Homework #3

Due date: 24 November 2023

Notes:

- For Question 5, you can use a Python module for arithmetic in $GF(2^8)$.
- You are expected to submit your answer document as well as the Python codes you used.
- Brute-forcing the solutions (trying to query all possible solutions) in a server-related question would not be considered a valid answer and will result in a score of 0 for the respective question.
- Do not submit .ipynb files, **only .py** scripts will be considered. You can work on Colab but please, submit a Python file in the end.
- Zip 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_hw03_yourname.zip"**

1. **(15 pts)** You are in a job interview, and you were given the following RSA parameters:
   N=
   2236578597661440233239751207512463082703496247483496750837699259187895034976146624899681362879032402174196764503614296944218715118439577009702182357594984436148333180929317653472169407133463599682795180753291679587362965545437286352461389292512087804466755671308918725627762040088094267449215491946365907708865695556836507394680104113645504915425469409338668871846196657118091769117800177888164137017835629189950811077544939787334230775085954832721576113160444529991439658194193935722559406802470156301977446603456884493329985847799775443583165975769943149863223322845804100465136003298924864661227031083743164223408532372817748489413136158368142555560015660721756239734932975912943353490875696080566103630439108934583450923383815434902213347700326633618836715343980988824339498954402105493313843332777747643519052910477812997665261704636288767786189581643268233582740187870594458378220574108448150639055908605373565614031125387128925135332916959736300899479885278863339911134505004369987817266351494010006365406145357111594264108281794157598294395557040240375375819143849361636145285684083548679457057970518221847924922013282242877805937507851590516843339593375327842430003170696068876068797437618214033096770832778285149442430541190091005047121996885939755682215133067231568840002327571338547563667337570541846089983283137179334670718219827924473532685861642794873504118832531828344030201577499982915225832296328049335695098728270948950169409438864123210940690429120671846542914524472716071233495114881710945388955106707869551993028202013217431570700808781141028641836700217121954352851945714621142371128880855259924062736762516185103147629591130984034378758720000552980487640428893696243003722258845149374306820343431779585768457822338073230946486265571241260689828122599899301541810972769459376914733384588437545684656515385087580733584751932 3533

C=
10996907317744048201180239191184801878870026359954251163808786991823988
39204072169495677454061351187067291518696767449734569538264543011182408
38003555705364085507482373224955566405958979509042669782559203004942060
63326443644038077060867859678853573135463689576016437189518186799433888
57540103021087567684524465658391113407839387642187763991165704754307039
54649878421704627993437675253932151916097740731902858067340173902708102
57697704746005511384590358282243168705497528675364827201498077988713907
30269607077902128491358540075672275189364330649762937864320632331055818
92480718317178083599555304489864818867493161753564243757278735971734662
77727860357595136

e = 2^4+1

You are asked to retrieve the plaintext "M" using only these given parameters, the plaintext is a 289-bit number and M << N (which means that M is too small than N).
Show your work.

2. (**20 pts**) Alice encrypts the private factors of the modulus using her public key. In order to increase security, she multiplies them with a random integer *k* (a process called blinding). Namely, she performs the following operations:

$c_p = (kp)^e \bmod n$ and $c_q = (kq)^e \bmod n$,

where:
n =
19183611594110704047944268098409679747777693533574914342013275888624116
34257797931230340362144109492849084860870935003922754310066663633684520
45717449925141244415163916556097303659155135064640066981760746153509294
67995146954635808899790086506055221316988321349606090745264195046707737
12974549258713219266086529801320700401681765125510257808966626173920533
48880495866486863856323591225962712524437746354453515200460749082471888
17066447525613684763393056337195755222389664186394854951462626817698807
21615070148094756692833171503752074705662188239798689102120997485064799
57592158602667131927181846125509231614343194096131658354647309923744975
54093731804089182916057801363331096233954143618550941403351980468446272
44986704201974351462960689505220298363091032183218874210416417719748655
05022657856438960391003887024801890732463761889931885281594686399919481
95126451096885004465294596624026184302367656235248167994715642845445947
38244329827946603609406266452608971474023507335797934547214673666130099
21843929894186517343883761488835334583725774585133982979515445799264349
07745133866160543325682099182609661124258828686229066139373979701887434
9737259094859390154397508474946005962187441191496287970898124145277567
15416006852063181232015540335657245930749321809072404240556207462634556
07699700524768075354406018616956119182772903779733007050892970918375262
17352732496319085314920708082969122976857702438125484641488846223006033

