

# Lab 2: Attacking Classic Crypto Systems

## 1. Checkpoint 1 — Caesar cipher (Marks: 5)

**Cipher (given):**

odroboewscdrolocdcwkbmyxdbkmdzvkdpybwyyeddrobo

### 1.1 Objective

Write a program to break the Caesar cipher and display the plaintext.

### 1.2 Approach

- A Caesar cipher is a substitution where each letter is shifted by a fixed offset (0 - 25).
- Brute-force all 26 shifts and inspect outputs. The correct shift yields readable English.
- Implementation choices: C++ program.

### 1.3 Steps performed

1. Created a small script to try all shifts (see Appendix for the code).
2. Ran the script and inspected the output lines.
3. Identified the readable plaintext and recorded the shift.

### 1.5 Result

```

File Edit Selection View ... ← → ⌂ ins lab
EXPLORER OPEN EDITORS
INS-LAB Tasks task-2 checkpoint1.cpp ...
checkpoint2.cpp...
aes.py INS-Lab-T...
rsa.py INS-Lab-T...
rsaSignature.py...
shaHashing.py...
timeGraph.py...
input.txt INS-Lab...
rsa_public.pem...
main.py INS-Lab...
INS LAB
vscode
INS-Lab-Tasks
task-2
checkpoint1.cpp
checkpoint1... M
checkpoint2.cpp
checkpoint2.exe
task-3
checkpoint1
cipher_cbc.bin
cipher_cfb.bin
cipher_ecb.bin
cipher_ofb.bin
decrypt_cbc.txt
decrypt_cfb.txt
plain.txt
> checkpoint2
> checkpoint3
OUTLINE
TIMELINE
File Explorer Search Taskbar
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\ins lab\INS-Lab-Tasks\Task-2" ; if ($?) { g++ checkpoint1.cpp -o checkpoint1 ; if ($?) { ./checkpoint1 }
clipertext: odroboewscrolocdcwkdmydbkmdzvkdpybwyeddrobo
Trying all 26 shifts (shift = number of positions the cipher was shifted):
shift 0: odroboewscrolocdcwkdmydbkmdzvkdpybwyeddrobo
shift 1: ncgnandvrbzqknrbcbvjaclwxcaj1cyujcovaaxdcqnan
shift 2: mbpmzmcuaqpmpjnaabauzbkwbz1kbtxb1bnrwuucbcpazm
shift 3: taolylbtpzaoliizazthya:jvuiayhjaws:hamytyvbaoyl
shift 4: kznikkasoyznkhgzyyspxiutzg1zvngzluxsaaznnkk
shift 5: jymjwzrnxymjgjxyxrlyhtsyfwtwzzywjjw
shift 6: ixliivijymxlfiflwxaqevgsrxvegtxpejsqvyyxxlivi
shift 7: hmduhxp1wlkhehwwpuwlfrpmudwsodwiruprxwkuuh
shift 8: gvigtgwokurgdguvuoctveqptcevrmvcvhgtogwvrijtg
shift 9: fulifsfvnjtuifcftutnsdupsbdunbugsgnpnuuifsf
shift 10: ethereumisthebestsmartcontractplatformoutthere
shift 11: dsgpdqtlhrsgldadrs1zqsbmmsgzbsokzseng1nttsqdq
shift 12: crfpcsksggrfczzcrqkpram1rpnnryrdmpksnrfpc
shift 13: bgeobrnfppqbyppqjxog1lkqozxqnlxqloj1rqebob
shift 14: apdاناqlepdaxaooipnypkjpwwyj1lwpbkni1kppdانا
shift 15: zoczmzphdmbyzwnnhmoxj1omwxokpovojej1poocmz
shift 16: ynblyycgcmbyyymmgulwhinluwjfunzilgionbyly
shift 17: xmaxxnmfb1maxux1mlf1kmvhgmkvtwcmhkhfrmmxx
shift 18: w1avjwmaek1zwblkesj1ugfl1su1hds1xgjeqn1lwjw
shift 19: vkyvi1dzlykysvjkjdr1ktfekirtkgcrkw1df1kkyiv
shift 20: ujxuhukcyljxuruji1chjedjhqsffhjvhvehckj1xuhu
shift 21: t1wgtjtjbxhiwtqthhbgi1rdcigprieapuidgbhj1iwgt
shift 22: shvsiawhspshgaofhqchbfqohdzohtcfaci1hvsfs
shift 23: rgurerhzvfgurorfgfzneppbagengcynpsbzehgurer
shift 24: afpqdgqyueftqnoefeyndfaorfdmofbxmfradygf1ftqdq
shift 25: pescpfxtdespmddx1cenye1neadleqzcxzfeespcp
PS E:\ins lab\INS-Lab-Tasks\Task-2"

