Q1 Team Name

0 Points

Enigma

Q2 Commands

5 Points

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

go, go, go, go, go, give, read.

Q3 Analysis

30 Points

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.)

1) After reaching the spirit we saw a sequence of numbers that were the hash values of our actual password which were, “23 100 15 94 25 9 45 105 40 55 104 16 108 85 94 107 41 19 87 36 93 115 49 116 114 49 55 96 23 116 92 84”.

2) We are told by the spirit, that the password contains letters from ‘f’ to ‘u’ and these letters are present in the password in alphabetic order. So now we had to decide whether repetitions were to be considered or not.

3) For hashing the password, the password was seen as a

sequence like $x_1, x_2, ..., x_m$ in the field F_{127} which means that all the letters of the password were mapped to numbers between 0 to 126 and there was a total to 'm' letters in the password.

4) The i_{th} number of the hashed sequence equals $x_1^{i-1} + x_2^{i-1} + ... + x_m^{i-1}$. The given equation is $(i-1)_{th}$ power symmetric polynomial mentioned in Newton's identities also known as the Girard–Newton formulae, give relations between two types of symmetric polynomials, namely between power sums and elementary symmetric polynomials. Evaluated at the roots of a monic polynomial P in one variable, they allow expressing the sums of the k_{th} powers of all roots of P (counted with their multiplicity) in terms of the coefficients of P, without actually finding those roots.

5) Also we are given a sequence of 32 numbers, this tells that the value of 'i' iterates from 1 to 32 i.e we will need to calculate 32 hash values for the correct password.

6) We know $h(i) = x_1^{i-1} + x_2^{i-1} + ... + x_m^{i-1}$
So first we consider the value of i =1
Doing this we got $h(1) = x_1^0 + x_2^0 + ... + x_m^0$
h(1) = 1 + 1 + ... + 1

And we know that the value of h(1) is 23 as per the given sequence therefore the value of 'm' is 23 which is also the number of letters in our password if one letter in the password is mapped to one number.

7) Now we know that our password length is 23 in which letters range from 'f' to 'u' giving us 16 characters. This shows that repetition of the alphabet in the password is allowed.
Now we just have to map alphabets range to numbers, so we did character to Ascii mapping f-'102',g-'103',.....u-'117', now possible values of x_i is in the range [102,117] in non-

how possible values of x_i is in the range [102,117] in non-decreasing order, in {Field 127}, it means that values should always lie between [0-126].

8) We know that 'i' ranges between 1 to 32 and $h(i) = x_1^{i-1} + x_2^{i-1} + ... + x_m^{i-1}$, so the h(i+1) should always be greater than h(i), given $x_1, x_2, \ldots, x_m \in [0, 126]$. But we observed that in the given sequence of hash values this was not holding true, which could indicate to only one thing, that every value is taken a mod127 while performing hash operation. Making the actual equation of h(i) as:

$$h(i) = (x_1^{i-1}\%127 + x_2^{i-1}\%127 + ... + x_m^{i-1}\%127)\%127$$

9) We applied a BRUTEFORCE attack to decrypt the password. After figuring out all the above information we generated all the possible combinations of 23 numbers from 102 to 117 in non-decreasing order and for every combination calculated h(1) to h(32) values till we didn't find the combination which gave all the correct hash values(matching the original sequence). The output we found was (102, 102, 103, 104, 104, 105, 106, 107, 108, 108, 108, 108, 109, 109, 111, 111, 114, 114, 114, 116, 116, 117, 117). This is the password's Ascii values.

10) Now, performing the reverse mapping of Ascii values to its representation in alphabets we got the password as "ffghhijkllllmmoorrrttuu".

No files uploaded

Q4 Password

15 Points

What was the final command used to clear this level?

ffghhijkllllmmoorrrttuu

Q5 Codes

0 Points

It is MANDATORY that you upload the codes used in the cryptanalysis. If you fail to do so, you will be given 0 for the entire assignment.

passcrack (1).ipynb Download

importing Libraries

In [1]:
```
from itertools import
combinations_with_replacement #printing
all the combinations with replacements i
a given range and length
letters = [ i for i in range(102,118)]
#producing list in range [102,118]
allcombo=
combinations_with_replacement(letters,23
```