33609697539815317426726928647381987950131331715044601820176825782919092
77355683737362686615312733563660679559873862037089763914089959724933586
05228390683894692756825505389326726472693267858287411509970899277244713
14364367502593122376235668982872064691932255585246865387923606731047553
13300203837226598476214001721836367864196351049976579073142410364805784
05353865022554985486575064306462132741635256070685397034325118442921285
4679
e = 65537

a. (**10 pts**) Explain why this is not secure as anyone who obtains c_p or c_q can factor n.

b. (**10 pts**) Factor *n* assuming

c_p =
851981557817028663180501789609647407567201542089841455627327356398545
828618487300888192159283842112619538927550062944039231761324497687980
388057286117564517137213293019868685782639869702929608521619592268535
229492028029634820515189201256121144377516210646924865432305711831525
008988947515797126410611036738460263123054867022095937573549240340925
254091916638534390288411432539369888827553476153519678858720450565692
176315412675291771897611394319368108029911640959591350150421488906242
934415514983630107918903946261995406986686430038313043890273679453834
690423843286897558114707250513165137877470596586573123707812665592592
403567834932132278135356380337936297100779430354167187083997494595094
626591016495726463422933061481013504294768979677472371131930976861335
597522841734695274312040622145269057593028252460451835507800545385424
502204384514998189188158541525373820209794737665646233760704396689189
013105691772177688370635231713069776525934039464735163426253774229903
564605724843628061985053109841452458158235245878493737946522700803668
031486833088641375203452949089151179418388706769255178293828545447152
214805300142824095390634151626234438344708486656313860224440149446722
221773277601414826225728090233257030707578815037615238171938849710818
800602540059212528045709993461890163651551768045405513467695102682020
172652244680414719946537811168638190661135178875524417373837188607629
6501484373607288892622240823128194832387029814711445604226771482932408
669331982883441090083879922610487695221496282020974515170482237913623
702763753281683261062144500298216845651293952563391167113383475580475
004579290031046918313469746663676387276717946185669112899532246928548
744793878632505301548248953025475737831739558711532898642671658507699
806421088325882846474494127348401007338603755298718246577185430053697
3027299524764873622012478186053079729537911008670556589

c_q =
481840298664363604495284312982970592272598829909770503091840104133429
708387402669960589564882745326799433253339913828385661768902915590951
047163160952994682086860237519873938678640684683675476253340298611644
451471067970735870998171972254926788874546499007912960592213046678850
271235001904569375670359398880187061346879629543066714462075142812804
693377981661541795907941297815977594271133066667709141027445297857502

163529394138183404237144169663527133174773536159137945022177934890635
861519276630746405021785100085199481432235286728710838680038183325580
823073100104889260912835689314010992269224282118532472630063169032809
486043104170975122972336150648885074336979724273438107955433290844704
554774601124399273754634264200434805982471716346751564474543154143168
010147841403678223200780563122421700652314574052373543883682988262626
055565406514565544785499920937894141761629329305262232749160465735783
073148132620587236630188290626418599438951025378390234248538603599305
890996491285365606950360984155080103977727943391938516943054066444068
240777914846080186389778286759908543916864954350522120137481870567787
718067714505896960150074352845990208577546227332683220614011378153697
204324467228191746838325461109101500590338511292274754440316602536200
817073720120287920450412264820261886896726850414012135036569685968452
862865401583341945821578694132085707023896197865214356700952282608050
844563374699708268398347506061814891641873337818396723512965428682703
342893493064538194299786321958056522054911165462229038429380301978454
562149441981973005359813664296944638360005703922303896239027526410477
488555244662402772070807508484838169841655834967849210103607268128340
331066841660547762499024738708078521397110907145752942416867861875208
265140255303922784980951983417127468195297494833725856801833466986306
5908317748964814977376118579837622866095503768197661413

