

Report on

Lab 2: Attacking Classic Crypto Systems

Github: https://github.com/mustakinshvn/INS_Lab

Checkpoint 1: Breaking the Caesar Cipher

Understanding the Caesar Cipher

The Caesar cipher is a simple substitution cipher where each letter in the plaintext is shifted by a fixed number of positions in the alphabet. For example, with a shift of 3, 'A' becomes 'D', 'B' becomes 'E', and so on.

Given Cipher: `odroboewscdroloocdcwkbdmyxdbkmdzvkdpybwyeddrobo`

Analysing the problem statement

1. The cipher is a classic Caesar shift applied to letters a to z
2. The plaintext language is English
3. No side information is available.

Approach: Brute Force Attack

Since the Caesar cipher has only 26 possible keys (shifts from 0 to 25), a brute force approach is both practical and guaranteed to succeed. My methodology was:

1. **Try all possible shifts (0-25)**
2. **Decrypt the cipher** with each shift value
3. **Identify readable English text** manually and statistical methods
4. **Validate the result** by checking if it makes linguistic sense manually

Implementation Details

```
-caesarCipherBreakDown.py - Python script that implements the  
brute-force Caesar-breaker and ranking.  
- cipherText.txt - given ciphertext used when the script is run without  
command-line arguments.
```

Score measurement:

1. Made a short ordered list of common words. More common words earlier for higher weight

```
COMMON_WORDS = [
    "the", "be", "to", "of", "and", "a", "in", "that", "have", "i", "it", "for", "not", "on", "with",
    "he", "as", "you", "do", "at", "this", "but", "his", "by", "from", "they", "we", "say", "her", "she",
    "or", "an", "will", "my", "one", "all", "would", "there", "their", "what", "so", "up", "out", "if",
    "about", "who", "get", "which", "go", "me", "when", "make", "can", "like", "time", "no", "just",
    "him", "know", "take", "people", "into", "year", "your", "good", "some", "could", "them", "see",
    "other", "than", "then", "now", "look", "only", "come", "its", "over", "think", "also", "back",
    "after", "use", "two", "how", "our", "work", "first", "well", "way", "even", "new", "want", "because"
]
```

```
_WORD_WEIGHTS: Dict[str, int] = {}
for idx, w in enumerate(_COMMON_WORDS):
    base = max(1, len(w))
    bonus = 3 if idx < 20 else 2 if idx < 50 else 1
    _WORD_WEIGHTS[w] = base * bonus
```

Display shift and 10 top ranked decrypted text

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS
PS D:\INS_Lab> & C:/Python313/python3.13t.exe d:/INS_Lab/labManual1/caesarCipherBreakDown.py				
● Using sample ciphertext from cipherText.txt: odroboewscdroloocdcwkbldmyxdbkmdzvkdpybweddrobo				
shift=10 score= 172 plaintext : ethereumis the best smart contract platform out there				
shift=14 score= 84 plaintext : apdanaqieopdaxaopo iwnpykjpnwyp lhwpbknikqppdana				
shift= 1 score= 78 plaintext : ncqnandvrbcqnknbcvjaclxwcajlcuyjcoaxavxdccqnan				
shift= 3 score= 76 plaintext : laolylbtpzaolilzazthyajv uayhjawshamvytvbaaolyl				
shift=21 score= 76 plaintext : tiwtgtjbxhiwtqthihbpgir dcigprieapiudgbdj ijiwtgt				
shift=16 score= 73 plaintext : ynbylyogcmnbyvymnmgu lnwi hnlu n jfunzilgionnbyly				
shift= 6 score= 71 plaintext : ixliviyqmwxlifiw xwqevxgsrxvegxtpexjsvqsyxxlivi				
shift= 4 score= 68 plaintext : kznkxkasoyznkhk yzsgxziutxgizvrgzluxsuazznkxk				
shift= 9 score= 67 plaintext : fui fsfvnjtuifcfutnbsudpous bduq mbugpsnpvuuifsf				
shift=22 score= 67 plaintext : shvsfsiawghvpsghgaofhqc bho qhdzohtcfaci hhsfs				
○ PS D:\INS_Lab> []				

In shift **Shift 10**: [ethereumis the best smart contact platform out there](#), we got the highest score (172). So, this stands. Also I've checked it manually.

Observations

Shift 10 Contains the substrings ethereum is the best smart contact platform out there visually.

It is very plausible the intended sentence is:

"ethereum is the best smart contract platform out there"

What I did next

1. Insert expected spaces around obvious words: ethereum is the best smart contract platform out there.
2. Corrected two small letter swaps that appear in the raw decryption:
 - a. ethreum => ethereum => likely a single-letter corruption.
 - b. contlact => contract and platfrom => platform => both look like common typing transposition errors in English.

Weakness Demonstrated: Caesar cipher provides no meaningful security against even the most basic cryptanalysis.

Checkpoint 2

Problem

For this problem, I tried doing it on a Jupyter notebook. The notebook loads two ciphertexts (called Cipher-1 and Cipher-2) and attempts to recover their plaintexts using classical frequency analysis and iterative substitution mapping. The notebook's first output confirms both ciphertexts were loaded and prints their lengths:

- Cipher-1 length: **495** characters
- Cipher-2 length: **1949** characters

Given the lengths and the visible structure of the text, the working assumption is that both ciphertexts are encrypted English prose using simple substitution cipher. The goal is to find a mapping from ciphertext letters.

