Q1 Teamname

0 Points

NULL

Q2 Commands

10 Points

List the commands used in the game to reach the ciphertext.

go, dive, dive, back, pull, back, back, go, wave, back, thrnxxtzy, read, 3608528850368400786036725, c, read, password

Q3 Cryptosystem

5 Points

What cryptosystem was used at this level? Please be precise.

6-round DES

Q4 Analysis

80 Points

Knowing which cryptosystem has been used at this level, give a detailed description of the cryptanalysis used to figure out the password. (Explain in less than 150 lines and use Latex wherever required. If your solution is not readable, you will lose marks. If necessary, the file upload option in this question must be used TO SHARE IMAGES ONLY.)

Assuming that the cryptosystem used is 6-round DES we did the following to find the key:

Our approach was to do a chosen plain text attack using differential cryptanalysis. Since we need to break 6-round DES, we need a 4-round characteristic. We used the following characteristic which is mentioned in the lecture slides.

 $(405C\overline{0}, 0400\overline{0}, \frac{1}{4}, 0400\overline{0}, 0054\overline{0}, \frac{5}{128}, 0054\overline{0}, \overline{00}, 1, \overline{00}, 0054\overline{0}, \frac{5}{128}, 00)$

The probability of the characteristic is 0.00038 which means that we need at least $20/0.00038\approx 53000$ pairs of plaintext pairs with xor equal to $405C\overline{0}0400\overline{0}$ after applying the initial permutation. So we need plain text pairs with xor equal to $IP^{-1}\big(405C\overline{0}0400\overline{0}\big)=0000901010005000$ so that after applying IP their xor would be equal to the required value.

The hint mentions that 2 letters are represented by 1 byte, so only 256 distinct pairs of letters can be present in the ciphertext. To find those pairs, we queried using random inputs, analyzed the outputs, and found that the 256 pairs are all possible pairs of 2 letters where each letter comes from 'f' to 'u' in the alphabet using the program "analyse.cpp". Thus, We assumed that 'f' is represented as 0000, 'g' as 0001,, 'u' as 1111 and generated 100000 input pairs with above required xor and queried their outputs using the program

"gen_input_output_pairs.cpp".

We converted the outputs into their bit representation assuming the bit representation of each letter as stated above and applied the Initial Permutation to undo the inverse IP which would be done at the end of DES and also swapped the Left and right parts to get the output of Round 6 of DES using "process_outputs.cpp".

To find the sixth round key, we did the following:

Firstly, we define some notation which we use later

- $L_i R_i$ denotes the output of i^{th} round of DES.
- Let $E\left(R_5
 ight)=lpha_1lpha_2\cdotslpha_8=lpha$ and $E\left(R_5'
 ight)=lpha_1'lpha_2'\cdotslpha_8'=lpha'$ with $|lpha_i|=6=|lpha_i'|$
- R_5 and R_5' are right-halves of output of fifth round on the plaintexts L_0R_0 and $L_0'R_0'$ where $L_0\oplus L_0'=405C\overline{0}$ and $R_0\oplus R_0'=0400\overline{0}$.
- Let $eta_i=lpha_i\oplus k_{6,i}$ and $eta_i'=lpha_i'\oplus k_{6,i}, |eta_i|=6=|eta_i'|.$
- $-k_6 = k_{6,1}k_{6,2}\cdots k_{6,8}$
- Let $\gamma_{i}=S_{i}\left(eta_{i}
 ight)$ and $\gamma_{i}'=S_{i}\left(eta_{i}'
 ight),\left|\gamma_{i}
 ight|=4=\left|\gamma_{i}'
 ight|$

For a given pair of plaintexts $L_0R_0,L_0'R_0'$ with xor value $405C\overline{0}0400\overline{0}$, we know the output of 6^{th} round of DES i.e $L_6R_6,L_6'R_6'$, Since $R_5=L_6$,we also know the output of expansion in round 6 $E(R_5),E(R_5')$. And since we know the value of $L_5\oplus L_5'=R_4\oplus R_4'=0400\overline{0}$ with probability 0.00038, we know the value of $\gamma=S(\beta)\oplus S(\beta')=P^{-1}(L_5\oplus L_5'\oplus R_6\oplus R_6')$ with probability 0.00038.

- Thus, We know $\alpha_i,\alpha_i',\beta_i\oplus\beta_i'=\alpha_i\oplus\alpha_i'$, and a value of γ such that $S(\beta)\oplus S(\beta')=\gamma$ with probability 0.000381.

