CS 579: Foundations of Cryptography, Spring 2026
Stevens Institute of Technology
Course Staff: Alex Hoover (instructor), Aya Salama

Project 1: Classical Ciphers

Teams (proj1.0) Due at 11:59pm Thursday, January 29
Part 1 (proj1.1) Due at 11:59pm Thursday, February 12
Part 2 (proj1.2) Due at 11:59pm Sunday, February 22

Introduction

This assignment covers the first (approximately) 2400 years of cryptography. Several ciphertexts, which were encrypted using classical methods, have been captured and stored in the attached files `01.txt`, ..., `20.txt`. Your job is to decrypt them and obtain the messages hidden inside.

This project is divided into a two parts: The first is just a check-in on initial guesses where you can get structured feedback. The second part will consist of actually decrypting the ciphertexts.

In this project you'll get to see hands-on how a cipher that appears to gibberish may have some hidden structure that is easily found by a computer. We recommend you use Python 3 (and not Python 2 or earlier) for this project. This project is also designed to be a gentle introduction to Python, which is a handy tool to know beyond this class. The attached Python file contains a few relevant code snippets to get you started.

Rules and Guidelines

Teams and collaboration policy. This assignment is designed to be completed by teams of three. **Please register your teams at `https://forms.gle/wrWu2EtepdQ9vift9` by 11:59pm Thursday, January 29.** Only one member needs to list the team when it's formed. You can use Piazza to help find a team. You may work in smaller teams if you prefer. Do not discuss the assignment across teams.

Sources. You may Google liberally to learn basic Python if needed, and you should use libraries for non-crypto steps (like dealing with encodings) if needed, but you should not Google for anything crypto-related, including searching for posted solutions from other universities.

Piazza and spoilers. We encourage you to post questions on Piazza related to Python 3 or any aspect of the project. For questions that reveal some of your insights about the ciphertexts, please make your post as **private** to the course staff. **Please do not post any spoilers publicly; when in doubt, mark your question as private to the staff**. We will make public posts if any issues come up that the entire class should know.

Sanity and bugs. It is impossible to guarantee that the assignment is bug-free, so there may be issues that create a headache for you. If something doesn't look right, ask the staff before spending a lot of time debugging it yourself. One year, a (multi-byte!) Unicode character was accidentally left in a plaintext, leading to a lot of headache. We've taken pains to eliminate issues like this,

but it's possible we overlooked something. **Please get in touch with us if you are spending several hours without progress.**

FUN. The first team to find the magic phrase earns a bag of candy. This prize is here so that the course staff can monitor that *someone* has been able to break the ciphers. If no one claims the prize in a reasonable amount of time, we will consider offering hints. The candy has no effect on your grade.

Part 1 (proj1.1): Initial Analysis (10 points)

The point of this part is for you to make some initial progress and receive feedback from the staff. Examine the ciphertexts and perform the initial statistical and syntactic analyses required to determine what types of ciphers were likely employed, and which ciphertexts appear to be encrypted via the same cipher and key.

CONTEXT. You may assume:

- The messages are from multiple sources and not necessarily related. The senders may have chosen different methods independently. There is not a meaningful correspondence between the file names and senders.
- All ciphers used are classical methods. They may not be *exactly* like the ones described in lecture. If you have evidence that a particular type of cipher was used, it might be helpful to consider how small variants can explain what you are seeing.
- All of the messages are in English. The senders chose different methods for dealing with punctuation and spaces, which may or may not appear in the messages at all.

DELIVERABLES AND SUBMISSION. Write a short report structured the following way:

- Give a partition of the ciphertexts into groups, where each group should consist of ciphertexts encrypted with the same key and the same cipher. You may leave some ciphertexts in an uncertain status, and not in any of the groups.
- For each group, record your initial observations about the composition of the ciphertext and your best guess at the type of cipher used. Also describe evidence for your guess and how you derived that evidence.

Submissions will be accepted on Gradescope. Please submit a single PDF/txt file named `writeup.pdf/txt` for your group.

Part 2: Cipher Breaking (90 points)

Now it is time to decrypt as many (and as much) of the ciphertexts as possible. As you're doing so, be sure to remember how you proceeded so that you'll recall major breakthroughs and insights for your write-up.

Deliverables and submission.

There are two deliverables for this part: (1) a report and (2) decrypted messages, which we describe in turn.

REPORT. You should submit an expanded version of our report from Part 1. It should again clearly give the groupings of the ciphertexts. This time, for each group, your report should describe the cipher used (with the details of the specific variation that you found; if you're not entirely sure then report as much as you could figure out). Describe your methodology for how you decrypted the ciphertexts.

DECRYPTED MESSAGES. After you've decrypted as much as possible, create files `01.txt`, `02.txt`, ..., `20.txt` containing your guesses for the decrypted messages. Your will receive partial credit in the case that you decrypted part of the message correctly, so be sure to include what you have even if it is not complete.

Note that we will grade the percentage of the message you get correct, in a byte-aligned way. This means you should not delete or omit unknown characters. Instead, you can put in a special symbol for characters you don't know, or better yet put in guesses. This will keep your correct guesses aligned with the plaintext.

HOW TO SUBMIT. Submissions will be accepted on Gradescope. Please submit a single PDF/txt file named `writeup.pdf/txt` and a zip file `ptxts.zip`. The latter file should unzip to the files `01.txt`, ..., `20.txt`, *in the current directory*, that is, without creating a subdirectory. Only one submission per team is required.

Grading

Part 1 is worth 10 points and Part 2 is worth 90 points.

Part 1 will be graded based on clarity and thoroughness of thinking behind your conclusions. You will not graded on "how far" you got in breaking the ciphertexts beyond making some basic observations.

For Part 2, 60 points will depend on your write-up and the determination of cipher details there. The write-up points will be assigned based on clarity and thoroughness of thinking behind conclusions. You may earn most of these points even if you only made partial progress on some ciphertexts, so be sure to include whatever insights you achieved.

The remaining 30 points of Part 2 depend on the number of correct characters you decrypt, according to the formula: For $i = 1, \ldots, 20$, let $0 \leq s_i \leq 1$ be the fraction of characters correctly recovered from the message in file `i.txt`. Drop the lowest s_i. Then your score is the average of the remaining s_i, scaled between 0 and 30. This part will be computed automatically. This means it is worth noting that your submitted plaintexts should be the *same length* as the plaintexts which were encrypted.

If your plaintext is longer than the correct plaintext, then we will only test character matches up to the length of the correct plaintext. If your plaintext is shorter, then all missing characters will be marked as incorrect. Notably this means if you think a cipher replaces a single character with multiple characters, then in your submitted plaintexts you should not just leave encrypted characters encrypted, even if you don't know the correct decryption. If you do, the plaintext you submit will be much longer than the correct plaintext.