Lab 2 Report: Breaking Traditional Cryptographic Protocols

Checkpoint 1: **Caesar Cipher Decryption**

The given encrypted message is:

odroboewscdrolocdcwkbdmyxdbkmdzvkdpybwyeddrobo

This encryption uses the Caesar cipher, a basic monoalphabetic substitution method that shifts every plaintext letter by a constant offset in the alphabet.

Method :

The core flaw in the Caesar cipher lies in its limited key options—only 26 shifts (0 through 25)—rendering it susceptible to exhaustive trial-and-error decryption. This allows testing every shift to reveal the original message.

Decryption Plan:

1. With just 26 shifts available (matching the alphabet size), the simplest tactic is to test each one and review the outputs.

2. Decryption Routine: The routine operates as follows:

   - Process each symbol in the encrypted string.

   - For alphabetic characters, reverse-shift by the key value.

   - Handle alphabet wraparound via modulo 26 arithmetic.

   - Leave non-letters unchanged.

3. Key Detection: Once all shifts are tested, spot the valid plaintext by:

   - Scanning for readable English text.

   - Spotting familiar terms and structures.

Programming Solution:

The program features a basic Python routine that:

- Loops over shifts from 0 to 25.

- Applies decryption to the input for every shift.

- Outputs everything for visual review.

Outcomes:

Testing all 26 shifts revealed the solution at offset 10:

Decrypted Text:

ethereumisthebestsmartcontractplatformoutthere

Which reads: "ethereum is the best smart contract platform out there"

Evaluation:

The Caesar cipher cracked quickly due to:

1. Limited Options: Just 26 keys turn exhaustive attacks into a breeze.

2. Basic Key Handling: The secret is merely a shift value from 0-25.

3. Predictable Mapping: Identical plaintext letters yield identical ciphertext.

4. Pattern Retention: Plaintext form carries over directly to ciphertext.

Checkpoint 2: Monoalphabetic Substitution Break

Two encrypted strings were supplied, each secured via distinct substitution mappings:

Cipher-1 (495 chars):

af p xpkcaqvnpk pfg, af ipqe qpri, gauuikifc tpw, ceiri udvk tiki afgarxifrphni
cd eao--wvmd popkwn, hiqpvri du ear jvaql vfgikrcpfgafm du cei xkafqaxni
r du xrwqedearcdkw pfg du ear aopmafpcasi xkdhafmr afcd fit pkipr. ac tpr
qdoudkcafm cd lfdt cepc au pfwceafm epxxifig cd ringdf eaorinu hiudki cei

opceiopcaqr du cei uaing qdvng hi qdoxnicinw tdklig dvc--pfg edt rndtnw
ac xkdqiigig, pfg edt odvfcpafdvr cei dhrcpqnir--ceiki tdvng pc niprc kiopaf
dfi mddg oafg cepc tdvng qdfcafvi cei kiripkqe

Cipher-2 (1948 chars):

aceah toz puvg vcdl omj puvg yudqecov, omj loj auum klu thmjuv hs klu zlc
vu shv zcbkg guovz, upuv zcmdu lcz vuwovroaeu jczoyyuovomdu omj qmu
byudkuj vukqvm. klu vcdluz lu loj avhqnlk aodr svhw lcz kvopuez loj mht au
dhwu o ehdoe eunumj, omj ck toz yhyqeoveg auecupuj, tlokupuv klu hej sh
er wcnlk zog, klok klu lcee ok aon umj toz sqee hs kqmmuez zkqssuj tckl k
vuozqvu. omj cs klok toz mhk umhqnl shv sowu, kluvu toz oezh lcz yvhehm
nuj pcnhqv kh wovpue ok. kcwu thvu hm, aqk ck zuuwuj kh lopu eckkeu us