In [2]:
```
import numpy as np
```

In [3]:
```
for i in allcombo:
    print(i)
    break
```

```
(102, 102, 102, 102, 102, 102, 102, 102,
```

defining helper function

In [4]:
```
def power2(x):
    return pow(x,2,127)  #efficient
way to do modulo
def power3(x):
    return pow(x,3,127)
def power4(x):
    return pow(x,4,127)
```

```
def power5(x):
    return pow(x,5,127)
def power6(x):
    return pow(x,6,127)
def power7(x):
    return pow(x,7,127)
def power8(x):
    return pow(x,8,127)
def power9(x):
    return pow(x,9,127)
def power10(x):
    return pow(x,10,127)
def power11(x):
    return pow(x,11,127)
def power12(x):
    return pow(x,12,127)
def power13(x):
    return pow(x,13,127)
def power14(x):
    return pow(x,14,127)
def power15(x):
    return pow(x,15,127)
def power16(x):
    return pow(x,16,127)
def power17(x):
    return pow(x,17,127)
def power18(x):
    return pow(x,18,127)
def power19(x):
    return pow(x,19,127)
def power20(x):
    return pow(x,20,127)
def power21(x):
    return pow(x,21,127)
def power22(x):
    return pow(x,22,127)
def power23(x):
    return pow(x,23,127)
def power24(x):
    return pow(x,24,127)
def power25(x):
    return pow(x,25,127)
def power26(x):
    return pow(x,26,127)
def power27(x):
    return pow(x,27,127)
def power28(x):
    return pow(x,28,127)
def power29(x):
    return pow(x,29,127)
def power30(x):
    return pow(x,30,127)
def power31(x):
    return pow(x,31,127)
```

equating 32 hash values into the symmetric polynomial equation

i = 1 to 32

$$x_1^{i-1} + x_2^{i-1} + ... + x_m^{i-1}$$

In [ ]:
```
condition = (sum(i)%127==100) and
(sum(map(power2,i))%127==15) and
(sum(map(power3,i))%127==94) and
(sum(map(power4,i))%127==25) and
(sum(map(power5,i))%127==9) and
(sum(map(power6,i))%127==45) and
(sum(map(power7,i))%127==105) and
(sum(map(power8,i))%127==40)  and
(sum(map(power9,i))%127==55) and
(sum(map(power10,i))%127==104) and
(sum(map(power11,i))%127==16) and
(sum(map(power12,i))%127==108) and
(sum(map(power13,i))%127==85) and
(sum(map(power14,i))%127==94) and
(sum(map(power15,i))%127==107) and
(sum(map(power16,i))%127==41) and
(sum(map(power17,i))%127==19) and
(sum(map(power18,i))%127==87) and
(sum(map(power19,i))%127==36) and
(sum(map(power20,i))%127==93) and
(sum(map(power21,i))%127==115)and
(sum(map(power22,i))%127==49) and
(sum(map(power23,i))%127==116) and
(sum(map(power24,i))%127==114) and
(sum(map(power25,i))%127==49) and
(sum(map(power26,i))%127==55) and
(sum(map(power27,i))%127==96) and
(sum(map(power28,i))%127==23) and
(sum(map(power29,i))%127==116) and
(sum(map(power30,i))%127==92) and
(sum(map(power31,i))%127==84)
```

In [52]:
```
searching = [ i for i in allcombo
if condition ]
```

```
(102, 102, 103, 104, 104, 105, 106, 107,
```

In [64]:
```
f="102, 102, 103, 104, 104, 105,
106, 107, 108, 108, 108, 108, 109,
109, 111, 111, 114, 114, 114, 116,
116, 117, 117"
password=f.split(', ')
li= [ int(i) for i in password]
li
```

Out [64]: [102

```
Out [64]:  [102,
            102,
            103,
            104,
            104,
            105,
            106,
            107,
            108,
            108,
            108,
            108,
            109,
            109,
            111,
            111,
            114,
            114,
            114,
            116,
            116,
            117,
            117]
```

Password (Ascii to character Mapping)

In [65]:
```
g=""
i=0
for i in li:
    g+=chr(i)
g
```

Out [65]:
```
'ffghhijkllllmmoorrrttuu'
```

Assignment 7

GRADED

GROUP

Kajal Sethi

Kajal Sethi
Gajender Sharma
Pranshu Sahijwani
View or edit group

TOTAL POINTS

50 / 50 pts

QUESTION 1

Team Name **0** / 0 pts

QUESTION 2

Commands **5** / 5 pts

QUESTION 3

Analysis **30** / 30 pts

QUESTION 4

Password **15** / 15 pts

QUESTION 5

Codes **0** / 0 pts