DES 加密解密算法

孙一鸣　Andy
bearsugar@foxmail.com
2017-3-9

IP 置换表　(表 1)

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

Step 1

要被加密的数据被分割成为若干以 64bit 为单位的数据
如果位数不够，则补 00 或者 FF，然后按 表 1 进行置换操作

密钥生成置换表 (表 2)

C0

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

D0

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

Step 2

把 step1 得到的 64bit 密钥按 表 2 做置换，去除密钥中作为奇偶校验位的第 8、16、24、32、40、48、56、64 位，剩下的 56 位作为有效输入密钥

56 位
前 28bit C0，后 28bit D0

左移位数表（表 3）

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

Step 3

将 56 位有效的密钥按照 【表 3】 左移位，共进行 16 轮左移操作。每一轮左移结束之后，得到的新的 56 位子密钥作为下一轮移位的有效密钥。

该过程总计得到 16 个 56 位的子密钥

密钥生成置换表（压缩）（表 4）

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

Step 4

按照 【表 4】 对 16 个 56 位子密钥进行置换压缩，压缩后每个子密钥中的第 9、18、22、35、38、43、54 共 8 位数据丢失。

Step 4 完成后，得到 16 个 48 位的子密钥

E 位选择表 扩展置换表（表 5）

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

Step 5

将 step1 生成的 64 位子密钥的右半部分 Ri 的 32 位数据根据 【表 5】 进行置换，由原来的 32 位扩展到 48 位

Step 6

将扩展后的有效数据的新的 R[i+1]和 K[i+1] 做异或运算

(表 6)

// S1

14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7
0 15 7 4 14 2 13 1 10 6 12 11 9 5 3 8
4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0
15 12 8 2 4 9 1 7 5 11 3 14 10 0 6 13

//S2

15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10
3 13 4 7 15 2 8 14 12 0 1 10 6 9 11 5
0 14 7 11 10 4 13 1 5 8 12 6 9 3 2 15
13 8 10 1 3 15 4 2 11 6 7 12 0 5 14 9

//S3

10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8
13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1
13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7
1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12

//S4

7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15
13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9
10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4
3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14

//S5

2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9
14 11 2 12 4 7 13 1 5 0 15 10 3 9 8 6
4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14
11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3

//S6

12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11
10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8
9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6
4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13

//S7

4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1
13 0 11 7 4 9 1 10 14 3 5 12 2 15 8 6
1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2
6 11 13 8 1 4 10 7 9 5 0 15 14 2 3 12

//S8

13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7
1 15 13 8 10 3 7 4 12 5 6 11 0 14 9 2
7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8
2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11

Step 7

盒运算

将 step6 产生的 48bit 的加密数据分为 8 个 6 位的数据 按照 【表 6】取值。每 6bit 压缩成 4bit，最终形成 8 个 6bit 的数据，组合成新的 32bit 的数据

例如：100101

第 1 位：1

第 6 位：1

行：11 = 3

列：中间 4bit = 0010 = 2

位置：3，2 = 8 = $(1000)_2$

P 换位表（单纯换位表）（表 7）

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

Step 8

将 step7 产生的 32 位数据按照【表 7】置换，生成新的 R[i]数据

Step 9

把 R[i]和 step 1 产生的 L[i]按位异或后的值赋给 step1 产生的 R[i+1]，然后原来的 R[i]值赋给 L[i+1]，得到新的 L[i+1]和 R[i+1]

Step 10

回到 step5，重复运算到 step 10，每轮计算由 R[i]， L[i]， K[i]来计算，总计重复运算 16 轮结束，最终生成 L[16]和 R[16]

逆置换 IP-1 表（表 9）

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

Step 11

合并 L[16]和 R[16]为 64 位的数据，然后按照【表 9】做置换，生成最终加密数据