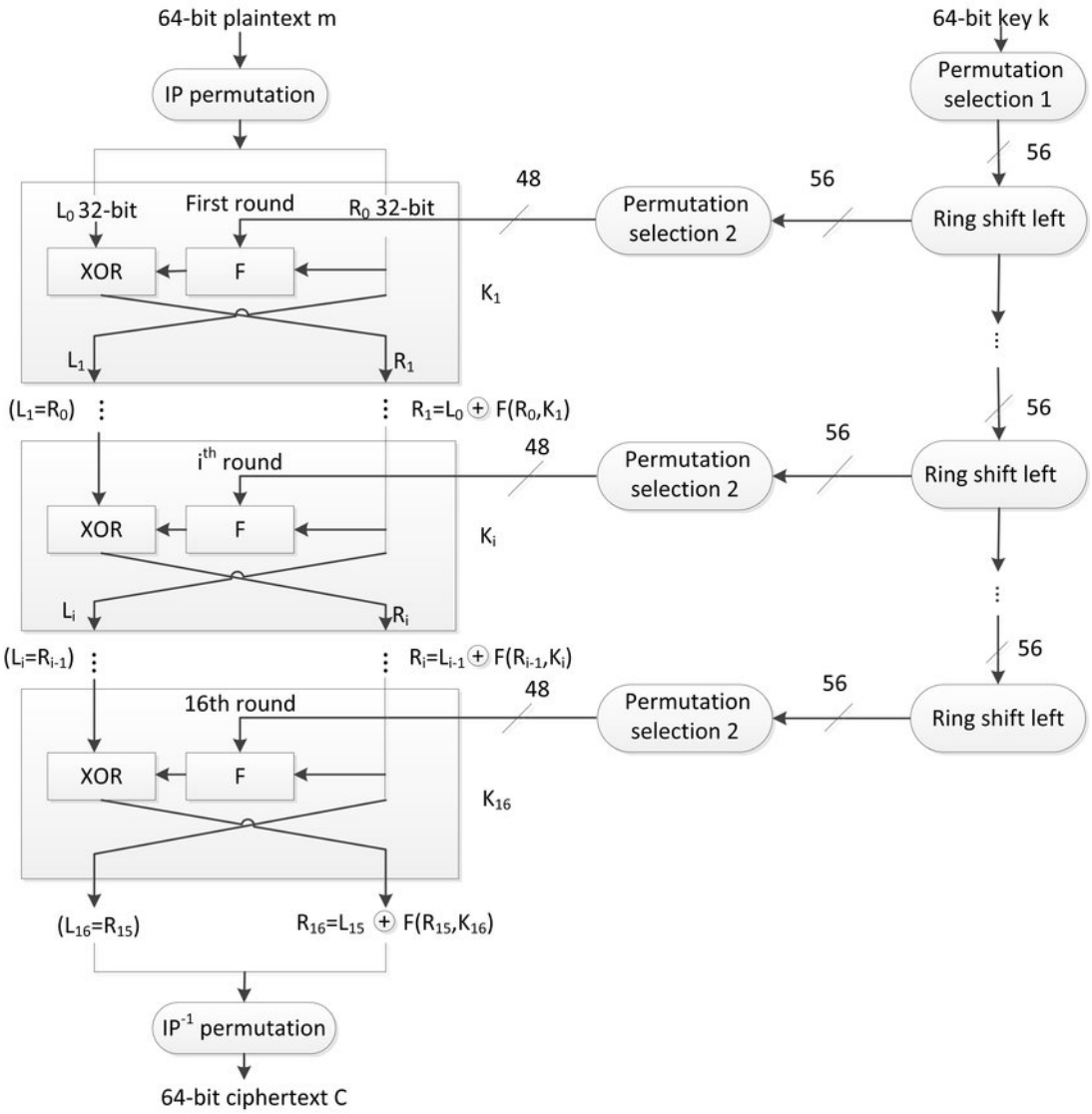

Image source: ResearchGate

$F : \{0,1\}^{32} \times \{0,1\}^{48} \to \{0,1\}^{32}$ (takes 32 bit right half of IP output and 48 bit key as argument and outputs 32 bit).
$L_i = R_{i-1}$; $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$
Note: In decryption we do not require the F to be invertible.