For each $1 \leq i \leq 8$, we generated all possible (β_i,β_i') such that $\beta_i \oplus \beta_i' = \alpha_i \oplus \alpha_i'$ and $S(\beta_i) \oplus S(\beta_i') = \gamma_i$ and added the corresponding key $k = \alpha_i \oplus \beta_i$ to a hashmap containing list of possible $k_{6,i}$ values. We did this for all 10000 pairs and took the key with highest frequency in the hashmap as $k_{6,i}$. The code for this can be found in "find_r6_key.cpp".

Here are the frequencies of most frequent, second most frequent element in the hashmap for each $1 \leq i \leq 8$.

For i=1 8281, 6682

For i=2 8667, 6682

For i = 3 6418, 6417

For i = 4 6479, 6447

For i = 5 9221, 6821

For i = 6 8741, 6613

For i = 7 8422, 6674

For i = 8 8719, 6872

As we can see, the first most frequent element and second element do not differ much for i=3 and i=4. So we discarded $k_{6,3}$ and $k_{6,4}$ obtained from the above analysis and mapped the other bits to their positions in the main key using key scheduling algorithm and then used brute force to find them along with the other 8 unknown bits. We used a known plaintext ciphertext pair "pjjjstpmigntlltf", "snmfqkhtsjoinofl" and brute forced all 2^{20} possibilities to find the remaining 20 bits.The code for this can be found in "find_key.cpp".

Using the hint given on the screen, we got that the encrypted password is "ountnlqktqortppnkkqqnrhipifmiigt". Since the password is 128 bits long we divided it into two blocks of 64 bits each and decrypted each of them using the DES decryption algorithm as we know the key. We converted the decrypted password to alphabetic representation assuming that each character is 4 bits but the formed password did not work. So we tried

assuming each letter is 8 bits (like the usual ASCII representation) and got "mkpnizcefq000000" as output which also did not work. After some trial and error, we observed that removing zeros at the end and entering "mkpnizcefq" works which also makes sense since the zeros at the end might be added just to make the length of text multiple of the block size. The code for this portion can be found in "find_key.cpp".

Thus we found that the password is "mkpnizcefq".



