Homework #4

Due date: **18 December 2022**

Notes:

- Note that there are five attached files: "RSA_Oracle_client.py" for Question 1, "RSA_OAEP.py" for Question 2, "ElGamal.py" for Questions 3 & 4 and "rainbow_table.py" and "rainbowtable.txt" for Question 5.
- Print out your numerical results in integer format, without "-e". (We do not want to see results like 1.2312312341324523e+24).
- Winzip your programs and add a readme.txt document (**if necessary**) to explain the programs and how to use them.
- Name your **Winzip** file as "cs411_507_hw04_yourname.zip"
- Create a PDF document explaining your solutions briefly (a couple of sentences/equations for each question). Also, include your numerical answers (numbers that you are expected to find). Explanations must match source files. Please also add the same explanations as comments and explanatory output.

1. (**20 pts**) Consider a deterministic RSA Oracle that is implemented at the server "http://10.92.55.4:6000". Connect to the server using the *RSA_Oracle_Get()* function, and it will send a ciphertext **"**C" **,** modulus **"**N", and public key **"e".**

- You are expected to find out the corresponding plaintext "m". You can query the RSA Oracle with any ciphertext $\overline{c} \neq c$ using the python function *RSA_Oracle_Query()*, and it will send the corresponding plaintext $\overline{m}$. You can send as many queries as you want as long as $\overline{C} \neq C$**.** Then, check your answer using *RSA_Oracle_Checker()*
- You can use the Python code RSA_Oracle_client.py to communicate with the server.

Important Note: You have to find a mathematical way to find the message "m". Once you find it, code it then check your answer. Querying the server blindly won't get you the right answer.

2. (**20 pts**) Consider the RSA OAEP implementation given in the file "RSA_OAEP.py", in which the random number R is an 8-bit unsigned integer. I used the following parameters for encryption:

ciphertext (**c**) =
10874572375620617789377153154263475798901864318895755165739361956409713948425
public key (**e**) = 65537
modulus (**N**) =
39011863995815647013266848060295512705184137160777355248310252490843225091289

I selected a random four-decimal digit PIN and encrypted it using RSA. Your mission is to find the randomly chosen PIN.

3. (**15 pts**) Consider the ElGamal encryption algorithm implemented in the file "ElGamal.py", which contains a flaw. We used this implementation to encrypt a message using the following parameters:

q = 151495026364772303137088254449581723810624208326119092779676949241 41

p =
171718105075276118274598889704825582800497596295907934721505597237658487113838350492459376465499393979662596416451032255319304216456084312430422988938972444666526144514216866190532109004247724680139502248165179522383857411329391879224520189847034534112880123298864917584089941792749451330945257187043448725163201740366151584011631029651898117055768816362957316693711888375676950687240314967296293259395915441887006587627445736218810133827848123271261308896539946193687466155686433855109999946425835614802348189515546592003895613191737220505209001280184109030499158271474596380606306035492321197130082925734374334534971650411262608683430756255655791151680148243618979698091756943767880225842362780653108888018279860395813593437445421511546422450162022642747925383435933592756710314126111136975695580605259079148292038044255708425837618072280519511078502071266251702163606248284827166513084142128618317809947279340210471761097721467980749185743693750395466097749993300733121543406039704618338149950629698745103870615092975062601561887370460417072914139610489463158418975021909830441659491169656483985783643811024154247645895751601948480351504264896332094236936412612431357609177554305285800308148879921585692026646113562498503806874318 5

public key (h) =
133738483737273040740995738721861248951611170245608037031779014875627872897573708459103898544695469581235976431164577015861438046257239451690951611546923146667028321324791343488839857918408844248228478827376017680242404588627804354954096880872380944975787024487532718019603976868890785163470683308948322345771339495882835771266984115292249801985636284071177242249853402102730290216862186164276262360307183998983185674479983210867236119107722156611461858991941563986347871101765010689544263287145624570602590315216084269427715686677018328443653117706640348843915306802178660607737994632475729873746024133926454526136 40

144567103119627180304101971994665090794489890193574241345628502628652199408779233388816942828648434257460644626506323604151334383753506018126066682418102617777994406459003384865107538075587771611036550416944875543450064240089041596421259607748082976461301064066032475153360385068064863314804124679983423871349361671859752839124503218

t =

219797278117801716244808775509686442389328113551136694738323351501955248547819668953931225593097251025109297577662745346022041290548338865519391240280077428989987283539048731688312589885841222996426195917602190629041975255270391054887169826613827629997131792031249257443219336078567657442198939766798664067508917665443260370491499590584900913900071729679396892277016877923649319632123379734565631286496085104823047980924930299126319159284697409608679671176571989977647906559552435039733460762948852653149229376885570373881284278928955770440410461633709314225496884902626988287771853109576076808689758788807918785694

Can you find the message?

4. (**15 pts**) We encrypted two messages, m_1 and m_2, using the ElGamal encryption algorithm given in "ElGamal.py", however, it contained a flaw, and we lost m_2.

q = 126756382935791072119261053234924095790569582470 1

p =

154474724567024505552326668667624254530346668891516550168600112845390987269072109602304547293643022259327026218001120791870575104603071951983783939178506546068762532687120696273925570767784225223472959435756693935028851519062765262076784037127804385249530791458749773813851085918817576722179601544649985814629

g =

103909996904124632993131371858766165669729607982465958461739488117164223912656915892337662801494883198703452648194921406170215661998544589380930634851778368778250267832833231632014974502522850108470857947473599059781490699897828182798112413851433613933100792383641135076229528835761463113619032894070860192807

$\left(message_1,\ r_1,\ t_1\right)$ = (b'This is my last message to you',

138463133012013282634913641892758158801762848592172407698472135927781184457014021159308423563284158903125454277831143312262376170389801812594838453962655356699229091103625269373494506043148105103719155747470849279981239429070325302155209235130687521331639091969580507394146320441457615197051837733672345300228,

106265595589369404671099487302170469354962925840925742209713793365234534431026758828912055935037813410518539768824133366973085636520219854456990152125504837648179032393648480733938344778228349463627323405812952123710763683733706558368525987706254967419548469786325235631013665472833542539156237682564045597699)

$\left(message_2,\ r_2,\ t_2\right)$ = (b'????????????????????????????????????',
1384631330120132826349136418927581588017628485921724076984721359277811844570140211593084235632841589031254542778311433122623761703898018125948384539626553566992290911036252693734945060431481051037191557474708492799812394290703253021552092351306875213316390919695805073941463204414576151970518377336723453002 28,
1355630594875829751215660880628984944751752550035600716287100464626304422626361346892397849509890517600618806866598052800942577227179045053035350346976770806510927883184866819674887752623400370220203639302124236664242809596176847675627667002690124863528374303127593020456307774528352072364871344842589127108 65)

Can you recover m_2 using the given settings? If yes, demonstrate your work.

5. (**30 pts**) Consider ten digests in the attached file "rainbow_table.py", each of which is the hash of a six-character password. Your mission is to find those passwords using the rainbow table given in the attached file "rainbowtable.txt". Complete and submit the Python code in the file "rainbow_table.py" such that it finds and prints out the ten passwords corresponding to the digests.