COMP2700 Lab 7 – Classical ciphers and Modular Arithmetic

The following are selected problems from Paar & Pelzl's "Understanding Cryptography" (Chapter 1).

Lab environment setup

For this lab, and the rest of the labs in this course, you are not required to run the lab VM that was used in Lab 1 – Lab 6, since we will not be modifying any system settings. Instead, we will use the ANU Linux virtual desktops (linuxvdi). If you have not used linuxvdi before, see the appendix in this lab guide on how to connect to ANU Linux virtual desktops.

For Lab 7 – Lab 12, most of the practical exercises use only tools that are already installed in linuxvdi. Additional tools that are not already installed will be added to the course directory on linuxvdi:

/courses/comp2700/public/

For the first two exercises in this lab, we will use JCryptool to do the frequency analysis. Here are a couple of options to run this software:

- If you are using linuxvdi, you can find JCryptool in the directory /courses/comp2700/public/. Create a shortcut (symbolic link) to JCryptool on your desktop using the following commands from the terminal:

  ```
  cd ~/Desktop/
  ln -s /courses/comp2700/public/jcryptool/JCrypTool
  ```

  Then you can double-click on that shortcut from your desktop to launch JCryptool.

- Or, you can install your own copy of JCryptool. Jcryptool runs in Windows, Linux and Mac. You can download the latest version of JCrypTool from:

  https://www.cryptool.org/en/jct/downloads

A short tutorial video (~11 minutes) on how to use JCryptool to do substitution analysis to solve Exercise 1 can be found on the Wattle page for this lab.

Exercise 1. The ciphertext below was encrypted using a substitution cipher. Decrypt the ciphertext without knowledge of the key.

```
lrvmnir bpr sumvbwvr jx bpr lmiwv yjeryrkbi jx qmbm wi
bpr xjvni mkd ymibrut jx irhx wi bpr riirkvr jx
ymbinlmtmipw utn qmumbr dj w ipmhh but bj rhnvwdmbr bpr
yjeryrkbi jx bpr qmbm mvvjudwko bj yt wkbrusurbmbwjk
lmird jk xjubt trmui jx ibndt
  wb wi kjb mk rmit bmiq bj rashmwk rmvp yjeryrkb mkd wbi
iwokwxwvmkvr mkd ijyr ynib urymwk nkrashmwkrd bj ower m
vjyshrbr rashmkmbwjk jkr cjnhd pmer bj lr fnmhwxwrd mkd
wkiswurd bj invp mk rabrkb bpmb pr vjnhd urmvp bpr ibmbr
jx rkhwopbrkrd ywkd vmsmlhr jx urvjokwgwko ijnkdhrii
ijnkd mkd ipmsrhrii ipmsr w dj kjb drry ytirhx bpr xwkmh
mnbpjuwbt lnb yt rasruwrkvr cwbp qmbm pmi hrxb kj djnlb
bpmb bpr xjhhjcwko wi bpr sujsru msshwvmbwjk mkd
wkbrusurbmbwjk w jxxru yt bprjuwri wk bpr pjsr bpmb bpr
riirkvr jx jqwkmcmk qmumbr cwhh urymwk wkbmvb
```

a) Compute the relative frequency of all letters a...z in the ciphertext. You may want to use a tool such as the open-source program JCrypTool for this task. However, a paper and pencil approach is also still doable.
b) Decrypt the ciphertext with the help of the relative letter frequency of the English language (see Table 1.1 in Sect. 1.2.2 in [Paar & Pelzl]). Note that the text is relatively short and that the letter frequencies in it might not perfectly align with that of general English language from the table.

Remark:

- *For your convenience, the cipher text in this question is provided as a separate file (cipher.txt), which you can download from Wattle.*
- *Deciphering the text can take some time; you are not expected to completely decipher the text during the lab session; but you are expected to be at least familiar with the frequency analysis technique used to solve this exercise.*

Exercise 2. We received the following ciphertext which was encoded with a shift cipher:

```
xultpaajcxitltlxaarpjhtiwtgxktghidhipxciwtvgtpilpitghlxiwiwtxgqadds
```

Perform an attack against the cipher based on a letter frequency count: How many letters do you have to identify through a frequency count to recover the key? What is the plaintext?

Exercise 3. As we learned in this chapter, modular arithmetic is the basis of many crypto systems. As a consequence, we will address this topic with several problems in this and upcoming chapters. Let's start with an easy one: Compute the result without a calculator.

a) $15 \cdot 29 \bmod 13$
b) $2 \cdot 29 \bmod 13$
c) $2 \cdot 3 \bmod 13$

d) −11·3 mod 13

The results should be given in the range from 0,1,..., modulus-1. Briefly describe the relation between the different parts of the problem.

Exercise 4. Construct the addition and the multiplication table for the rings Z_5 and Z_6. There are elements in Z_6 without a multiplicative inverse. Which elements are these. Why does a multiplicative inverse exist for all nonzero elements of Z_5.

Exercise 5. Compute without a calculator:

a) 1/5 mod 13
b) 1/5 mod 7
c) 3·2/5 mod 7

Exercise 6. This problem deals with the affine cipher with the key parameters *a* = 7, *b* = 22. Decrypt the text below:

```
falszztysyjzyjkywjrztyjztyynaryjkyswarztyegyyj
```

Exercise 7. In an attack scenario, we assume that the attacker Oscar manages somehow to provide Alice with a few pieces of plaintext that she encrypts. Show how Oscar can break the affine cipher by using two pairs of plaintext–ciphertext, (x_1, y_1) and (x_2, y_2). What is the condition for choosing x_1 and x_2?