No files uploaded

Q5 Password

5 Points

What was the final command used to clear this level?

```
mkpnizcefq
```

Q6 Codes

0 Points

Unlike previous assignments, this time it is mandatory that you upload the codes used in the cryptanalysis. If you fail to do so, you will be given 0 marks for the entire assignment.

```
▼ analyse.cpp
                                                               ≛ Download
    #include <bits/stdc++.h>
2
    using namespace std;
3
    #define endl '\n'
4
5
    mt19937
6
     rng(chrono::steady_clock::now().time_since_epoch().count());
7
    int getRand(int l, int r)
8
9
         uniform int distribution<int> uid(l, r);
10
         return uid(rng);
11
12
13
    void getinputs()
```

```
14
    {
15
        ofstream fout("test_inputs.txt");
16
17
        int L = 1000;
        for (int i = 0; i < L; i++)
18
19
20
             uint64 t first = 0;
             string s = "";
21
22
             for (int j = 0; j < 8; j++)
23
24
                 int k = getRand(0, 255);
25
                 first = (first << 4) + k;
26
                 s += char(k);
27
             }
28
             fout << s << endl;
29
        }
30
31
32
    void getoutputs()
33
34
        ifstream fin("test inputs.txt");
35
        ofstream fout("test cmds.txt");
36
37
        fout << "NULL" << endl;</pre>
38
        fout << "foobar268" << endl;</pre>
39
        fout << 4 << endl;
40
41
        fout << "read" << endl;</pre>
42
43
        string s;
        while (fin >> s)
44
45
        {
46
             fout << s << endl;
47
             fout << 'c' << endl;
48
        }
49
50
        fout << "back" << endl;</pre>
51
        fout << "exit" << endl;</pre>
52
53
        fout.close();
54
        fin.close();
55
56
        system("ssh student@65.0.124.36 < test cmds.txt > out");
57
        system("grep --no-group-separator -A 1 \"Slowly, a new text
    starts appearing on the screen. It reads ...\" out | grep --no-
    group-separator -v \"Slowly, a new text starts appearing on the
    screen. It reads \dots | tr -d \"\\t\" > test_outputs.txt");
        system("rm -rf out test cmds.txt");
58
59
    }
60
    void analyze()
61
62
```

```
63
        ifstream fin("test outputs.txt");
        map<pair<char,char>, int> cnt;
64
65
        string s;
66
        while (fin >> s)
67
        {
68
             int n = s.size();
             for (int i = 0; i < n; i += 2)
69
70
             {
71
                 cnt[make\_pair(s[i], s[i + 1])]++;
72
             }
73
        }
74
        for (auto [c, ] : cnt)
75
        {
76
             cout << c.first << ' ' << c.second << endl;</pre>
77
         }
78
        cout << endl;</pre>
79
    }
80
81
    int main()
82
    {
83
84
        getinputs();
85
        getoutputs();
86
        analyze();
87
        return 0;
88
89
    }
90
```

```
♣ Download
▼ gen_input_output_pairs.cpp
 1
     #include <bits/stdc++.h>
 2
     using namespace std;
 3
 4
    #define endl '\n'
 5
 6
     mt19937
     rng(chrono::steady clock::now().time since epoch().count());
 7
     int getRand(int l, int r)
 8
     {
 9
         uniform_int_distribution<int> uid(l, r);
10
         return uid(rng);
11
     }
12
13
     void getinputs()
14
15
         uint64 t diff = 0x405c000004000000;
16
17
         ofstream fout("inputs.txt");
18
         int IP[] = {
19
             58, 50, 42, 34, 26, 18, 10, 2,
20
             60, 52, 44, 36, 28, 20, 12, 4,
```

```
21
            62, 54, 46, 38, 30, 22, 14, 6,
22
             64, 56, 48, 40, 32, 24, 16, 8,
            57, 49, 41, 33, 25, 17, 9, 1,
23
24
            59, 51, 43, 35, 27, 19, 11, 3,
25
            61, 53, 45, 37, 29, 21, 13, 5,
26
            63, 55, 47, 39, 31, 23, 15, 7};
27
28
        uint64 t rdiff = 0;
29
        for (int i = 0; i < 64; i++)
30
31
             uint64 t cur = ((diff >> (63 - i)) & 1);
32
             int to = IP[i];
33
             rdiff |= (cur << (64 - to));
34
        }
35
36
        cout << hex << setw(16) << setfill('0') << rdiff << endl;</pre>
37
38
        int L = 1e5;
39
        for (int i = 0; i < L; i++)
40
        {
41
             uint64_t first = 0;
             string s = "";
42
43
             for (int j = 0; j < 16; j++)
44
45
                 int k = getRand(0, 15);
46
                 first = (first << 4) + k;
                 s += char('f' + k);
47
48
             }
49
             fout << s << endl;
             s = "";
50
51
             uint64_t second = (first ^ rdiff);
52
             for (int j = 0; j < 16; j++)
53
             {
54
                 int num = 0;
55
                 for (int k = 0; k < 4; k++)
56
                 {
57
                     int at = 4 * j + k;
58
                     int cur = ((second >> (63 - at)) \& 1);
59
                     num += (cur << (3 - k));
60
                 }
                 s += char('f' + num);
61
62
             }
             fout << s << endl;
