CS 6222: Introduction to Cryptography **Spring 2020**

Homework 1A: Many-Time Pad Attack

Due: January 22, 2020 at 5pm (Submit on Collab) **Instructor:** David Wu

Instructions. This problem is one component of Homework 1 (and is worth 20% of the credit on Homework 1). Please read the submission instructions carefully before submitting your assignment.

Collaboration Policy. You may discuss your general *high-level* strategy with other students, but you may not share any written documents or code. Your challenge is *unique* to your computing ID, and you must solve your challenge independently. Do *not* share your challenge with anyone other than members of the course staff. You must include the computing IDs of all of your collaborators with your submission (see specific instructions below).

Acknowledgments. This problem is adapted from a homework assignment from Stanford's CS 255 course by Prof. Dan Boneh.

Problem 1: Many-Time Pad Attack [18 points]. In lecture, we said that we should *never* reuse a one-time pad (or more generally, a stream cipher) to encrypt multiple messages. In this exercise, we will see why this is the case. On Collab (under your private folder in "File Drop"), you will find a file (`ctxts.txt`) that contains a collection of 12 hex-encoded ciphertexts that are the result of encrypting 12 plaintext messages with the *same* one-time pad. Each ciphertext appears on a separate line. The file will look something like the following:[1]

```
cd82fe1e777f924ff523a67eca9592dd10d9e61de69bcb778ffae13729173d50206de595878f353a15292ab4d8f3
9e8ae909776c9200f26ba6648d96ddc514d3b107f899887787e0f624281d26576f77f999c184346e412f20e088e0
dbcbf8033a66981df522a67c8d80dddd1dd2b207f792887886e1a222351b72776e6defc78f83353a072e3db4c9f5
db99e84c3464900ab624b26492d79289319ca703b690c76ac9f2e03a385e20576771fed998c6357541203fe4dae4
dc87fe0177649b4fe022b565de95c18911cfe61af393d8719bf2f02f7d1f3c5a206ee3d98dc6237f413220f8dee4
db80e84c366c924fff38e77cc49bd78914d3a905ff90cf3e88e7a235321a371e7976ff958094243a12242afdc6e6
df80fe1f776add02f728af79c39592cf19cfb240b6dee16acee0a22532182649616bef95958e206e412c2effcdf2
dacbef03777c9c1ce22ee77dd4d0c6c015d9e619ff8ac03e88b3e139300e274a656baadf9495353a03242cf5ddf2
f382f81e38789209e267e764c591c6890fd3b302f2deca7bc9f2a23532132252656defd998c63474082f3bf1c6e5
cb98fe4c386ddd2cd909885c8d93c0c008ccaa0be5dedc768cb3ef3f331a691e696df99595832079092821f388f2
dbcbec093660930ae538a2638d99dc8910c9ab0ff8deca7b81f2f43f320c724a6f39efdb9293337f413527f1d1a1
fd99e21c23649a1df73baf75df8392da1dd0a201fbdedb728cf6f2762a1b3e522e39f495ab89243a2a2823fdc9ef
```

Your goal is to decrypt the *last* ciphertext in the file (shown in blue in the above example). In this example, the answer is:

```
Cryptographers seldom sleep well. ~ Joe Kilian
```

[1]The real file will contain ciphertexts for 60-character messages. The example shown here is for shorter (46 character) messages.

Submission instructions. To submit, please upload two files to Collab: `answer.txt` and `collab.txt`:

- `answer.txt`: This file should consist of a *single* line which is the decrypted ciphertext (see example above). You should only decrypt the last ciphertext.
- `collab.txt`: This file should consist of a *single* line with a comma-separated list of your collaborators' computing IDs.

This assignment will be *auto-graded* so not conforming with the above requirements will result in your assignment automatically receiving a grade of zero.

Additional information. In case it is useful, the ciphertexts for this assignment were generated using the following Python script:

```
import os

msgs = [ ACTUAL MESSAGES REMOVED ]

def encrypt(pad, msg):
    return bytes([x ^ ord(y) for (x, y) in zip(pad, msg)]).hex()

pad = os.urandom(60)
ctxts = [encrypt(pad, m) for m in msgs]
print('\n'.join(ctxts))
```

Some additional hints:

- Every message is an English sentence (with possible punctuation). The start and end of each message may be in the middle of a word.
- In Python, you can use the `bytes.fromhex(...)` function to obtain a byte array from a hex-encoded value.
- Think about what happens when a space is xored with a letter.