and decrypt the following ciphertext

c_m =

172848288211416685603282354932360557971304059670763682199383435963868
500148706386534697210638390065530228267133581948041498904505178993814
709246745819492418376689358753522918376523234014124949810716237794880
580694971790977763648689436017132643101332471248541108959265828346683
832844081458155545765675070124650186739032982423587127281455778587634
781124790027712594935258076846574264513386531287376117948606657102686
393038586248002979140492947693573131113900731438147478193840830771683
963251800718979780571443537325408508047240097427436806820452665019208
185532305920139291552495894763989335439835817719740957265732107570025
545100042355203025131498999148945291892978898093833012754835506941859
064119833009544011419051103543206791596224369984223100745495349443635
647092809400249940376709338464949084207198123184305476857855749352971
821224837400061554821364342399132173772322306047721098584729481043319
178789341922232042370548196035735694344887475793552722848670366636796
039898579010488098662015401043739129264362079729625258989826963376632
855021468752069298038092356506751963797471127681548557726163971959183
308048398541583396562719850200716894128921616843291792231224770771296
188859608117033536180990203048365360571984059429934849548930023677107
432522510532345030063725400909468322747914183018679157609553010222962
120361168898245945104842943025306480915927643841940595576576345784412
927187873442141610419140468955306683198059500031909617696646193211695
824296022734313413912356798844029355290791743757022938354148262978207
948086266397804325039832972221950838870499908928290666475485857553146

2514889313482491646255044031348119733602661865586633091791695502190691818488605967807302727318082842483243701340921660030815554124647719995117645208273368037359335651906737986038473902686977967049445908642512694650162097605982512146831017115578920338711896567958 8

3. (**15 pts**) Consider the combining function given in the following table, that is used to combine the outputs of four **maximum-length** LFSR sequences:

   $F(x_1, x_2, x_3, x_4) = x_4 \oplus x_1x_4 \oplus x_1x_2x_4 \oplus x_1x_2x_3x_4$.

   Analyze the function F in terms of three criteria:
   - Nonlinearity degree
   - Balance
   - Correlation

   Is this a good combining function? Explain your answer.

4. (**15 pts**) We challenge you to get the plaintext of a ciphertext C that was calculated using an RSA setting, however, we lost the decryption keys, we only have the following:

   N = 9244432371785620259
   C = 655985469758642450
   e = 2^16+1

   (RSA Encryption: m^e mod N | Decryption: C^d mod N)
   Can you retrieve the message using only this information? If yes, show how.
   - You are not allowed to use external tools (including online tools).

5. (**15 pts**) Consider $GF(2^8)$ used in AES with the irreducible polynomial $p(x) = x^8+x^7+x^6+x+1$. You are expected to query the server using *get_poly()* function which will send you two binary polynomials a(x) and b(x) in $GF(2^8)$. Polynomials are expressed as bit strings of their coefficients. For example, p(x) is expressed as '111000011'. You can use the Python code "**client.py**" given in the assignment package to communicate with the server.

   a. (**7.5 pts**) You are expected to perform $c(x) = a(x) \times b(x)$ in $GF(2^8)$ and return c(x) as bit string using *check_mult()* function.

   b. (**7.5 pts**) You are expected to compute the multiplicative inverse of a(x) in $GF(2^8)$ and return $a^{-1}(x)$ using *check_inv()* function.

6. (**20 pts**) We want to perform modular multiplication for the three values given below (i.e., $a_i \times b_i \, mod \, q_i = r_i$).

$a_1 = 2700926558$
$b_1 = 967358719$
$q_1 = 3736942861$

$a_2 = 1759062776$
$b_2 = 1106845162$
$q_2 = 3105999989$

$a_3 = 2333074535$
$b_3 = 2468838480$
$q_3 = 2681377229$

However, instead of performing three different modular multiplications to calculate the results r_1, r_2, r_3 for these values, we want to perform only one multiplication operation modulo Q where $Q = \prod_{i=1}^{3} q_i$ and get a result R. Utilizing the Chinese Remainder Theorem techniques discussed in the lectures, show that you can reconstruct the results of the three operations (r_1, r_2, r_3) from R.