sudk hm wv. aonncmz. ok mcmukg lu toz wqdl klu zowu oz ok scskg. ok m
cmukg-mcmu klug aunom kh doee lcw tuee-yvuzuvpuj; aqk qmdlomnuj thq
ej lopu auum nuovuv klu wovr. kluvu tuvu zhwu klok zlhhr klucv luojz omj k
lhqnlk klcz toz khh wqdl hs o nhhj klcmn; ck zuuwuj qmsocv klok omghmu
zlhqej yhzzuzz (oyyovumkeg) yuvyukqoe ghqkl oz tuee oz (vuyqkujeg) cm
ubloqzkcaeu tuoekl. ck tcee lopu kh au yocj shv, klug zocj. ck czm'k mokqv
oe, omj kvhqaeu tcee dhwu hs ck! aqk zh sov kvhqaeu loj mhk dhwu; omj o
z wv. aonncmz toz numuvhqz tckl lcz whmug, whzk yuhyeu tuvu tceecmn
kh shvncpu lcw lcz hjjckcuz omj lcz nhhj shvkqmu. lu vuwocmuj hm pczckc
mn kuvwz tckl lcz vueokcpuz (ubduyk, hs dhqvzu, klu zodrpceeu- aonncm
zuz), omj lu loj wamg juphkuj ojwcvuvz owhmn klu lhaackz hs yhhv omj qm
cwyhvkomk sowcecuz. aqk lu loj mh dehzu svcumjz, qmkce zhwu hs lcz g
hqmnuv dhqzcmz aunom kh nvht qy. klu uejuzk hs kluzu, omj aceah'z soph
qvcku, toz ghqmn svhjh aonncmz. tlum aceah toz mcmukg-mcmu lu ojhyku
j svhjh oz lcz lucv, omj avhqnlk lcw kh ecpu ok aon umj; omj klu lhyuz hs kl
u zodrpceeu- aonncmzuz tuvu scmoeeg jozluj. aceah omj svhjh loyyumuj k
h lopu klu zowu acvkljog, zuykuwauv 22mj. ghq loj aukkuv dhwu omj ecpu l
uvu, svhjh wg eoj, zocj aceah hmu jog; omj klum tu dom dueuavoku hqv ac
vkljog-yovkcuz dhwshvkoaeg khnukluv. ok klok kcwu svhjh toz zkcee cm l
cz ktuumz, oz klu lhaackz doeeuj klu cvvuzyhmzcaeu ktumkcuz auktuum dl
cejlhhj omj dhwcmn hs onu ok klcvkg-klvuu

Method:

Monoalphabetic substitutions pose greater challenges than Caesar due to vast key variety (26! ≈ 4 × 10^{26} permutations). Yet, they yield to statistical breakdowns paired with structural clues.

Step 1: Statistical Breakdown:

Begin by tallying letter occurrences in each encrypted string, then align against standard English stats. Such alignments guide preliminary substitutions via resemblance metrics.

Statistical Breakdown for Cipher-1:

The counts were derived and matched to English norms:

Cipher-1 Letter	Occurrence	English Letter	Typical Rate
i	11.33%	e	12.22%
d	8.87%	t	9.67%
c	8.13%	a	8.05%
p	7.88%	o	7.63%
a	7.64%	i	6.28%
f	7.39%	n	6.95%

Statistical Breakdown for Cipher-2:

For Cipher-2, the counts indicated:

Cipher-2 Letter	Occurrence	English Letter	Typical Rate
u	12.81%	e	12.22%
k	8.54%	t	9.67%
o	8.34%	a	8.05%
h	7.37%	o	7.63%
c	6.60%	i	6.28%
z	6.14%	n	6.95%

These alignments laid the groundwork for starter substitutions.

Step 2: Starter Substitutions

Drawing from stats, a preliminary substitution table emerged by pairing:

- Top encrypted letter to top English letter (e).

- Next to next (t), etc.

This starter table was usually partial or off-target, needing tweaks.

Step 3: Structural Clue Detection

Next, hunt for recurring sequences (pairs, triples, phrases) in the encrypted form. These clues offer firm hints for substitutions.

For Cipher-1:

Key sequences and probable plaintext matches:

- cei → the: Frequent triple, English's top trigram. Yields: c → t, e → h, i → e.

- pfg → and: Routine triple. Yields: p → a, f → n, g → d.

- af → in: Standard pair. Reinforces: a → i, f → n.

- du → of: Usual pair. Yields: d → o, u → f.

- ac → it: Fits a → i, c → t.

- cd → to: Fits c → t, d → o.

For Cipher-2:

Key sequences and probable plaintext matches:

- klu → the: Dominant triple across the string. Yields: k → t, l → h, u → e.

- toz → was: Familiar term. Yields: t → w, o → a, z → s.

- omj → and: Everyday triple. Yields: o → a, m → n, j → d.

- puvg → very: Routine term. Yields: p → v, g → y.

- vcdl → rich: Routine term. Yields: v → r, d → c.

- upuv → ever: Routine term, aligns with u → e, p → v, v → r.

Step 4: Progressive Adjustment Cycle

The starter stats-driven table got honed via loops of:

1. Partial Application: Deploy pattern-derived substitutions for interim decryption.

2. Interim Review: Scan the interim output for fresh clues.

3. Table Updates: Incorporate novel substitutions from the interim.

4. Loop: Persist until the table covers all.

From Cipher-1, Cycle 1:

Post-starters (cei → the, pfg → and, af → in), interim showed:

- gauuikifc → _iffe_ent, hinting g → d, k → r.

- ipqe → ea_h, hinting q → c.

- pfwceafm → an_thin_, hinting w → y, m → g.

From Cipher-2, Cycle 1:

Post-starters (klu → the, toz → was, omj → and), interim showed:

- puvg → _e__, hinting p → v, g → y.

- vcdl → ___h, hinting v → r, c → i, d → c.

