Computer Science Department University of Crete

CS458: Introduction to Cryptography
Assignment 1 - Classical Cryptography

Deadline: 30/10/23 23:59

Introduction

The main purpose of this assignment is to offer you the opportunity to become familiar with the internals and implementation aspects of classical encryption algorithms, help you understand their qualities and shortcomings and discuss techniques for defeating them.

Setup

For this and future assignments you will use Python 3 with Linux, Windows or a Unix based machine. We will walk you through installing Python 3 in Ubuntu Linux in this section. Installation will be slightly different for other versions of Linux or Unix and may be considerably different for Windows.

In order to not mess with your system Python environment, you should create a Python virtual environment using the venv module. This will configure a selected directory with a Python interpreter and associated modules. Any modules you install are only locally installed.

First, we need to install Python 3, Pip, and the venv module:

```
sudo apt install python3 python3-venv python3-pip
```

Next, we create a project directory:

```
mkdir hy458
cd hy458
```

Next, we use venv to set up the environment in an env directory:

```
python3 -m venv <environment name>
```

This will set up the interpreter and modules within the path. Once the installation is complete,the environment can be used at any time by the following command:

```
source <environment name>/bin/activate
```

When working on the assignments, remember to activate your Python virtual environment first if you don't want Python modules to be installed system-wide.

1. [60%] Frequency analysis

Choose three appropriately long (>100K characters) English texts (e.g., freely available textbooks, novels).

[5%] Calculate their (plaintext) letter frequency distribution.

[20%] Encrypt these texts using the classical substitution algorithms listed below. You do not have to implement the algorithms. Use 3rd party tools or libraries to perform all the encryption operations.

- Polybius square
- Caesar (ROT13 is one case)
- Monoalphabetic (Atbash is one case)
- Homophonic
- Book
- Playfair
- Polyalphabetic: Vigenère
- Running Key
- Auto Key
- OTP

[5%] For each algorithm, you must record and discuss the choices you made. These choices depend on the algorithm, for example:

- Encryption key, keyword, key text
- Encryption key size and complexity
- Ciphertext alphabet
- Symbols mapping, etc.

Calculate the ciphertext letter/symbol frequency distribution for all the combinations (i.e., 3 texts x all the algorithms).

[30%] For each text, draw ciphertext letter/symbol frequency distribution graphs based on the results obtained above.

- Use a suitable tool (e.g., excel) to visually compare the distributions in the ciphertext.
- See for example: Create an Interactive Chart with Checkboxes in Microsoft Excel
- https://www.youtube.com/watch?v=eMr1gFdLONU&ab_channel=Teacher%27sTech

Discuss the results. For example, elaborate on why some algorithms are better (i.e., more difficult to crack) than others.

2. [20%] Reverse columnar transposition cipher

In this exercise you are required to implement the encryption (10%) and decryption (10%) process of the reverse columnar transposition cipher. Write a python script that will take as **input** i) the **plaintext** to be encrypted and the **keyword** and produces a ciphertext and ii) the **ciphertext** to be decrypted and the **keyword** to produce the **original plaintext** .Your script should be verbose enough to include print statements for each step of the encryption and decryption process.

Example:

- python encrypt.py <keyword> <plaintext>
- Python decrypt.py <keyword> <ciphertext>

The Reverse Columnar Transposition Cipher, is a variation of the regular Columnar Transposition Cipher. This cipher rearranges the columns of a message in a specific order determined by a keyword. To encrypt a text with the cipher follow the next steps:

a) **Keyword Selection:**
Choose a word that will act as the key for this cipher e.g "**PASSWORD**". This should be secret and not shared.

b) **Arrange the keyword and plaintext in rows:**
The plaintext is written **row-wise** in a grid that has the same row length as the keyword. In the final row if there are empty cells and no letters left add a padding symbol (e.g. "#").

c) **Transpose the plaintext based on the keyword:**
Rearrange the columns based on the alphabetical order of the keyword. If the keyword contains a specific letter more than one time, place the same letter columns according to the order of appearance.

d) **Generate the ciphertext:**
Produce the ciphertext by combining the rows from **right to left** until there are no more rows left.

Decryption: To decrypt a ciphertext encrypted with this cipher follow the same process but in reverse.
Example:

i) **Plaintext:** "FAR OUT IN THE UNCHARTED BACKWATERS"

ii) **Keyword:** "PASSWORD"

Note: *You can capitalize all the letters and ignore all the whitespaces*

Will produce the following **ciphertext**:

UNBROUDEREETIHC#FTRWTCASNAK#AHTA

Original

4	1	6	7	8	3	5	2
P	A	S	S	W	O	R	D
F	A	R	O	U	T	I	N
T	H	E	U	N	C	H	A
R	T	E	D	B	A	C	K
W	A	T	E	R	S	#	#

Transposed

1	2	3	4	5	6	7	8
A	D	O	P	R	S	S	W
A	N	T	F	I	R	O	U
H	A	C	T	H	E	U	N
T	K	A	R	C	E	D	B
A	#	S	W	#	T	E	R

Ciphertext

U	N	B	R	O	U	D	E	R	E	E	T	I
H	C	#	F	T	R	W	T	C	A	S	N	A
K	#	A	H	T	A							

3. [20%] Polybius Square (variant)

In this assignment you are required to implement a variation of Polybius Square. You will need to provide a python file with the functions for grid creation, encrypting (10%) and decrypting (10%). You are also required to make the program verbose by adding the necessary prints (print starting plaintext grid ciphertext decrypt text etc.).

The Polybius Square, also known as the Polybius Checkerboard, is a simple method of encoding letters into a grid of numbers. It is based on a 5x5 grid, where each cell in the grid is assigned a unique combination of two numbers, typically ranging from 11 to 55.

This Encryption algorithm has the following steps:

- Create an 2D array with size 5x5 and fill it randomly with the english alphabet (put the i and j on the same cell). You will end up with something like this:

	1	**2**	**3**	**4**	**5**
1	X	Y	A	B	C
2	D	E	F	G	H
3	I/J	Z	K	L	M
4	N	O	P	Q	R
5	S	T	U	V	W

- Split you plain text in to characters (also remove any spaces)
- Find the corresponding number of your character base on the grid e.g.
  A = 13, O= 42
- Put together all the numbers of the previous steps e.g. HELLOWORLD
  The cipher would be **25**22**34**34**42**45**34**21.

Now reverse the process to decrypt the ciphertext that you created. **You will need to use the existing grid that you create for the encryption**.

For this part of the assignment you need to submit the python file with all your code and a txt named readme that explains how to run your code.

Submission

- Included with the code implementing the programming parts listed above, you should provide explanations for your implemented or unimplemented parts and a README file with your answers to the questions listed above.

- Submissions should be posted to **eLearn**.

- This assignment is an **individual creative process** and students must submit their own work. You are not allowed, under any circumstances, to copy another person's work. You must also ensure that your work will not be accessible to others.

- You are encouraged to post any questions you may have on the eLearn forum. If however you believe that, your question contains part of the solution or spoilers for the other students you can communicate directly with the course instructor and TAs by emailing **h*y458@csd.uoc.gr***.