Q1 Teamname
0 Points

CryptoCreeks

Q2 Commands
10 Points

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

'enter'-> 'jump' -> 'dive' -> 'back' -> 'pull' ->'back' -> 'back' -> 'enter' -> 'wave' ->'back' ->'back' -> 'thrnxxtzy' ->'read' -> '360852885036840078603672 5' -> '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.)

From the starting screen, we saw that there was a door with a panel nearby, so we tried to read from the panel, but nothing

appeared, so we tried the enter command to enter to through the opening. There we saw a lake in which we tried to jump, we came out, so we jumped again. There we tried to pull the magic wand, but we died. So this time, we tried the following commands : $'enter'->' jump'->' dive'->' back'$ and then pulled the wand. Then we went back to the first screen using the $'back'$ command. Here we again tried to read the panel but was unable to do so. We then went back to the spirit in level 3, entered the chamber and waved the wand at the spirit to free it. We then went back to the first screen of level 3 and entered $'thrnxxtzy'->' read'->' 360852885036840078603672 5'$ to clear the level. We then reached the first screen of level 4, and used $read$ command and got the screen where we can get the corresponding ciphertext for a given plaintext. The spirit mentioned that the DES can be 4-round or 6-round, so we went ahead with 6-round DES first, which ultimately gave us the solution. It also mentioned that 2 letters for 1 byte has been used, this means that one letter consists of 4 bits. So we can have atmost 16 letters.

On whispering "password", we got the encrypted password - **"ourngluhfosjqkqskstmkphnjphfhluq"**. On entering any other text on the next screen, we got the corresponding ciphertext.

I. Generate input-output

Since we needed many plaintext-ciphertext pairs to perform the DES technique, we generated a set of plaintexts. We observed that the ciphertext consisted of 16 letters from $\{f....u\}$. So, we created a 1-1 mapping between the characters such that f corresponds to '0000', g to '0001' ..... u to '1111'. We generated the plaintexts in the similar form. These plaintexts were generated in pairs such that their xored value is equal to the reverse initial permutation of the respective characteristic. We used 2 characteristics (with probability $\frac{1}{16}$) $\Omega^1 = "4008000004000000_x"$ and $\Omega^2 = "0020000800000400_x"$, and for each characteristic , we generated 5000 input pairs which are stored in **plaintext.txt** and **plaintext1.txt** respectively. The pairs are such that the XOR of the 1^{st} and 2^{nd} text, 3^{rd} and 4^{th}

text..... etc. give the required reverse initial permutation of the respective characteristic. Each characteristic can help us in finding 30 bits of K6, but 3 SBoxes $S2, S5, \ and \ S6$ are common in both, so in total we can get 42 out of 56 bit key, and rest 14 bits can be found by brute force.

We automated the process of extracting the required ciphertexts using a function.This function, `generate_cipher_via_ssh()` runs the ssh command by taking an input text file (`in.txt`) that contains the commands as well as plaintexts. The output gets stored into a list , from which the function extracts the ciphertexts. The ciphertexts for each characteristic are stored in `ciphertext.txt` and `ciphertext1.txt` respectively.

II. Break 6 round DES (Differential Cryptanalysis

In this part, we used the ciphertext pairs, performed differential cryptanalysis on them, and generated the required key which was then used to decrypt the required password.

The mapping of $\{f....u\}$ was used to convert the ciphertext into the corresponding binary form of 64 bits. The binary ciphertext is then passed through the `apply_rev_final_permutation()` , which applies the reverse final permutation on the ciphertext, and then from that we get the values of L6 and R6 from the first and second halves respectively. The value of R5 will be equal to that of L6, which was then passed through the expansion box using the function `apply_expansion()`. We know that since $\alpha \oplus \alpha' = \beta \oplus \beta'$ (i.e. XOR of output of Expansion box is equal to XOR of input of Sbox) , so we took the pairwise XOR $\{(a_i \oplus a_{i+1}), (a_{i+2} \oplus a_{i+3}).....\}$of obtained output of expansion box to get the XORed input of Sbox.

As stated in `DES-differential-cryptanalysis.pdf` , for the 1st characteristic, five Sboxes in the fourth round $(S2, S5, ..., S8)$ have zero input XORs $(S'_{Ed} = 0)$ and thus their output XORs are zero $(S'_{Od} = 0)$ and for the 2nd characteristic, five Sboxes in the fourth round

$(S1, S2, S4, S5\ and\ S6)$ have zero input XORs. $R_4 = L_3 \oplus F(R_3)$, where $F(x)$ denotes the fiestal output of x. Also, $L_5 = R_4$, and since we don't know the actual value nor the XORed value of R_4, we cannot tell the actual value or the XORed value of L_5. But we know that output XORs of some Sboxes (as specified above in case of both the characteristics) equals zero. So we can make use of this fact to find 6-bit key values corresponding to those Sboxes in round 6.