We have to learn the followings:

1. IP, IP^{-1}
2. F (round function)
3. How $K_1, K_2, ...K_{16}$ (round keys) are generated.

2.1 IP (Initial Permutation)

$$IP : \{0,1\}^{64} \to \{0,1\}^{64}$$

IP							
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	15	7

This table is basically having the 64 positions for 64 bits. We do the permutation in our message according to this table. Take the positions from the table row wise.

$$IP(m_1\ m_2\ ....\ m_{64}) = (m_{58}\ m_{50}\ .....\ m_{15}\ m_7)$$

58^{th} is placed at first position.
$L_0 = m_{58}\ m_{50}\ ....\ m_8$; $R_0 = m_{57}\ m_{49}\ ....\ m_7$

IP^{-1}							
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

2.2 Round Function

$F(R_{i-1}, K_i) = X_i$; $F : \{0,1\}^{32} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$

$$\boxed{\mathrm{F(R}_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))}$$

- $E : \{0,1\}^{32} \rightarrow \{0,1\}^{48}$ (Expansion function)
- $S : \{0,1\}^{48} \rightarrow \{0,1\}^{32}$ (Substitution box)
- $P : \{0,1\}^{32} \rightarrow \{0,1\}^{32}$ (Permutation box)

2.2.1 Expansion Function

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

E					
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

$$E(x_1\ x_2\ x_3\ x_4\ ....\ x_{32}) \rightarrow (x_{32}\ x_1\ x_2\ ....\ x_{32}\ x_1)$$

Some bits are repeated to expand the size to 48 from 32.

2.2.2 S Box

$S : \{0,1\}^{48} \rightarrow \{0,1\}^{32}$
S(X) = Y
X = $B_1\ B_2\ B_3\ B_4\ B_5\ B_6\ B_7\ B_8$, where length of B_i is 6 bit
$S_1,\ S_2,\ S_3,\ S_4,\ S_5,\ S_6,\ S_7,\ S_8$
$S_i : \{0,1\}^6 \rightarrow \{0,1\}^4 \qquad \forall\ \ i = 1, 2, ...8$
$S_i(B_i) = C_i$ (4 bit)

These S_i are fixed (pre-defined).

> $S_i(B_i)$ maps $B_i = b_1b_2....b_6$ to the 4-bit entry in row r and column c of S_i in table given below.
> Where, $r = 2.b_1 + b_6$ and $b_2b_3b_4b_5$ is radix-2 representation of $0 \leq c \leq 15$.
> E.g.: $S_1(011011)$ yields r=1, c=13, and output 5, i.e., binary 0101.

$0 \leq r \leq 3$; Basically r is the radix-2 representation of (b_1b_6)

row	column number															
	[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

Image source: Handbook of Applied Cryptography

2.2.3 Permutation Box

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

P			
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

$$P(x_1\ x_2\ ....\ x_{32}) = (x_{16}\ x_7\ x_{20}\ ....\ x_4\ x_{25})$$

2.3 Key Scheduling Algorithm

Input: 64 bit key K
Output: 16 round keys K_i , $1 \leq i \leq 16$, where length of K_i is 48 bit.

- Define v_i, $1 \leq i \leq 16$
  - $v_i = 1$ if $i \in \{1, 2, 9, 16\}$
  - else $v_i = 2$
- Discard 8 parity check bits from K.
- T = PC1(K)
  PC1 : $\{0,1\}^{56} \rightarrow \{0,1\}^{56}$
- (C_0, D_0) = T
  where C_0 is of 28 bit (most significant part) and D_0 is of 28 bit (least significant part)
- for i=1 to 16:
  - C_i = leftCircularShift(C_{i-1}) by v_i unit
  - D_i = leftCircularShift(D_{i-1}) by v_i unit
  - $K_i = PC2(C_i, D_i)$
    PC2: $\{0,1\}^{56} \rightarrow \{0,1\}^{48}$

PC1 and PC2 are substitution tables.

Table 1: **DES key schedule bit selections (PC1)**

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
above for C_i ; below for D_i						
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

Table 2: **DES key schedule bit selections (PC2)**

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