Q1 Team Name
0 Points

Group Name

force_de_fem

Q2 Commands
5 Points

List all the commands in sequence used from the start screen of this level to the end of the level. (Use -> to separate the commands)

go/enter -> dive-> dive-> back -> pull -> back -> back -> go-> wave -> back -> thrnxxtzy -> read -> the_magic_of_wand -> c -> read -> password

Q3 Cryptosystem
10 Points

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

The cryptosystem used at this level is 6- round DES (Block cipher).

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. (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.)

After successfully retrieving the magic wand from the bottom of the lake, we freed the spirit by returning back to

level 3. Then we tried to read the first screen of level 4 again and got some hints from the spirit and got the below ciphertext by typing 'password' as "$pmitsggknlsnfgfhmskumuomnltgirmu$". In order to decrypt this ciphertext, we utilized several hints given by spirit. In the hints, it was mentioned that the cryptosystem can be 4,6 or 10-round DES. It was also mentioned that a 4-round DES will be easy to break and spirit also that it is definitely not a 10-round DES. Therefore, we went ahead with the assumption that the cryptosystem for level is 6-round DES, as the same approach can be applied to 4-round DES if our assumption is wrong. The chosen-plaintext attack was used to break 6-round DES. In order to apply this type of attack, we need to obtain ciphertexts against a bunch of generated plaintexts. These plaintexts and ciphertext pairs will be utilized to find the encryption key. Steps of DES –Algorithm:

1. Initial Permutation [IP(m)]: Initial Permutation is applied to plaintext 'm' before the first round and happens only once.
2. Key transformation:

(a) CP1: The initial 64-bit master key is transformed into a 56-bit key by discarding every 8th bit of the master key.
(b) Shift: The output of KT1 is circularly shifted left by one or two positions depending on the number of rounds. In our case bits will be shifted by two positions.
(c) CP2: The shifted output of 56 bits is mapped to 48 bits. Here, CP stands for Compression Permutation.

3. Expansion [E(M)]: After IP operation the plaintext is divided into left and right plaintext known as LPT and RPT respectively. During expansion, the RPT 'M' is expanded from 32 bits to 48 bits.
4. Permutation [P(M)]: The 32-bit input 'M' is permuted in this step.
5. S – box substitution: There are 8 boxes namely $S_1, S_2, \ldots,$ and S_8 which take 6-bit input and provide a 4-bit output.
6. Final Permutation/Inverse Initial Permutation [IP inv(m)]: Applied on message 'm' after the completion of all 6 rounds of DES.

Our Cryptanalysis Approach: -

1. In the hints it was mentioned that two letters make up

one byte. Thus, 4 bits are used to represent one character, 4 bits can represent 16 characters only which means that there are only 16 ciphertext characters used in the game. We went through several ciphertexts in order to identify these 16 characters and finally came to the conclusion that only alphabets f to u are
used in the game.
2. Equipped with this knowledge we went ahead and mapped the letters f-u to 0-15 as follows: $f:0000, g:0001, h:0010, I:0011, j:0100, k:0101, l:0110, m:0111, n:1000, o:1001, p:1010, q:1011, r:1100, s:1101, t:1110, u:1111.$
3. We also noticed that the length of the ciphertexts generated in the game is 16 letters, which agrees with the input and output size of one DES block i.e., 64 bits = 8 bytes which means 16 letters.
4. At last, we decided to perform differential cryptanalysis using a chosen-plaintext attack and two 3-round characteristics:
$40080000, 04000000, and 00200008, 00000400$ with probability $1/16$ which were taught in class.
5. The detailed methodology is explained below sections:

Step 1 - Randomly generate plaintexts and their corresponding ciphertexts.
We generated 1000 plaintexts such that their XOR after applying the inverse of initial permutation on characteristic {40080000, 04000000} was {00008010, 00004000} and another 1000 plaintexts for characteristic {00200080, 00000400} has XOR {00000801, 00100000}. The code to generate plaintexts is in generating $generate\_inputs.ipynb$ and these plaintexts were stored in plaintexts1.txt and plaintexts2.txt. The ciphertexts corresponding to these plaintexts were generated by providing these plaintexts to the server and storing the obtained outputs. In order to automate this process, $server1.py$ and $server2.py$ were used to connect to the server and the outputs were stored in ciphertexts1.txt and ciphertexts2.txt respectively.

Step 2 - Generating round-6 key.
After we get the ciphertext to the corresponding plaintext satisfying the 3-round characteristic, we perform

differential cryptanalysis to get the 6^{th} round 48-bit key denoted as K_6. These ciphertexts are converted to binary form using the mapping mentioned above. Now we used $CryptAnalysis.ipynb$, along with the ciphertexts to apply to reverse final permutation to get the output of the 6^{th} round denoted as (L_6, R_6) and (L'_6, R'_6). We know that R_5 is equal to L_6, hence we can get the values R_5 and R'_5. We can compute the output of expansion and input to the S-box of the 6^{th} round using these values. The output of permutation for the 6^{th} round is calculated using $L_5 \oplus (R_6 \oplus R'_6)$, where L_5 = 04000000 from the first characteristic and L_5 = 00000400 from the second characteristic. Next, we applied inverse permutation on the output to get the output XOR of the S-box for the 6^{th} round. The detailed differential cryptanalysis for DES is carried out as follows:

* Expansion: Let $E(R_5) = \alpha_1\alpha_2\alpha_3\alpha_4\alpha_5\alpha_6\alpha_7\alpha_8$ and $E(R'_5) = \alpha'_1\alpha'_2\alpha'_3\alpha'_4\alpha'_5\alpha'_6\alpha'_7\alpha'_8$, where $|\alpha_i| = |\alpha'_i|$ = 6.
* XOR: Compute β_i = $\alpha_i \oplus k_{5,i}$ and β'_i = $\alpha'_i \oplus k_{6,i}$. Here, the key K_6 = $k_{6,1}k_{6,2}k_{6,3}k_{6,4}k_{6,5}k_{6,6}k_{6,7}k_{6,8}$.
* S-box: Compute γ_i = $S_i(\beta_i)$ and γ'_i = $S_i(\beta'_i)$, such that $|\gamma_i| = |\gamma'_i|$ = 4.

