Q1 Team Name

0 Points

DECODERS

Q2 Commands

10 Points

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

exit1, exit3, exit4, exit4, exit1, exit3, exit4, exit1, exit3, exit2, read

Q3 Analysis

60 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.)

We entered exit2 command first to proceed as it was stated on the first screen that we came from exit 1. Despite this, we still could not progress to new screens so we returned to the beginning of the level and made our way out via exit 1. As a result, the hexadecimal text 59 6f 75 20 73 65 65 appeared on the screen. Only one command or numbered exit from each screen took us to a new screen with hexadecimal characters different from those previously encountered, while other exits led us to some previously encountered screen. In order to avoid encountering the same screen (or combination of hexadecimal characters) twice, we used a unique combination of exits as follows:

"59 6f 75 20 73 65 65 20 61 20 47 6f 6c 64 2d 42 75 67 20 69 6e

20 6f 6e 65 20 63 6f 72 6e 65 72 2e 20 49 74 20 69 73 20 74 68 65 20 6b 65 79 20 74 6f 20 61 20 74 72 65 61 73 75 72 65 20 66 6f 75 6e 64 20 62 79"

On the last exit, no exit was taking us to a different screen, hence we stopped taking exits from here.
We then converted these hexadecimal values to decimal values then after that we found ASCII values for these decimal values. This procedure is done by using "cipher_to_message.ipynb". The message found by this procedure is as follows:

"You see a Gold-Bug in one corner. It is the key to a treasure found by "

We tried the command "go" on the last screen to check if we can explore more, but this did not help. We then entered the command "read" and it gave us the following message:

"n =
84364443735725034864402554533826279174703893439763343343863260342756678609216895093779263028809246505955647572176682669445270008816481771701417554768871285020442403001649254405058303439906229201909599348669565697534331652019516409514800265887388539283381053937433496994442146419682027649079704982600857517093

DECODERS: This door has RSA encryption with exponent 5 and the password is
34171865506568761235366341493790588282544840980265985400868758217892290366974519206338316594138947784237705432193781201437528572525674984876315116214158247120512293095872879010604762678625568544043246569469048652272836435130693197728856983127258786909134464110413886711222381976246871928091620326451745412077"

We can easily decrypt it since RSA is used for encryption using:

$$dec(C, d, n) = C^d mod\ n$$

where d is unknown, C is ciphertext and n is modulus.

We must either find factors of n or determine the value of d to decrypt the password. We cannot find the factors of n since it is a large number. Additionally, because n is not factorizable , we cannot compute $\phi(n)$ and, therefore, we cannot calculate d. As the public exponent is 5, we could conduct a low-exponent RSA attack. We did exactly that using Coppersmith's algorithm and LLL Lattice Reduction Technique.

As a first step to applying the algorithm, we had to check if padding is used or not. For this we determined $C^{1/e}$, where e is the public exponent key. This came out to be non-integer. Thus we concluded that , padding was used. After this the new equation will be:

$$(P + M)^e \equiv C \; mod \; n$$

where,
M is the original message.
P is the padding message.
$e, C \; n$ are also known here.

$Coppersmith's Algorithm$:

Consider n is a positive integer and $f(x) = \; Z(x)$ is polynomial of degree d,
Given n and f , all integers x_0 such that $f(x_0) \equiv 0 \; mod \; n$ and $x_0 < n^{(1/d)-\epsilon}$ can be recovered in polynomial time , where, $(1/d) > \epsilon > 0$.[2]

This allows us to formulate our problem as :
$f(x) = (P + x)^e \; mod \; n.$
Here,
x denotes a polynomial ring of integers modulo n.
P is the padding used.
e is equal to 5.

For the padding P, we first used the message which we got from "cipher_to_message.ipynb", i.e. "You see a Gold-Bug in one

corner. It is the key to a treasure found by " but this did not give any root for $f(x)$. Hence we tried using the padding "DECODERS: This door has RSA encryption with exponent 5 and the password is " which we encountered on the last screen of this level and we found the root for $f(x)$ by performing the below operations using "root_generation.sage" which is taken and modified from [1].
We ran this sage code at https://sagecell.sagemath.org/

1. Padding P is converted to binary.
2. According to the assumption $x < n^{1/e}$, the length of password x cannot be longer than 200 bits.
3. As a consequence, the final polynomial becomes :
$((binary\ P << password\ length) + x)^e - C.$
Having variable password length from 1 to 200 in multiples of 4.

When password length was 80, the following root was obtained using Coppersmith's algorithm and LLL:
$$1000011001110000101100101010000001101110110111010$$

We added one 0 to the root to make it 80-bit, and finally the root became:
$$01000011\ 00111000\ 01011001\ 01010000\ 00110111\ 0110111$$

We took 8 bits at a time and mapped it to the the corresponding ASCII values, and thus we got the following 10 characters long password:
$$C8YP7oLo6Y$$

We used the code "root_to_password.ipynb" to convert the root to the above password.

$References$:
[1] https://github.com/mimoo/RSA-and-LLL-attacks/
[2] https://en.wikipedia.org/wiki/Coppersmith_method

No files uploaded

Q4 Password

10 Points

What was the final command used to clear this level?

C8YP7oLo6Y

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.

DECODERS.zip Download

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

Assignment 6

GRADED

GROUP

Akash Gajanan Panzade
Manthan Kojage
Abhishek Dnyaneshwar Revskar
View or edit group

TOTAL POINTS

80 / 80 pts

QUESTION 1

Team Name **0** / 0 pts

QUESTION 2

Commands **10** / 10 pts

QUESTION 3

Analysis **60** / 60 pts

QUESTION 4

Password **10** / 10 pts

QUESTION 5

Codes **0** / 0 pts