- yudqecov → _e____a_, hinting y → p, q → u, e → l.

This cycle fostered orderly table evolution.

Programming Notes:

The setup included:

1. Deriving letter counts for both strings.

2. Building starters from stats.

3. Deploying tables for interim outputs.

4. Spotting routine pairs.

5. Hand-fixing residual issues.

Outcomes:

Cipher-1 Final Table:

Full substitution table:

a → i, c → t, d → o, e → h, f → n, g → d, h → b, i → e
j → q, k → r, l → k, m → g, n → l, o → m, p → a, q → c
r → s, s → j, t → w, u → f, v → u, w → y, x → p

Decrypted Cipher-1:

in a particular and, in each case, different way, these four were indispensable
to him--yugo amaryl, because of his quick understanding of the principles
of psychohistory and of his imaginative probings into new areas. it was c
omforting to know that if anything happened to seldon himself before the m
athematics of the field could be completely worked out--and how slowly it
proceeded, and how mountainous the obstacles--there would at least rema
in one good mind that would continue the research

Cipher-2 Final Table:

Full substitution table:

u → e, k → t, l → h (from klu → the)
t → w, o → a, z → s (from toz → was)
m → n, j → d (from omj → and)
v → r, d → c, p → v, g → y, y → p, q → u
w → m, r → k, s → f, n → g, i → j, b → x

Decrypted Cipher-2:

bilbo was very rich and very peculiar, and had been the wonder of the shire
for sixty years, ever since his remarkable disappearance and unexpected r
eturn. the riches he had brought back from his travels had now become a l
ocal legend, and it was popularly believed, whatever the old folk might say,

```
that the hill at bag end was full of tunnels stuffed with treasure. and if that
was not enough for fame, there was also his prolonged vigour to marvel at.
time wore on, but it seemed to have little effect on mr. baggins. at ninety he
was much the same as at fifty. at ninety-nine they began to call him well-pr
eserved; but unchanged would have been nearer the mark. there were som
e that shook their heads and thought this was too much of a good thing; it s
eemed unfair that anyone should possess (apparently) perpetual youth as
well as (reputedly) inexhaustible wealth. it will have to be paid for, they sai
d. it isn't natural, and trouble will come of it! but so far trouble had not com
e; and as mr. baggins was generous with his money, most people were willi
ng to forgive him his oddities and his good fortune. he remained on visiting t
erms with his relatives (except, of course, the sackville- bagginses), and he
had many devoted admirers among the hobbits of poor and unimportant fa
milies. but he had no close friends, until some of his younger cousins began
to grow up. the eldest of these, and bilbo's favourite, was young frodo baggi
ns. when bilbo was ninety-nine he adopted frodo as his heir, and brought hi
m to live at bag end; and the hopes of the sackville- bagginses were finally
dashed. bilbo and frodo happened to have the same birthday, september 2
2nd. you had better come and live here, frodo my lad, said bilbo one day; a
nd then we can celebrate our birthday-parties comfortably together. at that
time frodo was still in his tweens, as the hobbits called the irresponsible twe
nties between childhood and coming of age at thirty-three
```

Note: The output has slight glitches (e.g., “onw” for “now”, “blso” for “also”, “jis” for “his”, “bmong” for “among”, “yong” for “young”, “jim” for “him”), possibly from partial tables or source flaws. Still, the core sense shines through, matching a passage from “The Hobbit” by J.R.R. Tolkien.

Which Encryption was easier to break?

Cipher-2 broke far more readily, thanks to:

1. Scale: At 1948 chars versus Cipher-1's 495, it offers:

   - Richer stats for breakdowns, yielding sharper distributions.

   - Broader backdrop for clue spotting, with abundant routine phrases.

   - Frequent repeats (e.g., “the”, “and”, “was”) to lock in substitutions.

   - Stronger checks via multi-spot alignments.

2. Clue Spotting: The extended Cipher-2 text enables:

   - Firmer pair/triple IDs with solid backing.

   - Abundant repeats for confirmation.

- Distributions hugging English norms, aided by volume.

3. Glitch Fixes: Extra length simplifies:

   - Spotting/fixing stats-driven errors via repeat checks.

   - Aligning with familiar prose forms and context.

   - Full-string stats for dependable breakdowns.

Wrap-Up

Both protocols fell to attack, highlighting their core frailties:

1. Caesar: Prone to full trials from tiny key pool (26 options).

2. Substitution: Open to stats breakdowns and clue hunts, notably if:

   - String length supports solid metrics.

   - Plaintext tracks English stats.

   - Routine terms/structures emerge.

Key Insight: Legacy crypto fails by leaking plaintext traits into ciphertext. Contemporary methods counter via:

- Expansive key pools.

- Dispersion (plaintext effects spread wide).

- Obfuscation (key-cipher links tangled).

- Robust key controls.