Q1 Team Name
0 Points

Lazarus

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.)

It was displayed on the first screen that we had entered from the exit1, and only exit2 was open, so we typed the exit2 command, and we got 20 61 20 47 6f 6c 64 these characters. On this screen, it gave us options from exit1 to exit4, and exit5 was closed. After randomly typing exit command, we observed that some commands gave us new characters sequences and some gave us previously seen character sequences. After a lot of hit and trial we reached a screen where none of the number exit commands worked so we tried other commands go enter but none worked finally we tried read command we got the values of N, C and E.

$N =$
$84364443735725034864402554533826279174703893439$
$34275667860921689509377926302880924650595564757$
$16481771701417554768871285020442403001649254405$
$95993486695656975343316520195164095148002658873$
$49699444214641968202764907970498260085751709 3$

$C =$
$17682209225809987046236362123205092817571021518$
$70072948215265400013715893408073990521685683709$
$06581569868080227308437728529982122166164583798$
$80739459372111880901370252804864618683832876421$
$39312169141905791185180371730846533787629266 1$

$E = 5$

Hexadecimal Sequence	Converted String
59,6f,75,20,73,65,65	You see
20,61,20,47,6f,6c,64	a Gold
2d,42,75,67,20,69,6e	-Bug in
20,6f,6e,65,20,63,6f	one co
72,6e,65,72,2e,20,49	rner. I
74,20,69,73,20,74,68	t is th
65,20,6b,65,79,20,74	e key t
6f,20,61,20,74,72,65	o a tre
61,73,75,72,65,20,66	asure f
6f,75,6e,64,20,62,79	ound by

The encryption system used was RSA with exponent 5 which was displayed on the window.
Now to decrypt the cipher either we need to calculate d which is not feasible as we cannot efficiently calculate $\phi(N)$ as N is very large or we need to factorize N and this to is not efficient as N is 308 digits which is 1023 bits.

Encryption : $C = M^e \pmod N$
Decryption : $M = C^d \pmod N$

As e is small we can use a low exponent attack (i.e. Coppersmith's algorithm coupled with LLL reduction algorithm) to recover the message. In this attack calculating d is not required.

Coppersmith's Algorithm:-
- First we checked whether $C^{\frac{1}{e}}$ is integer or not. If it is not an integer then some padding has been added to the original message. In our case, the result came out to be not an integer so some padding has been added.
Let M be the message and P be the padding then the encryption formula becomes

$$C = (P + M)^e \pmod N$$

- Coppersmith's Theorem states that given a polynomial f of degree d and integer N. We can compute all roots of polynomial f such that $f(x) = 0 \pmod N$ where all roots are less than $N^{\frac{1}{d}}$ in polynomial time.
So our problem can be stated as $f(M) = (P + M)^e \pmod N$

- First we converted padding to its binary form and then binary to integer and stored as "msgint".

- As we know that N is 1023 bits long and $x_0 < N^{\frac{1}{e}}$ and $N^{\frac{1}{e}}$ cannot be longer than 1023/5 which is approximately 200 bits long. Therefore x_0 is atmost 200 bits long.

- The final polynomial is $((msgint << length_m) + m)^e - c$.
- Roots of this polynomial is converted into string using ASCII code and that is the required password.
- Roots of this polynomial can be calculated using Coppersmith's algorithm and LLL reduction.

We first tried using "You see a Gold-Bug in one corner. It is the key to a treasure found by " as padding with parameters values

but we could not find the root. So we look for another possible padding and so we tried "Lazarus: This door has RSA encryption with exponent 5 and the password is " and we got the root.

Root:
1000011001110000101100101010000001101110110111101001100011011110011011001011001

To recover the message we need to convert the binary into ASCII code but the length of the root was 79 since each ASCII value is represented using 8 bits, we need to make length of the root into a multiple of 8, so we add 0 at the start.

Root:01000011001110000101100101010000001101110110111101001100011011110011011001011001

Starting from the left, we take 8 bits to convert them into decimal and then convert them to a character using ASCII codes and append the character to the password.

The password decrypted is "C8YP7oLo6Y".

References -

1. Referred code from this github link https://github.com/mimoo/RSA-and-LLL-attacks.

2. https://iacr.org/archive/eurocrypt2004/30270487/bivariate.pdf

3. 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.

Lazarus.zip Download

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

Assignment 6

GRADED

GROUP

Aditya Loth
Varun Vankudre
Harsh Agarwal

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

✔	**+ 10 pts**	**Padding:** *"{group_name}: This door has RSA encryption with exponent 5 and the password is "*
✔	**+ 15 pts**	Brief description of Coppersmith Attack
✔	**+ 5 pts**	Compute the upper bound on the length m
✔	**+ 10 pts**	Extract the final password by converting the root to ASCII
✔	**+ 20 pts**	Proper code implementation
	+ 0 pts	Incorrect or NA

QUESTION 4

Password

10 / 10 pts

QUESTION 5

Codes

0 / 0 pts