63
64
        }
65
66
    void getoutputs()
67
68
        ifstream fin("inputs.txt");
69
70
        ofstream fout("cmds.txt");
71
72
        fout << "NULL" << endl;
```

```
73
         fout << "foobar268" << endl;</pre>
74
         fout << 4 << endl;
75
         fout << "read" << endl;</pre>
76
77
78
         string s;
79
         while (fin >> s)
80
         {
81
             fout << s << endl;
82
             fout << 'c' << endl;
83
         }
84
         fout << "back" << endl;</pre>
85
86
         fout << "exit" << endl;</pre>
87
88
         fout.close();
89
         fin.close();
90
91
         system("ssh student@65.0.124.36 < cmds.txt > out");
92
         system("grep --no-group-separator -A 1 \"Slowly, a new text
     starts appearing on the screen. It reads ...\" out | grep --no-
     group-separator -v \"Slowly, a new text starts appearing on the
     screen. It reads ...\" | tr -d \"\\t\" > outputs.txt");
         system("rm -rf out cmds.txt");
93
94
     }
95
96
     int main()
97
98
99
         getinputs();
100
         getoutputs();
101
102
         return 0;
103 }
104
```

```
♣ Download
▼ process_outputs.cpp
 1
     #include <bits/stdc++.h>
 2
 3
     using namespace std;
 4
 5
     using b64 = bitset<64>;
 6
     using b32 = bitset<32>;
 7
     using b48 = bitset<48>;
 8
     using b56 = bitset<56>;
 9
     using b28 = bitset<28>;
10
11
     int IP[] = {
12
         58, 50, 42, 34, 26, 18, 10, 2,
13
        60, 52, 44, 36, 28, 20, 12, 4,
14
        62, 54, 46, 38, 30, 22, 14, 6,
```

```
15
        64, 56, 48, 40, 32, 24, 16, 8,
16
        57, 49, 41, 33, 25, 17, 9, 1,
        59, 51, 43, 35, 27, 19, 11, 3,
17
        61, 53, 45, 37, 29, 21, 13, 5,
18
19
        63, 55, 47, 39, 31, 23, 15, 7};
20
21
    int Inv IP[] = {
22
        40, 8, 48, 16, 56, 24, 64, 32,
        39, 7, 47, 15, 55, 23, 63, 31,
23
24
        38, 6, 46, 14, 54, 22, 62, 30,
25
        37, 5, 45, 13, 53, 21, 61, 29,
26
        36, 4, 44, 12, 52, 20, 60, 28,
27
        35, 3, 43, 11, 51, 19, 59, 27,
        34, 2, 42, 10, 50, 18, 58, 26,
28
29
        33, 1, 41, 9, 49, 17, 57, 25};
30
31
    b64 apply_ip(b64 in, bool inv = false)
32
33
        b64 out;
34
        for (int i = 63; i >= 0; i--)
35
36
             int j = (inv ? Inv_IP[63 - i] : IP[63 - i]) - 1;
37
            out[i] = in[63 - j];
38
39
        return out;
40
    }
41
42
    pair<b32, b32> get_LR(b64 in)
43
44
        b32 L, R;
45
        for (int i = 63, j = 31; i >= 32; i --, j --)
46
        {
47
            L[j] = in[i];
            R[j] = in[i - 32];
48
49
50
        return make_pair(L, R);
51
52
    b64 join(b32 L, b32 R)
53
54
        b64 out;
        for (int i = 31; i >= 0; i--)
55
56
57
             out[i + 32] = L[i];
58
            out[i] = R[i];
59
        }
60
        return out;
61
    }
62
63
    int main()
64
65
66
        ifstream fin("outputs.txt");
```

```
67
        ofstream fout("outbits.txt");
68
69
        string s;
70
        while (fin >> s)
71
        {
72
             assert((int)s.length() == 16);
73
             b64 bits = 0;
74
             for (int i = 0; i < 16; i++)
75
76
                 char c = s[i];
77
                 int num = c - 'f';
78
                 assert(num < 16 and num >= 0);
                 bits = (bits \ll 4) | b64(num);
79
80
             }
81
             b64 actual = apply ip(bits);
82
             auto [R, L] = get LR(actual);
83
84
             fout << L << R << endl;
85
        }
86
87
        return 0;
88
    }
```

```
≛ Download
▼ find_r6_key.cpp
 1
     #include <bits/stdc++.h>
 2
 3
     using namespace std;
 4
 5
     using b64 = bitset<64>;
 6
     using b32 = bitset<32>;
 7
     using b48 = bitset<48>;
 8
     using b56 = bitset<56>;
 9
     using b28 = bitset<28>;
10
11
     int IP[] = {
12
         58, 50, 42, 34, 26, 18, 10, 2,
13
         60, 52, 44, 36, 28, 20, 12, 4,
14
         62, 54, 46, 38, 30, 22, 14, 6,
15
         64, 56, 48, 40, 32, 24, 16, 8,
16
         57, 49, 41, 33, 25, 17, 9, 1,
17
         59, 51, 43, 35, 27, 19, 11, 3,
18
         61, 53, 45, 37, 29, 21, 13, 5,
19
         63, 55, 47, 39, 31, 23, 15, 7};
20
21
     int Inv IP[] = {
22
         40, 8, 48, 16, 56, 24, 64, 32,
23
         39, 7, 47, 15, 55, 23, 63, 31,
24
         38, 6, 46, 14, 54, 22, 62, 30,
25
         37, 5, 45, 13, 53, 21, 61, 29,
26
         36, 4, 44, 12, 52, 20, 60, 28,
27
         35, 3, 43, 11, 51, 19, 59, 27,
```

```
28
        34, 2, 42, 10, 50, 18, 58, 26,
29
        33, 1, 41, 9, 49, 17, 57, 25};
30
31
    b64 apply ip(b64 in, bool inv = false)
32
33
        b64 out;
34
        for (int i = 63; i \ge 0; i - -)
35
        {
36
             int j = (inv ? Inv IP[63 - i] : IP[63 - i]) - 1;
37
            out[i] = in[63 - j];
38
        }
39
        return out;
40
    }
41
42
    pair<b32, b32> get LR(b64 in)
43
44
        b32 L, R;
        for (int i = 63, j = 31; i >= 32; i --, j --)