The value of R_6 is then passed through the `apply_inv_permutation()` to apply the inverse permutation (since we do not know the exact value of L_5 but we can find key corresponding to some SBoxes as explained above.),then we calculated the pairwise XORed values at the output of Sbox at round 6.

Now , we created an array $keys[]$ of size 8x64 of zeros , such that row corresponds to Sbox and column represents the 6-bits of key of each Sbox.For each xored input Sbox value , we tried to find pairs (β, β') for each Sbox such that $X_i = \{(\beta, \beta')\ |\beta \oplus \beta' = \beta_i \oplus \beta'_i$ and $S_i(\beta) \oplus S_i(\beta') = \gamma_i \oplus \gamma'_i\}$.Now , for these (β, β') values , we can XOR any one of them with the corresponding output of the expansion box , to get a 6-bit key , which can be converted to integer and then we perform $keys[i][key] += 1$. By this we are trying to find how many times a particular 6-bit key value has occurred corresponding to each Sbox.We represented the unknown bits by **#**.

For each row of array $keys$, we found the most occurring 6-bit key value corresponding to the Sboxes $S2, S5, ..., S8$ in case of 1st characteristic and {$S1, S2, S4, S5\ and\ S6$} in case of 2nd characteristic .As per the $DES - differential - cryptanalysis.pdf$, we will now have 30 bits corresponding to the specified Sboxes for each characteristic.The computed key values of the common Sboxes $\{S2, S5, S6\}$, should be same in both the calculations, else we need to repeat the analysis with more pairs. We will merge the keybits obtained in both the calculations. The tables for each characteristic stating maximum frequency along with 6-bit key corresponding to the maximum frequency for each SBox (for the attached plaintext and ciphertext

files) are attached below.

For characteristic : 40 08 00 00 04 00 00 00

SBox	6-bit key	frequency
2	110011	1585
5	110110	788
6	001011	1519
7	000010	955
8	100011	957

For characteristic : 00 20 00 08 00 00 04 00

SBox	6-bit key	frequency
1	101101	799
2	110011	846
4	000111	1468
5	110110	829
6	001011	1498

The obtained 56 bit key with 42 bits known and 14 bits unknown is:
['#', '1', '1', '#', '#', '1', '#', '#', '0', '1', '0', '1', '1', '#', '1', '0', '0', '#', '#', '1', '1', '#', '1', '1', '0', '0', '0', '#', '1', '1', '0', '0', '1', '1', '0', '#', '0', '0', '1', '0', '0', '1', '1', '1', '#', '0', '1', '#', '0', '0', '1', '0', '#', '0', '0', '1']

So we now have got 42 bits out of the 56-bit key and for the rest 14 bits, we applied brute-force.
We obtained the final 56-bit key as:

```
01101110010111100111101100001100110100100111101(
```

For the above final 56 bit key, we got the following 48-bit keys for each round:

```
Round 1 key : 11101100010011110000011110001100001010
```

```
Round 2 key : 01101111001101110110001010110101010000
```

```
Round 3 key : 11101010110101001110110111001011000010
```

Round 4 key : 110110011100001101011010110101101110C

Round 5 key : 001001001101101110111011001100010001C

Round 6 key : 101101110011100101000111110110001011C

We then used this key to decrypt the encrypted password, and we got the following binary output:

011011100110110001110000011011010110100101100011011

00110101001110001001100000011000000110000001100000

We noticed that even when we gave the input consisting of the characters other than $f...u$, we still got some ciphertext for that. So, we concluded that our plaintext can consist of any character, so we tried to convert them using ASCII character codes. We got the decrypted password as:

nlpmicbjjq000000

We tried this password, but that did not get accepted, so we tried the same password without trailing $0's$ and that got accepted.
So the final password is :

nlpmicbjjq

NOTE: The code has been run using Jupyter Notebook

No files uploaded

Q5 Password
5 Points

What was the final command used to clear this level?

nlpmicbjjq

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.

▾ DES.zip Download

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

Assignment 4

GRADED

6 DAYS, 21 HOURS LATE

GROUP
PRADEEP KUMAR TIWARI
TANISHA RASTOGI
SHRUTI WASNIK
View or edit group

TOTAL POINTS
90 / 100 pts

QUESTION 1
Teamname **0** / 0 pts

QUESTION 2
Commands **0** / 10 pts

QUESTION 3

Cryptosystem **5** / 5 pts

QUESTION 4

Analysis **80** / 80 pts

QUESTION 5

Password **5** / 5 pts

QUESTION 6

Codes **0** / 0 pts