```

- **Detected shift:** 10
- **Recovered plaintext:** **ethereum is the best smart contract platform out there**

## 2. Checkpoint 2 — Substitution ciphers (Marks: 8 + 7)

**Cipher-1:** af p xpkcaqvnpk pfg, af ipqe qpri, gauuikfc tpw, ceiri udvk tiki  
 afgarxitphni cd eao- -wvmd popkwn, hiqpvri du ear jvaql vfgikrcpfafm du cei  
 xkafqaxnir du xrwqedearcdkw pfg du ear aopmafpcasi xkdhafmr afcd fit pkpr. ac tpr  
 qdoudkcafcd lfdt cepc au pfwceafm epxxifig cd ringdf eaorinu hiudki cei  
 opceiopcaqr du cei uaing qdvng hi qdoxnicinw tdklig dvc- -pfg edt rndtnw ac  
 xkdqjigig, pfg edt odvfcfafdvr cei dhrcpqnir--ceiki tdvng pc niprc kiopaf dfi mddg  
 oafg cepc tdvng qdfcafvi cei kiripkqe

**Cipher-2:** aceah toz puvg vcdl omj puvg yudqecov, omj loj auum klu thmjuv hs klu  
 zlcu shv zcbkg guovz, upuv zcmdu lcz vuwovroaeu jczoyyuovomdu omj  
 qmubyudkuj vukqvm. Klu vcdluz lu loj avhqnlk aodr svhw lcz kvopuez loj mht  
 audhwu o ehdoe eunumj, omj ck toz yhyqeoveg auecupuj, tlokupuv klu hej sher  
 wcnlk zog, klok klu lceee ok aon umj toz sqee hs kqmmuez zkqssuj tckl kvozqvu.

omj cs klok toz mhk umhqnl shv sowu, kluvu toz oezh lcz yvhehmnuj pcnhqv kh wovpue ok. kcwu thvu hm, aqk ck zuuwuj kh lopu eckkeu ussudk hm wv. aonncmz. ok mcmukg lu toz wqdl klu zowu oz ok scskg. ok mcmukg-mcmu klug aunom kh doee lcw tuee-yvuzuvpuj; aqk qmdlomnuj thqej lopu auum muovuv klu wovr. kluvu tuvu zhwu klok zlhhr klucv luojz omj klhqnlk klcz toz khh wqdl hs o nhhj klcnn; ck zuuwuj qmsocv klok omghmu zlhqej yhzzuzz (oyyovumkeg) yuvyukqoe ghqkl oz tuee oz (vuyqkujeg) cmubloqzkcaeu tuoekl. ck tcee lopu kh au yocj shv, klug zocj. ck czm'k mokqvoe, omj kvhqaeu tcee dhwu hs ck! aqk zh sov kvhqaeu loj mhk dhwu; omj oz wv. aonncmz toz numuvhzq tckl lcz whmug, whzk yuhyeu tuvu tceecmn kh shvncpu lcw lcz hjjckcuz omj lcz nhhj shvkqmu. Lu vuwocmuj hm pczckcmn kuvwz tckl lcz vueokcpuz (ubduyk, hs dhqvzu, klu zodrpceeu-aonncmzuz), omj lu loj womg juphkuj ojwcuvuz owhmn klu lhaackz hs yhhv omj qmcwyhvkomk sowcecz. aqk lu loj mh dehzu svcumjz, qmkce zhwu hs lcz ghqmnuv dhqzcmz aunom kh nvht qy. klu uejuzk hs kluzu, omj aceah'z sophqvcku, toz ghqmnuv svjhj aonncmz. tlum aceah toz mcmukg-mcmu lu ojhykuj svjhj oz lcz lucv, omj avhqnlk lcw kh ecpu ok aon umj; omj klu lhyuz hs klu zodrpceeu-aonncmzuz tuvu scmoeeg jozluj. aceah omj svjhj loyyumuj kh lopu klu zowu acvkijog, zuykuwaav 22mj. ghq loj aukkuv dhwu omj ecpu luvu, svjhj wg eoj, zocj aceah hmu jog; omj klum tu dom dueuavoku hqv acvkijog-yovkcuz dhwshvkoae khnukluv. ok klok kcwu svjhj toz zkcee cm lcz ktuumz, oz klu lhaackz doeeuj klu cvvuzyhmzcaeau ktumkcuz auktuum dlcejlhj omj dhwcmn hs onu ok klcvkg-klvuu

## **Frequency distribution English characters**

### **1.2 Substitution Ciphers.**

**a: 8.05% b: 1.67% c: 2.23% d: 5.10%**

**e: 12.22% f: 2.14% g: 2.30% h: 6.62%**

**i: 6.28% j: 0.19% k: 0.95% l: 4.08%**

**m: 2.33% n: 6.95% o: 7.63% p: 1.66%**

**q: 0.06% r: 5.29% s: 6.02% t: 9.67%**

**u: 2.92% v: 0.82% w: 2.60% x: 0.11%**

**y: 2.04% z: 0.06%**

## 2.1 Objective

Write programs to decipher both substitution ciphers. Explain which input was easier to break and why.