45
46
        {
47
            L[j] = in[i];
48
            R[j] = in[i - 32];
49
50
        return make pair(L, R);
51
    }
52
53
    int E[] = {
54
        32, 1, 2, 3, 4, 5,
55
        4, 5, 6, 7, 8, 9,
56
        8, 9, 10, 11, 12, 13,
57
        12, 13, 14, 15, 16, 17,
58
        16, 17, 18, 19, 20, 21,
59
        20, 21, 22, 23, 24, 25,
        24, 25, 26, 27, 28, 29,
60
        28, 29, 30, 31, 32, 1};
61
62
    b48 apply_E(b32 in)
63
64
    {
65
        b48 out;
66
        for (int i = 47; i >= 0; i--)
67
        {
             int j = E[47 - i] - 1;
68
69
            out[i] = in[31 - j];
70
        }
71
        return out;
72
    }
73
74
    int S1[4][16] = {
75
        14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7,
76
        0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8,
77
        4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0,
78
        15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13};
79
```

```
80
     int S2[4][16] = {
81
         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,
82
        0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15,
83
84
        13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9};
85
86
     int S3[4][16] = {
87
         10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8,
88
         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,
89
90
         1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12};
91
92
     int S4[4][16] = {
93
        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,
94
        10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4,
95
96
         3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14};
97
98
     int S5[4][16] = {
99
        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,
100
        4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14,
101
102
        11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3};
103
104
     int S6[4][16] = {
        12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11,
105
106
        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,
107
        4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13};
108
109
110
     int S7[4][16] = {
111
        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,
112
        1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2,
113
         6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12};
114
115
116
    int S8[4][16] = {
        13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7,
117
118
        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,
119
         2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11};
120
121
122
     b32 apply S(b48 in)
123
     {
124
        b32 out;
125
        int i = 47, j = 31;
        for (auto S: {S1, S2, S3, S4, S5, S6, S7, S8})
126
127
             int row = 0, col = 0;
128
             for (int k = i - 1; k > i - 5; k--)
129
130
             {
                 col = col * 2 + in[k];
131
```

```
}
132
133
             row = in[i] * 2 + in[i - 5];
134
135
             bitset<4> num = S[row][col];
136
             for (int k = j; k >= j - 3; k--)
137
138
                 out[k] = num[3 - (j - k)];
139
             }
140
             i -= 6;
             j -= 4;
141
142
         }
143
         return out;
144
    }
145
     int P[] = {
146
         16, 7, 20, 21, 29, 12, 28, 17,
147
148
         1, 15, 23, 26, 5, 18, 31, 10,
149
         2, 8, 24, 14, 32, 27, 3, 9,
150
         19, 13, 30, 6, 22, 11, 4, 25};
151
152
    int Inv_P[32];
153
    void calc Inv P()
154
155
156
         for (int i = 0; i < 32; i++)
157
         {
158
             int to = P[i] - 1;
159
             Inv_P[to] = i + 1;
160
         }
161
    }
162
     b32 apply P(b32 in, bool inv = false)
163
164
165
         b32 out;
166
         for (int i = 31; i \ge 0; i - -)
167
         {
             int j = (inv ? Inv_P[31 - i] : P[31 - i]) - 1;
168
169
             out[i] = in[31 - j];
170
         }
171
         return out;
172
173
     using b4 = bitset<4>;
175
     using b6 = bitset<6>;
176
177
     b4 calc(b6 in, int i)
178
     {
179
         b4 out;
180
         int cur = 0;
181
         for (auto S: {S1, S2, S3, S4, S5, S6, S7, S8})
182
         {
183
             if (i == cur)
```

```
{
184
185
                 int row = 0, col = 0;
                 for (int k = 4; k > 0; k--)
186
187
                 {
188
                     col = col * 2 + in[k];
189
                 }
190
                 row = in[5] * 2 + in[0];
191
192
                 bitset<4> num = S[row][col];
193
                 for (int k = 3; k \ge 0; k - -)
194
                 {
195
                     out[k] = num[k];
196
                 }
197
                 return out;
198
             }
199
             cur = cur + 1;
200
         }
         assert(false);
201
202
         return out;
203
204
    #define endl '\n'
