Q1 Team Name
0 Points

Cryptophilic

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.)

First of all, we were looking for a meaningful pattern formed by the hexadecimal shown on the screen while typing the different exits. After several random tries, we got a meaningful string 'You see a Gold-Bug in one corner. It is the key to a treasure found by' from the hexadecimal : (using our code **hex_to_str.cpp**)
"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" by using the following sequence of commands :

exit1 -> exit3 -> exit4 -> exit4 -> exit1 -> exit3 -> exit4 -> exit1 -> exit3 -> exit2 ,
followed by 'read' to reach the ciphertext.

After reaching the final window, we were given the integer 'n', the encoded password 'C' and exponent e :

N =
8436444373572503486440255453382627917470389343976334334386326034275667
8609216895093779263028809246505955647572176682669445270008816481771701417554768871285020442403001649254405058303439906229201909599348669565697
5343316520195164095148002658873885392833810539374334969944421464196820276490797049826008575170 93

C =
1054261731681260611700938856143544706244944887499106979941764305273456392
4031819201792287582487321550335693586508146334904410099802254012366347172805343852300263016411959723628545502071706540767910888672400082308192725794467962196524589021851991962347154733264651228023278822054009453882
3473219746567541532

e = 5

Decryption -

If the message to be encrypted is ‘M’, and encoded message is ‘C’, then RSA encryption is given as:
o For encryption: $C = M^e$ (mod N)
o For decryption: $M = C^d$ (mod N)

Now to decrypt the password we need ‘d’ but it can be too long to guess. Also ‘N’ is very large so we cannot factorize it. Since e is very small (= 5), we can decrypt the message even without knowing ‘d’ using the **low exponent attack**.

Therefore, we used **Coppersmith’s algorithm** (which is based on lattice reduction for low exponent RSA) to decrypt the password. We needed to check if padding is required or not. For this we calculated $C^{1/e}$: if this was an integer then padding was not needed. But this is not the case in our problem, so we require a padding.
We considered the padding to be ‘p’.
Then : $(M + p)^e = C$ (mod N).

Coppersmith’s Theorem :

Let N be an integer and f be a polynomial of degree δ. Given N and f, one can recover in polynomial time all x_0 such that $f(x_0) = 0$ mod N, and $x_0 < N^{1/\delta}$

So, the polynomial will be written as $f(M) = (M + p)^e$ mod N.

To decode the message, we referred to the following github repo and wrote a sage code for 'RSA low exponent attack' with its help (our code **Assn6.sage** is attached in the zip folder in answer to Q5) :
https://github.com/mimoo/RSA-and-LLL-attacks

According to Coppersmith's theorem, $x_0 < N^{1/e}$. On calculating this, we get $N^{1/e} \approx 10^{60}$, which means that the maximum length of message will be approximately 200 bits.

To break the RSA we needed the padding string ‘p’. Since at last we need to convert binary to character, so we looked for a padding which would yield us the length of the binary form

Assignment 6 ● GRADED

GROUP
Shubhi Kesarwani
Tanishq Rajesh Chourishi
Abhinav Maheshwari
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

of password to be multiple of 8, as ASCII characters are stored in 8 bits each with leftmost bit equal to 0.

We considered different paddings which could lead us to a binary password as discussed above -
(1) p = "You see a Gold-Bug in one corner. It is the key to a treasure found by"
(the string we got after converting the hexadecimal to text)
(2) p = "Cryptophilic: This door has RSA encryption with exponent 5 and the password is "
(written on the final window before the encrypted password)

and their different variants...

Finally, after several premutations and combinations, the padding string which gave us the desired binary length of 79 (since the leftmost bit would be 0, so we got the desired length of 79 + 1 = 80) was the (2) one :
“Cryptophilic: This door has RSA encryption with exponent 5 \newline and the password is ”

For the analysis we first converted the padding string to binary using
-> binary_padding = ''.join(['{0:08b}'.format(ord(x)) for x in padding])

Then, for all lengths of M (encoded message) given by $x_0 < N_{1/e}$
(which gave max length to be 200 bits), we calculated the polynomial as :

$$Polynomial = ((binary\_padding << message\_length) + M)^e - C$$

Finally, we used Coppersmith's algorithm to calculate the **roots of the polynomial** which gave us the password in binary form. We converted it from ASCII representation to char and then we outputted 'm' and the password. The final password came out to be :

C8YP7oLo6Y.

References :

1) https://github.com/mimoo/RSA-and-LLL-attacks
2) Lectures on 'Lattices in Cryptography' from Georgia Tech
(attached in a separate folder 'References' uploaded in the zip file in Q5)

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.

▾ Cryptophilic.zip Download

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

Select a question.

Group Members Submission History Next Question