## 2.2 Background and approach

- A monoalphabetic substitution cipher replaces each plaintext letter with a unique ciphertext letter (a permutation of the alphabet). It preserves letter frequencies and many word patterns.
- **Attack strategy used:**
  1. **Frequency analysis** to build an initial mapping: map the most frequent ciphertext letters to the most frequent English letters.
  2. **hill-climbing:** starting from the initial key, repeatedly propose small changes (swap two plaintext-letter assignments) and accept changes that improve a scoring function.
  3. **Scoring function:** measures how English-like a candidate decryption is. Options used in experiments: common-word match counts, n-gram (quadgram) log-probability scoring.

This approach is standard and effective for reasonably long ciphertexts.

## 2.3 Steps performed

1. Implemented a substitution-solver program
  - reads the ciphertext,
  - computes letter frequencies,
  - builds an initial key via frequency ordering,
  - runs randomized hill-climbing with swap moves,
  - prints the best-scoring plaintext and the inferred mapping.
2. Ran the solver separately on Cipher-1 and Cipher-2.

- Manually inspected the best candidates and performed small manual fixes when needed to obtain readable plaintext.

## 2.5 Results and analysis

```

File Edit Selection View ... ← → Q ins lab 08 □ - ×
EXPLORER ... checkpoint1.cpp checkpoint2.cpp aes.py rsa.py rsaSignature.py shaHashing.py timeGraph.py input.txt rsa_public.pem ...
OPEN EDITORS
INS-Lab-Tasks > task-2 > checkpoint1.cpp > ...
1 #include <bits/stdc++.h>
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
E:\ins lab> cd "E:\ins lab\INS-Lab-Tasks\task-2"
Shift 21: tuitgtjbjhxhxtqthhbpgjrcigpriepiudghjji.lwigt
Shift 22: shufsfisamghvsnphgaofhqchbhfoghdzohtcfacihvhsfs
Shift 23: rgurerhazvfgurorfgfzneqphagongcngshbzthgurer
Shift 24: afferdguguettquoeeyewfzoafdmoflxmfradygffftqdq
Shift 25: pescprfxtdespapdxdlcenzyeclnawleqzczfeespc
PS E:\ins lab\INS-Lab-Tasks\task-2> cd "E:\ins lab\INS-Lab-Tasks\task-2"
if ($?) { g++ checkpoint2.cpp -o checkpoint2 } ; if ($?) { ./checkpoint2 }
==== Decrypting Cipher 1 ====
in a particullar and, in each case, different way, these four were indingermable to his-yupo asaryl, because of him juicy understanding of the grincigl
em of grycholimtry and of his iaspinatike grobing into new aresan. it was conforting to know that if anything happened to maldon himself before the sa
thesatium of the field could be cospletely woved out--and how slowly it groeeded, and how sountainoum the obtaxlem=there would at leant resain one
poond sind that would continue the remeck
==== Decrypting Cipher 2 ====
cilco gas verp riyh and verp beyuliar, and had been the gonder of the shire for sijtp peers, ever sinje his remarkaicle disabbaraneye and unejbeyted ret
urn. the ryfles he had crownt cayk from his travels had nog ceome a loyal lewend, and it gas bobularlp believed, ghattever the old folk miwt sap, tha
t the hill at caw end gas full of tullen stufed gith treasure, and if that gas not enough for fame, therne gas also his broldom viour to marvel at.
time gone on, cut it seemed to have little effeyt on mr. cawwins, at ninete he gas mylh the same as at fift, at ninetp-nine thep ewan to yall him ge
ll-preserved; cut unyhaned gold had been nearer the mark. there gera some that shook their heads and thownd this gas too mylh of a wood thrin; it's
eemed unfair that anyone should bossess (abberantlp) berberthal pouth as gell as (rebutedlp) inejhasticle gealth. it gill have to be bald for, thep sai
d. it isn't natural, and troule gill yome of it! cut so far troule had not yome; and as mr.
PS E:\ins lab\INS-Lab-Tasks\task-2>

```

### Which input was easier to break?

- Cipher-2** (the longer text) was easier to break in automated experiments because: longer ciphertexts provide more reliable statistics (letter frequency and n-gram patterns), they contain more repeated words and grammatical structure which the scoring function exploits, and the hill-climber can lock onto consistent mappings more easily.