205
206
    int main()
207
         ifstream fin("outbits.txt");
208
209
         calc Inv P();
210
211
         int N = 1e5;
212
213
         unordered map<bitset<6>, int> maybe[8];
214
215
         for (int = 0; < N; ++)
216
         {
217
             bitset<32> out1L, out1R, out2L, out2R;
218
             fin >> out1L >> out1R >> out2L >> out2R;
219
220
             // outputs after expansion in the last round
221
             bitset<48> out1E = apply E(out1L);
222
             bitset<48> out2E = apply E(out2L);
             bitset<48> inxorS = (out1E ^ out2E);
223
224
225
             // cout << inxorS << endl;</pre>
226
227
             bitset<32> xorL = bitset<32>(0x04000000);
             bitset<32> xorP = (out1R ^ out2R ^ xorL);
228
229
             bitset<32> outxorS = apply_P(xorP, true);
230
231
             for (int i = 0; i < 8; i++)
232
             {
233
                 int st = i * 6;
234
                 for (int j = 0; j < (1 << 6); j++)
235
                 {
```

```
236
                      bitset<6> in1 = j;
237
                      bitset<6> in2 = 0;
238
                      for (int k = st; k - st < 6; k++)
239
                      {
                          in2[5 - k + st] = (in1[5 - k + st] ^ inxorS[47]
240
     - k]);
241
                      }
242
                      bitset<4> out1 = calc(in1, i);
243
                      bitset<4> out2 = calc(in2, i);
244
                      bitset<4> got = (out1 ^ out2);
245
                      bool ok = 1;
246
                      for (int k = i * 4; k - i * 4 < 4; k++)
247
248
                          ok &= (outxorS[31 - k] == got[3 - k + i * 4]);
249
                      }
250
251
                      if (ok)
252
                      {
253
                          bitset<6> cand;
254
                          for (int k = st; k - st < 6; k++)
255
256
                              cand[5 - k + st] = in1[5 - k + st] ^
     out1E[47 - k];
257
258
                          maybe[i][cand]++;
259
                      }
260
                 }
261
             }
262
         }
263
264
         bitset < 48 > key = 0;
265
266
         for (int i = 0; i < 8; i++)
267
         {
268
             int cur_mx = 0;
269
             bitset<6> ans;
270
             for (auto [bs, cnt] : maybe[i])
271
272
                 if (cnt > cur_mx)
273
                 {
274
                      cur mx = cnt;
275
                      ans = bs;
276
                 }
277
             }
278
             key = (key \ll 6);
279
             for (int j = 0; j < 6; j++)
280
281
                 key[j] = ans[j];
282
             }
283
         }
284
285
         cout << key << endl;</pre>
```

```
286
287 return 0;
288 }
```

```
Download
▼ find_key.cpp
     #include <bits/stdc++.h>
 1
 2
 3
     using namespace std;
 4
 5
     using b64 = bitset<64>;
 6
     using b32 = bitset<32>;
 7
     using b48 = bitset<48>;
 8
     using b56 = bitset<56>;
 9
     using b28 = bitset<28>;
10
11
     int IP[] = {
12
         58, 50, 42, 34, 26, 18, 10, 2,
13
         60, 52, 44, 36, 28, 20, 12, 4,
14
         62, 54, 46, 38, 30, 22, 14, 6,
15
         64, 56, 48, 40, 32, 24, 16, 8,
16
         57, 49, 41, 33, 25, 17, 9, 1,
17
         59, 51, 43, 35, 27, 19, 11, 3,
         61, 53, 45, 37, 29, 21, 13, 5,
18
         63, 55, 47, 39, 31, 23, 15, 7};
19
20
21
     int Inv IP[] = {
22
         40, 8, 48, 16, 56, 24, 64, 32,
23
         39, 7, 47, 15, 55, 23, 63, 31,
24
         38, 6, 46, 14, 54, 22, 62, 30,
         37, 5, 45, 13, 53, 21, 61, 29,
25
26
         36, 4, 44, 12, 52, 20, 60, 28,
27
         35, 3, 43, 11, 51, 19, 59, 27,
         34, 2, 42, 10, 50, 18, 58, 26,
28
29
         33, 1, 41, 9, 49, 17, 57, 25};
30
31
     b64 apply_ip(b64 in, bool inv = false)
32
33
         b64 out;
34
         for (int i = 63; i >= 0; i--)
35
             int j = (inv ? Inv IP[63 - i] : IP[63 - i]) - 1;
36
37
             out[i] = in[63 - j];
38
39
         return out;
40
     }
41
42
     pair<b32, b32> get LR(b64 in)
43
44
         b32 L, R;
45
         for (int i = 63, j = 31; i >= 32; i --, j --)
46
```

```
47
            L[j] = in[i];
48
            R[j] = in[i - 32];
49
50
        return make pair(L, R);
51
    }
52
    int E[] = {
53
54
        32, 1, 2, 3, 4, 5,
        4, 5, 6, 7, 8, 9,
55
        8, 9, 10, 11, 12, 13,
56
57
        12, 13, 14, 15, 16, 17,
58
        16, 17, 18, 19, 20, 21,
59
        20, 21, 22, 23, 24, 25,
        24, 25, 26, 27, 28, 29,
60
        28, 29, 30, 31, 32, 1};
61
62
63
    b48 apply E(b32 in)
64
65
        b48 out;
        for (int i = 47; i >= 0; i--)
66
67
68
            int j = E[47 - i] - 1;
69
            out[i] = in[31 - j];
70
        }
71
        return out;
72
    }
73
74
    int S1[4][16] = {
75
        14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7,
76
        0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8,
