Dartmouth College Department of Mathematics
Math 75 Cryptography
Spring 2022

Final Cipher Challenge (upload to Canvas by Tuesday, June 7, 7 pm EDT)

For each of the following ciphertexts, your job is to discover the plaintext. In several cases, the plaintexts of earlier ciphertexts contain clues for later ones.

Show your work! There is no need to be laboriously detailed, but you must clearly indicate your method of attack and the steps you followed. Very little credit will be given for a plaintext with little indication of your decryption method!

If you use any computational resources, attach/upload all code and the output of your code runs. Please also explain in words (or in comments) what your code is doing. Feel free to use `Sage CoCalc`.

Don't overlook the text file `final.sage`, which contains relevant ciphertexts and some `python/sage` code that will be necessary for several decryptions.

You are free to (re)use any code, algorithm, or method from classwork or homework, including (hint! hint!) posted problem set solutions, and you may use any programming language you like. However, the rules for cooperation are completely different than for the homework. You may not work with anyone else. No communication about the exam is permitted with anyone except the instructor. In particular, do not give away solutions and do not share code that you've created since working on the exam. The output of web apps (except in the case of an Enigma simulator) will not count for credit unless you know (and can explain) what the app is doing. It is better to use or develop your own code. Exceptions to this rule may be allowed on a case-by-case basis, so contact me first! If you consult with or use any resource other than the textbooks or classwork, state this clearly.

If you are stuck on one of the ciphertexts, please send me an e-mail! I will be happy to help. Have fun and good hunting!

Problems:

1. *Substitution cipher.*

```
ovdkljoodkbplabzgufvdklacvqlkbzvwljjvgvlybzvzujjqlkbzvjlyvsuyvdtl
fvzvkdyobzvjlrdjlyvcizljlfvzvkgufvdklacvgufvdklacviubzdiujjqlkbzv
clyclqljoodkbplabzqlkbzvodagzbvkclqodkbplabzbzlagzklayobzvgukojvo
vdkbzbzvrkldpzvkcsvjjlybzvpkvpduycbzvrzdfvbzvcbujjylkbzuybzvukzvd
kbcbzvzujjiuyocuybzvukfvuycdyobzvgkdyubvlqyvizdpsczukvuybzvukpacw
jvcdyobzvuktkduycdzuybqlkwuszvkcuhucbzvlkovklqqufvploajlsuccpdjj
```

2. *Vigenère cipher.*

```
hcbxpcjlemyzlgjwagtfjhtnvvriarrqzvuqbipjrqhggrzwtfnahgkqfesrqszv
odyabgcwafvvrotsotdreoaqnbnfzgcbqetqloafvvnpapnqvzrzvyarnrpzgoas
hycwzvvwaphmbssvvhyammusjhnsfwnbbhbshhuwtkjylhrangemwsjnvprdatnr
pshseanrumabcbbphcacygjogiainojnvvbsdmhcbqgtvjeyloavjoianmrrln
```

3. *Affine cipher.* Bob, a captured enemy agent, was carrying the following message:

```
6917141364293641, 5044493105177484, 10208794241351887, 16394322558427148,
11758121930809893, 15571898457877977, 7672722015089403, 13661070158473411,
17999297470735005, 12313955920676335, 5960590266677512, 1613421779734456,
1750819096862416, 3118598423638319, 14816640742963862, 4952241931583899,
12257144082730227, 7862771476786858, 5006500927265261, 11323114722137903,
22833602100630408, 8963415721169565, 15595638667025459, 8028339051359388,
3385708046121353, 12190779082257523, 8983375210790796, 15571898457877977,
15147654701575566, 16361132341028484, 5962327355151718, 8901193427034701,
5179568152435730, 3672045789372412, 23610469115026974, 1577294047287513,
15642317927380556, 15571898457877977, 10282634434851196, 10749617216933305,
17838746455253440, 21499666401460178, 1037344909841996, 17413814796435480,
16269186929768054, 10449344135634668, 24087490685235750, 10768725190149571,
6484888204271905, 22185358129776042, 19377417029468988, 16267449841293848,
16555381474675390, 21520574190817628, 14140526597210259, 19733309797334806,
16283129124025650, 16538093542757166, 24098448719654436, 16798515250649044,
13879801264995293, 10264930014031131, 7946076055449771, 18258106201941864,
423054714981679, 17458353983971638, 9294184051519018, 19030921054252445
```

Using enhanced mind-reading techniques, Eve was able to able to extract the following information: Alice uses a general affine cipher, and the plaintext alphabet consists of blocks of five letters written as ASCII bytes (extended ASCII) and then interpreted as an integer modulo n. Apparently, even the coefficients of the affine cipher transformation are ASCII byte encodings of important codewords. (See `strtoint` and `inttostr` in `final.sage` for the precise encoding used.) The first part of the corresponding plaintext was also recovered:

```
314077111660, 464400513312, 495875089509
```

Unfortunately, Bob's mind went dark and he could not disclose n.