Remark: *In practice, such an assumption turns out to be valid for certain settings, e.g., encryption by Web servers, etc. This attack scenario is, thus, very important and is denoted as a chosen plaintext attack.*

Extension Exercise(s) -- Optional

Exercise 8 (*). We consider the long-term security of the Advanced Encryption Standard (AES) with a key length of 128-bit with respect to exhaustive key-search attacks. AES is perhaps the most widely used symmetric cipher at this time.

a) Assume that an attacker has a special purpose application specific integrated circuit (ASIC) which checks $5 \cdot 10^8$ keys per second, and she has a budget of $1 million. One ASIC costs $50, and we assume 100% overhead for integrating the ASIC (manufacturing the printed circuit boards, power supply, cooling, etc.). How many ASICs can we run in parallel with the given budget? How long does an average key search take? Relate this time to the age of the Universe, which is about 10^{10} years.
b) We try now to take advances in computer technology into account. Predicting the future tends to be tricky but the estimate usually applied is Moore's Law, which states that the computer power doubles every 18 months while the costs of integrated circuits stay constant. How many years do we have to wait until a key-search machine can be built for breaking AES with 128 bit with an average search time of 24 hours? Again, assume a budget of $1 million (do not take inflation into account).

Hint: This is an instance of linear search. In this case, the underlying assumption is that keys are uniformly distributed so on average, given a key space of n, is (n+1)/2. Asymptotically this approaches n/2. See, e.g.,

https://en.wikipedia.org/wiki/Linear_search#Analysis

Exercise 9 (*). An obvious approach to increase the security of a symmetric algorithm is to apply the same cipher twice, i.e.:

$$y = e_{k2}(e_{k1}(x))$$

As is often the case in cryptography, things are very tricky, and results are often different from the expected and/or desired ones. In this problem we show that a double encryption with the affine cipher is only as secure as single encryption! Assume two affine ciphers

$e_{k1} = a_1 x + b_1 \ mod\ 26$ and $e_{k2} = a_2 x + b_2 \ mod\ 26$.

a) Show that there is a single affine cipher $e_{k3} = a_3 x + b_3 \ mod\ 26$ which performs exactly the same encryption (and decryption) as the combination $e_{k2}(e_{k1}(x))$.
b) Find the values for a_3, b_3 when $a_1 = 3, b_1 = 5$ and $a_2 = 11, b_2 = 7$.
c) For verification: (1) encrypt the letter K first with e_{k1} and the result with e_{k2}, and (2) encrypt the letter K with e_{k3}.
d) Briefly describe what happens if an exhaustive key-search attack is applied to a double-encrypted affine ciphertext. Is the effective key space increased?

Remark: *The issue of multiple encryption is of great practical importance in the case of the Data Encryption Standard (DES), for which multiple encryption (in particular, triple encryption) does increase security considerably.*

APPENDIX: Connecting to the ANU Linux Virtual Desktops

The following instructions to Linux VDI were copied from COMP1100 course.

The ANU School of Computing provides a virtual desktop environment for education. The VDI is a small piece of software that runs on your home computer that allows you to run a *virtual desktop* that is identical to the Linux desktop available in our physical labs at ANU. To use the VDI you must carefully follow these steps:

1. If you are trying to access the VDI from off-campus you must be connected to ANU via the **GlobalProtect** VPN. If you have not already installed GlobalProtect, you can follow the instructions provided by the University. If you are already on a university network you will not need to use GlobalProtect.

2. You can access the VDI either directly via a browser (using this link), or by installing the VDI client software, **VMWare Horizon**, following the instructions provided by the University. Note that the university instructions will set you up to use the standard ANU Microsoft Windows client (which we *do not* use in our course, but which may be useful in other ANU courses).

    - If you use the client, and you have the VMWare Horizon client installed and working, then you need to add the VDI as a server. The way to do this varies depending on your operating system (please see the ITS instructions). The server address you need to add is **linuxvdi.anu.edu.au**.

3. Now you should be able to create a connection and you'll see a login screen just like in the ANU's physical labs, and will see the standard Ubuntu image. (If you just see a blank screen that may be due to you using a second monitor. I suggest that you attempt the next step even if your screen is black.)

4. Finally, if you are using the client and VMWare Horizon is in full screen mode, you will probably want to move it into a **single window**. This step may be important in order to **screen share** with your tutor, which will be important in the labs. To move out of full screen mode:

    - Move your mouse to the top edge of the VDI screen. The VMWare Horizon menu should appear. Once it does, you can choose the Window menu and uncheck the "Full Screen" option.

    - If you're still seeing blank screens, move to another app (on Windows hold down the ALT key and press the space bar; on MacOS hold down the CMD key and press the window), and then move your mouse back to VDI.

Important: Please note that the VDI is available **6:00am to 12:00am AEST**, 7 days a week (see here for further information).

Some known issues and work-arounds:

- GlobalProtect gets stuck 'Connecting' on MacOS. You need to enable Palo Alto Network as a trusted developer:

    - On your Mac, launch System Preferences

    - Open the Security & Privacy preferences and then select General
    - Click the lock icon on the bottom left of the window to make changes and modify preferences
    - When prompted, enter your Mac User Name and Password and then Unlock the preferences
    - Click “Allow” next to the message “System software from developer “Palo Alto Networks” was blocked from loading.”
- Once you connect to the VDI you just get black screens.
    - If you are using an external monitor (dual monitors), you may need to disconnect one of the monitors. Once you turn off full screen mode (see above).