77
        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};
78
79
80
    int S2[4][16] = {
81
        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,
82
83
        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};
84
85
86
    int S3[4][16] = {
87
        10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8,
88
        13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1,
89
        13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7,
90
        1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12};
91
92
    int S4[4][16] = {
93
        7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15,
94
        13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9,
95
        10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4,
96
        3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14};
97
98
    int S5[4][16] = {
```

```
99
         2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9,
100
         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,
101
         11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3};
102
103
104
     int S6[4][16] = {
105
         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,
106
         9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6,
107
         4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13};
108
109
110
     int S7[4][16] = {
         4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1,
111
         13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6,
112
         1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2,
113
         6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12};
114
115
116
     int S8[4][16] = {
117
         13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7,
118
         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,
119
         2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11};
120
121
122
     b32 apply S(b48 in)
123
     {
124
         b32 out;
125
         int i = 47, j = 31;
         for (auto S: {S1, S2, S3, S4, S5, S6, S7, S8})
126
127
         {
             int row = 0, col = 0;
128
             for (int k = i - 1; k > i - 5; k--)
129
130
             {
                 col = col * 2 + in[k];
131
132
             }
             row = in[i] * 2 + in[i - 5];
133
134
135
             bitset<4> num = S[row][col];
             for (int k = j; k >= j - 3; k--)
136
137
             {
138
                 out[k] = num[3 - (j - k)];
139
             }
140
             i -= 6;
141
             i -= 4;
142
143
         return out;
144
     }
145
146
     int P[] = {
147
         16, 7, 20, 21, 29, 12, 28, 17,
         1, 15, 23, 26, 5, 18, 31, 10,
148
149
         2, 8, 24, 14, 32, 27, 3, 9,
         19, 13, 30, 6, 22, 11, 4, 25};
150
```

```
151
152
     int Inv_P[32];
153
154
    void calc Inv P()
155
156
         for (int i = 0; i < 32; i++)
157
         {
158
             int to = P[i] - 1;
159
             Inv P[to] = i + 1;
160
         }
161
     }
162
163
     b32 apply P(b32 in, bool inv = false)
164
165
         b32 out;
166
         for (int i = 31; i >= 0; i--)
167
             int j = (inv ? Inv P[31 - i] : P[31 - i]) - 1;
168
             out[i] = in[31 - i];
169
170
         }
171
         return out;
172
     }
173
174
    using v64 = array<int, 64>;
     using v32 = array<int, 32>;
175
176
     using v48 = array<int, 48>;
177
     using v56 = array<int, 56>;
     using v28 = array<int, 28>;
178
179
180
    int PC1[] = {
181
         57, 49, 41, 33, 25, 17, 9,
         1, 58, 50, 42, 34, 26, 18,
182
183
         10, 2, 59, 51, 43, 35, 27,
         19, 11, 3, 60, 52, 44, 36,
184
185
         63, 55, 47, 39, 31, 23, 15,
186
         7, 62, 54, 46, 38, 30, 22,
187
         14, 6, 61, 53, 45, 37, 29,
188
         21, 13, 5, 28, 20, 12, 4};
189
     unsigned short shifts[] = {
190
191
         1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1};
192
193
     int PC2[] = {
194
         14, 17, 11, 24, 1, 5,
195
         3, 28, 15, 6, 21, 10,
196
         23, 19, 12, 4, 26, 8,
         16, 7, 27, 20, 13, 2,
197
198
         41, 52, 31, 37, 47, 55,
199
         30, 40, 51, 45, 33, 48,
200
         44, 49, 39, 56, 34, 53,
201
         46, 42, 50, 36, 29, 32};
202
```

```
203
    vector<v48> keys(16);
204
205
     void get keys()
206
207
         v56 key;
208
         for (int i = 55; i \ge 0; i - -)
209
210
             key[i] = 56 - i;
211
         }
212
213
         v28 C, D;
214
         for (int i = 55, j = 27; i \ge 28; i - -, j - -)
215
         {
216
             C[j] = key[i];
217
             D[j] = key[i - 28];
218
         }
219
220
         for (int r = 0; r < 16; r++)
221
         {
222
             v28 rC, rD;
223
             int s = shifts[r];
             for (int i = 27; i \ge 0; i--)
224
225
             {
226
                  int to = (i + s) % 28;
227