At the end of these three operations, we know $\alpha_i, \alpha'_i, \beta_i \oplus \beta'_i, \gamma_i \oplus \gamma'_i$. Next, compute the set X_i = {(β, β') | $\beta \oplus \beta'$ = $\beta_i \oplus \beta'_i$ & $S_i(\beta) \oplus S_i(\beta')$}. Then, the keys k are found using the condition $\alpha_i \oplus k$ = β & $(\beta, \beta') \in X_i$ for some β'. For all these keys k that satisfied the condition, we incremented their count in the key matrix i.e., $keymatrix[i][k]$, where the key matrix is of dimension 8*64.

After performing the above analysis, we got the S-box results for the first characteristic as given below:

S-box	Max key frequency	Mean key frequency	Key
S1	85	62	1
S2	85	65	1
S3	80	59	48
S4	82	64	12
S5	83	65	6
S6	85	63	15
S7	77	60	45
S8	83	64	31

It can be seen from the results that the S-boxes

S_1, S_2, S_3, S_6, and S_8 have the maximum difference between max key frequency and mean key frequency values. Hence, their key values 1, 1, 48, 15, and 31 are taken for forming the key K.

Next, the S-box results for the second characteristic are as given below:

S-box	Max key frequency	Mean key frequency	Key
S1	150	70	45
S2	158	69	51
S3	134	67	37
S4	308	80	7
S5	166	68	23
S6	297	78	22
S7	121	67	2
S8	101	63	44

From the above table, we can infer from the results that the S-boxes S_1, S_2, S_4, S_5, and S_6 have the maximum difference between max key frequency and mean key frequency values. Hence, their key values 45, 51, 7, 23, and 22 respectively are taken for forming the key K.

Finally, from both the characteristic results we infer the key bit of S-boxes S_1, S_2, and S_6 i.e., are the same which verifies our results. The S-box $S_1, S_2, S_3, S_4, S_5, S_6$, and S_8 keys 45, 51, 37, 7, 23, 22, and 44 respectively are taken to get 42 bits of the 56-bit master key. The 42 bits of the key from S-boxes:

$101101110011100101000111010111010110XXXXXX1$

Step 3 - Finding the 56-bit master key

A key scheduling algorithm is applied to these 42 bits of the key to rearrange the bits into their actual position in the 56-bit key. Thus, we got the following key:

011X11X00101111001X11011000X10X101XX1X00001XX11XX100XX11

We know the 42 bits of the key and the remaining 14 unknown(X) bits of the key can be found using the brute force method i.e., going through all the 2^{14} possible keys. The correct key can be found by encrypting the plaintext = 'ffffffffffffffff' into ciphertext = 'oqfgjsltspnkpiik' using each possibility of the key. Finally, the correct 56-bit master key found is:

$01101110010111100111011000010010101100000110111$

Step 4 - Generation of all 6 round keys
The 48-bit round keys for all 6 rounds, derived from the master key are as follows:

Round 1
111011000100111100000111001011000110010100110100

Round 2
011011110011011101100010100001100101100011110001

Round 3
111010101101010011101101010000111010101101110001

Round 4
110110011100001101011010101100111000110100011000

Round 5
001001001101101110111011010010010001011100010110

Round 6
101101110011100101000111010111010110000010101100

Step 5 - Password decryption
The ciphertext corresponding to our password for this level is "$pmitsggknlsnfgfhmskumuomnltgirmu$". In order to obtain the decrypted password, we used the master key that we got from the above-mentioned procedure. There are 32 characters in the ciphertext with each character of 4 bits making it a 128-bit string, that is, 2 blocks of DES ciphertext. We obtained the mappings for the ciphertext by replacing letters f-u with 0-15 respectively and grouping two letters to make one byte. We used $stringtodec.py$ file for this. The ciphertext mapping is as follows: {167, 62, 209, 21, 134, 216, 1, 2, 125, 95, 127, 151, 134, 225, 60, 127}. Now, use the above mapping and decrypt.cpp, we perform decryption of the ciphertext considering 16 characters(=64 bits) at a time. The file $decrypt.cpp$ implements the decryption function of 6-round DES.
The plaintext we obtained as the output of $decrypt.cpp$ is $mttofyifmr000000$. The zeroes obtained in the password were removed as they might have been used as padding bits. Finally, by entering the plaintext "$mttofyifmr$" in the game, we were able to clear level 4.

Q5 Password
5 Points

What was the password used to clear this level?

mttofyifmr

Q6 Code
0 Points

Please add your code here. It is MANDATORY.

Assignment4-CS641 -force_de_fem.zip Download

```
1  Large file hidden. You can download it using the
   button above.
```

Assignment 4

Graded

Group

ASHEE JAIN
Kriti Majumdar
Srujana Sabbani
View or edit group

Total Points

40 / 100 pts

Question 1

Team Name — **0** / 0 pts

Question 2

Commands — **5** / 5 pts

Question 3

Cryptosystem — **10** / 10 pts

Question 4

Analysis — R **20** / 80 pts

Question 5

Password **5** / 5 pts

Question 6

Code R **0** / 0 pts