4. *Enigma.* Captain! We managed to get partial information about the daily settings.

```
Walzenlage: I II IV
Ringstellung: ?? ?? ??
Steckerverbindungen: ST AX UV FQ BM OP WY CD ?? ??
Kenngruppen: QZE TRF IOU TGB

SOP HIE = IOUTO XLIVE QVUAN MMGNC OMOUU GIHWR UKVIZ KBRQK IPIJU
          BWBTO ZHFNT BBZEU KCFRT IXOHJ AMKOE POYFV UFUQF ZTNGO
          LWAQK DQTVG INUFT NPZQH VMHCQ DVIDV GVLZA SNSOK FQD
```

If we could get the ring settings, the next tent over says we should be able figure out the rest of the plugboard.

5. *RSA*

```
alice> heya bob, how ar u
bob> im gr8
bob> you reT?
bob> n = 249073634649210472172972256733507625752814649331677815698197 43
         736316221493677086426330129288565212064717326462822433739607 31
alice> soo kewl that we hav the same modulus, soo much more secure
alice> i usu just ask sophie to make some more primes for me <33
bob> no prob alie
alice> e = 65537 for me
bob> f = 1000003 for me
charlie> hi y'all, just came outa meeting! stuff is going down, so act fast
charlie> alice: 11759303421060652710502832503103094193 1648
                 81563054015202904435486020358082832621 0173
                 42921220053673947972172591169033419560 86
charlie> bob:   17993061473926966067823108670106219044 5253
                 84013217357380235245077582495096861204 0118
                 66349734411372400461857850055156149876 81
charlie> btw i just strtoint'ed the whole thing, no blocks or anything fancy pantsy
eve> lolh smh
```

6. *RSA*

```
n = 16653052943296534009927166682117653018853597228304609699601636
    42347714232248789104789321176996103506182469478600423434171 59
e = 65537
y = 14944140254560528408209463017103768683741055494303520438904279
    77185400794852570061137872508953549166892608548178211214538 74
```

The plaintext is encoded as an integer in base 26 (modulo n), with 'digits'

$$A = 0,\ B = 1,\ \ldots,\ Z = 25$$

7. *Diffie–Hellman*

```
alice> p = 160077444158133421978060000000000000000000000000000000000000000
           000000000000000000000000000000000000000000000000000000000000000
           000000000000000000000000000000000000432209099226960239340763
bob> Yahoo! What's going on with your prime!?  Oh well, g = 2 as always!
alice> g^a = 115916852703688593321785760893301236896991672430420707679222309
             282509317751229144270012266510200309346808149667382577787878633
             000277154229376611446722889004057521676161442395039882541843793 6
bob> g^b = 575888213681660770475128571088550295674623296573691479085064213
           967623740013786721669248908515706185159465035805440862621304296
           869839120646574688710399284505691846199112507789898432268180523
alice> Let's just add our common secret to the message like a one-time pad
bob> Yeah! I don't trust ASCII--I think it's rigged.
alice> I'm just going to write my message as an integer like
alice>    h e l l o = 07 04 11 11 14 = 704111114
alice> Heading your way!
alice> m: 620945601844830146411372829211407124029166699782737399787363634
          438224667948005628550729901067384508056258649925961457566814249
          100027830822531155426663083200685298415308333701929015786144937
bob> You said it girl!
```

8. *Elliptic curve ElGamel* Elliptic curve ElGamal is used with the following parameters:

- $p = 2^{31} - 1 = 2147483647$
- $E : y^2 = x^3 + x + 1$ over $\mathbb{F}_p$
- $G = (2120200592, 1037835596) \in E(\mathbb{F}_p)$

Bob sends the point

$$(502702028, 397327625) \in E(\mathbb{F}_p).$$

Alice takes her message, encodes it as the x-coordinate of a point on E using `strtoint`, and she sends the pair of points

$$((1271659322, 1653304), (86041769, 166781836).$$

Use baby step–giant step to solve the discrete logarithm problem on E and discover the message.

9.