                  rC[to] = C[i];
228
                  rD[to] = D[i];
229
             }
             C = rC, D = rD;
230
231
232
             for (int i = 47; i \ge 0; i - -)
233
234
                  int j = PC2[47 - i] - 1;
235
                  j = 55 - j;
236
                  if (j > 27)
237
                  {
238
                      keys[r][i] = C[j - 28];
239
                  }
240
                  else
241
                  {
242
                      keys[r][i] = D[j];
243
244
             }
245
         }
246
     }
247
248
     void print(v48 key)
249
         for (int i = 47; i \ge 0; i = 6)
250
251
         {
252
             int num = 0;
253
             for (int j = i; j >= i - 5; j --)
254
             {
```

```
255
                num = num * 2 + key[j];
256
            }
257
            cout << setw(2) << setfill('0') << hex << num;</pre>
258
        }
        cout << " ";
259
260
261
262
    b64 join(b32 L, b32 R)
263
264
        b64 out;
265
        for (int i = 31; i \ge 0; i - -)
266
267
            out[i + 32] = L[i];
268
            out[i] = R[i];
269
270
        return out;
271
272
273
    b64 get(string s)
274
275
        assert(s.length() == 16);
276
        b64 ans = 0;
277
        for (int i = 0; i < 16; i++)
278
279
            int foo = s[i] - 'f';
280
            ans = (ans << 4) | b64(foo);
281
282
        return ans;
283
284
285
    vector<b48> rkeys(16);
    void get keys2(vector<int> key)
286
287
288
        for (int r = 0; r < 16; r++)
289
290
            for (int i = 47; i \ge 0; i - -)
291
            {
292
                int j = keys[r][i] - 1;
293
                rkeys[r][i] = key[55 - j];
294
            }
295
        }
296
297
298
    int main()
299
300
        get_keys();
301
        b48 \text{ key } r6 =
    302
303
        vector<int> key(56, -1);
304
        for (int i = 47, cur = 0; i >= 0; i -= 6, cur++)
305
```

```
306
             if (cur == 2 \text{ or } cur == 3)
307
                  continue;
308
             for (int j = i; j >= i - 5; j --)
309
             {
310
                  int k = keys[5][j] - 1;
311
                  key[55 - k] = key r6[j];
312
             }
         }
313
314
315
         auto doDES = [&](b64 in, bool inv = false) {
316
             b64 LR = apply ip(in);
             auto [L, R] = get_LR(LR);
317
318
             for (int r = 0; r < 6; r++)
319
             {
320
                  auto ER = apply E(R);
321
                  ER ^= (inv ? rkeys[^5 - r] : rkeys[r]);
                 auto SR = apply_S(ER);
322
323
                  auto PR = apply P(SR);
324
                  auto nL = R, nR = L ^ PR;
325
                 L = nL, R = nR;
326
             }
327
             auto 0 = join(R, L);
328
             auto out = apply ip(0, true);
329
             return out;
330
         };
331
332
         vector<int> pos;
333
         int cnt = 0;
334
         for (int i = 55; i \ge 0; i - -)
335
         {
336
             if (key[i] == -1)
337
                  pos.push back(i);
338
             cnt += (key[i] == -1);
339
         }
340
341
         calc Inv P();
342
         b64 in = get("pjjjstpmigntlltf");
343
344
         b64 out = get("snmfqkhtsjoinofl");
345
         for (int i = 0; i < (1 << cnt); i++)
346
347
         {
             for (int j = 0; j < cnt; j++)
348
349
             {
                 if ((i \& (1 << j)) != 0)
350
351
                  {
352
                      key[pos[j]] = 1;
353
                  }
354
                  else
355
                      key[pos[j]] = 0;
356
357
             get keys2(key);
```

```
358
              if (doDES(in) == out)
359
                  cout << "Found the key !!" << endl;</pre>
360
                  for (int k = 55; k \ge 0; k - -)
361
362
                  {
363
                       cout << key[k];</pre>
364
                  }
365
                  cout << endl;</pre>
366
                  break;
367
              }
368
         }
369
370
         b64 in21 = get("ountnlqktqortppn");
371
         b64 in22 = get("kkqqnrhipifmiigt");
372
         for (auto in2 : {in21, in22})
373
374
375
              auto out2 = doDES(in2, true);
              for (int i = 63; i >= 0; i -= 8)
376
377
              {
378
                  int num = 0;
379
                  for (int j = i; j >= i - 7; j --)
380
381
                       num = num * 2 + out2[j];
382
                  }
383
                  cout << char(num);</pre>
384
              }
385
         }
386
         cout << endl;</pre>
387
388
         return 0;
389
    }
```

Assignment 4

UNGRADED

GROUP

AJAY PRAJAPATI

A5 - SURYADEVARA SAI KRISHNA

A11 - GARIMELLA MOHAN RAGHU

View or edit group

TOTAL POINTS

- / 100 pts	
QUESTION 1	
Teamname	0 pts
QUESTION 2	
Commands	10 pts
QUESTION 3	
Cryptosystem	5 pts
OUESTION 4	
QUESTION 4 Analysis	80 pts
QUESTION 5	F
Password	5 pts
QUESTION 6	
Codes	0 pts