Q1 Team Name
0 Points

Pasta_Sandwich

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 used multiple exit commands like {exit3, exit4} to reach the final panel which said :
"You see the following written on the panel:

n =
84364443735725034864402554533826279174703893439763343343863260342756678609216895093779263028809246505955647572176682669445270008816481771701417554768871285020442403001649254405058303439906229201909599348669565697534331652019516409514800265887388539283381053937433496994442146419682027649079704982600857517093

$Pasta\_Sandwich$: This door has RSA encryption with exponent 5 and the password is
6996039713555737566830270753800707199242602001981859061404250414176491024151054344311278450234193863650998612357852121704911364237669203458171118374888295808002554682532468126028691809117877902354479405320000834593314386394727112702254140849571855171417338947831060704279731288265664395547219271305438413074"

The above text clearly says that the encryption algorithm used is RSA with exponent as 5.
We know that RSA is an asymmetric-key cryptographic algorithm which uses a key pair consisting of public and private keys.
Let, M be the message to be encrypted and C be the encrypted text.
Encryption:
$C = M^e mod n$, where e is the public key.
Decryption :
$M = C^d mod n$, where d is the private key.

Known paraemeters: n and e

In order to decrypt the given cipher text we either need to compute the prime factors of n, which is a tedious task to do as the value of n is huge or we need to find the value of d. Finding the value of d is also not possible as we are not able to compute $\phi(n)$ (To compute $\phi(n)$, we should be able to factorize n).

The public exponent is small, i.e. 5, so a low exponent RSA attack is feasible.
We compute $C^{1/e}$ which comes out to be 2.33 i.e. not an integer value which suggests a padding must have been added.
Therefore, the modified equation is :
$C = (P + M)^e$ mod n, where P is the Padding bits.
We figured out that there was a padding used by computing $C^{1/e}$. The next task was to determine what was used for padding. As the length of the padding wasn't known we used various lengths of padding by left shifting by 1 bit. We tried with the sequence of hex numbers that we determined while reaching the final panel, but that didn't help.
Then we tried with the string "$Pasta\_Sandwich$: This door has RSA encryption with exponent 5 and the password is" We converted each of the character of the string into its eight bit binary. We got a 79 bit root with the following padding. We padded one more zero to the root in different positions and then converted it into text to retrieve the password.
Positions were zeroes were added :
-Zero appended to the front
-Zero appended to the back

The second approach, gave us the password "#8YP7oLo6Y" while 1st gave us gibberish values. We tried with "#8YP7oLo6Y" but unfortunately it wasn't the password.

Then we changed the padding string to "$Pasta\_Sandwich$: This door has RSA encryption with exponent 5 and the password is " (Added a space) and repeated the steps as mentioned above. Adding a zero to the front of the root and then converting the same into text gave "C8YP7oLo6Y" which is the correct password.

Assignment 6 GRADED

GROUP
Ayush Sahni
Sharanya Saha
Manjyot Singh Nanra
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

Coppersmith Algortihm is used to make a low public exponent attack on RSA. We know that our exponent e = 5.
Coppersmith's theorem states that: Let N be an integer, and $f \in Z[x]$ be a monic polynomial of degree d. Set $X = N^{1/d-\epsilon}$ for some $\epsilon >= 0$. Then, given (N, f) an attacker can efficiently find all integers $|x0| < X$ satisfying f(x0) = 0 mod N. The running time is dominated by the time it takes to run the LLL algorithm on a latice of dimension O(w) with $w = min(1/\epsilon, log_2 N)$
Now for our problem, we have $f(x) = (P+x)^e mod N$. Here x is polynomial ring over modulo N, value of e is 5 and P is padding. Solving for roots of f(x) will give us the password.
We referred https://www.cryptologie.net/article/222/implementation-of-coppersmith-attack-rsa-attack-using-lattice-reductions/ blog, which contained an implementation of coppersmith- Howgrave-Graham attack.
Now we dont know length of the password, but according to the theorem, we know roots $|x0| < X$, and value of X is $3.84.. * 10^61$, and taking log base 2 of this number, we get value = 204. Thus our password will be less than 204 bits.
Thus since length of password is less than 204 bits, in our polynomial equation, we try padding values by left shifting it, so we will left shift it at most 204 times.
To make this work for our problem we had to first convert the padding into binary form with each character converted to 8 bits.
So the final equation for polynomial becomes : $((binaryPadding << leftShiftValue) + x)^e - C$ where root length is between 1 to 204.
Therefore, if we find the roots of the following polynomial Voila!! We have the password.
Using the algorithm we found the root to be :
1000011001110000101100101010000001101110110111101001100011011110011011001011001
The length of root is 79 and we appended 0 on the MSB to get the password.

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.

Pasta_Sandwich.zip Download

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