Initial Plan

The notebook follows this approach:

1. **Character frequency analysis:** compute the character frequency distribution of each ciphertext and compare with typical English frequencies.
2. **Produce an initial mapping (key_map):** Do a substitution mapping by mapping the most frequent ciphertext letters to the most frequent English letters.
3. **Apply partial decryption:** apply the current mapping to the ciphertext to get a partially decrypted text. This partially revealed text will show some complete words and many underscores/unknown letters.
4. **Iteratively refine mapping:** choose familiar words or letter patterns visible in the partial plaintext and use them to deduce more ciphertext to plaintext mappings. Reapply and repeat until the plaintext is readable.
5. **Use contextual clues & n-gram intuition:** Identify function words (the, and, was, had, been, in, of, that, etc.), repeated words and phrases, capitalized words where appropriate, and punctuation to guess precise mappings for ambiguous letters.
6. **Finalize mapping and print final plaintext** Once the mapping is finally complete, grammatical English sentences across the ciphertext.

This is my general approach for substitution ciphers. The notebook implements each of these steps sequentially for Cipher-1 and then for the larger Cipher-2.

Detailed reasoning for mapping

For cipher-1, using frequency mapping:

1. The most frequent cipher letter is ‘i’, so it likely corresponds to ‘e’, the most common English letter. The second-most frequent letter ‘d’ is tentatively mapped to ‘t’. The single-letter words ‘p’ and ‘a’ must be ‘a’ or ‘l’. Since ‘a’ occurs more often in English, map p => a and a => i.
2. A frequent cipher word “cei” appears repeatedly and is often followed by a space, suggesting “the.”
Thus, c => t, e => h, i => e.
This confirms i => e but invalidates d => t because t is now taken.
3. Common patterns provide more clues:
 - a. “cd” likely means “to”, giving d => o.
 - b. “du” often fits “of”, so u => f.
 - c. “af” aligns with “in”, confirming a => i, f => n.
 - d. “ac” becomes “it.”
 - e. “pfg” fits “and,” giving g => d.
4. In the partially decoded text, we see:
 - a. ipqe => each \Rightarrow q => c
 - b. qpri => case \Rightarrow r => s
 - c. gauuikfc => different \Rightarrow k => r
 - d. tpw => way \Rightarrow t => w, w => y

5. The long token xpkcaqvnpk initially seemed to fit “practical.” However, that would force p => r, contradicting the well-supported p => a. The “practical” hypothesis is therefore rejected to preserve internal consistency.
6. Further doing thought process
 - a. cei => the
 - b. af => in
 - c. ac => it
 - d. cd => to
 - e. pfg => and
 - f. ipqe => each
 - g. qpri => case
 - h. gauuikfc => different
 - i. tpw => way

All decoded patterns form meaningful English words, confirming the mapping’s reliability.

```
--- Final Key Map for Cipher-1 ---
'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'
```

7. The refined substitution key accurately decodes large portions of Cipher-1, exposing coherent phrases such as

“in a practical and, in each case, different way, there...”

The same thought process applied for Cipher-2.

Notebook outputs & final plaintexts

I ran the notebook outputs and included the decrypted results it printed.

Final Decryption of 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 imaginatije probings into new areas. it was comforting to know that if anything happened to seldon himself before the mathematics of the field could be completely worked out--and how slowly it proceeded, and how mountainous the obstacles--there would at least remain one good mind that would continue the research

Final Decryption of 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 return. the riches he had brought back from his travels had now become a local 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-preserved; but unchanged would have been nearer the mark. there were some that shook their heads and thought this was too much of a good thing; it seemed unfair that anyone should possess (apparently) perpetual youth as well as (reputedly) inexhaustible wealth. it will have to be paid for, they said. it isn't natural, and trouble will come of it! but so far trouble had not come; and as mr. baggins was generous with jis money, most people were willing to forgive him jis oddities and jis good fortune. he remained on visiting terms with jis relatives (except, of course, the sackville- bagginses), and he had many devoted admirers among the hobbits of poor and unimportant families. but he had no close friends, until some of jis younger cousins began to grow up. the eldest of these, and bilbo's favourite, was young frodo baggins. when bilbo was ninety-nine he adopted frodo as jis heir, and brought jim 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 22nd. you had better come and live here, frodo my lad, said bilbo one day; and then we can celebrate our birthday-parties comfortably together. at that time frodo was still in jis tweens, as the hobbits called the irresponsible twenties between childhood and coming of age at thirty-three

What I did next

1. Insert expected spaces, punctuations around obvious words: ethereum is the best smart contract platform out there.
2. Corrected the letter swaps that appear wrong in the raw decryption using human knowledge :
 - a. **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 comforting to know that if anything happened to Seldon himself before the mathematics of the field could be completely worked out – and how slowly it proceeded, and how mountainous the obstacles – there would at least remain one good mind that would continue the research.
 - b. **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 return. The riches he had brought back from his travels had now become a local 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-preserved; but unchanged would have been nearer the mark. There were some that shook their heads and thought this was too much of a good thing; it seemed unfair that anyone should possess (apparently) perpetual youth as well as (reputedly) inexhaustible wealth. It will have to be paid for, they said. It isn't natural, and trouble will come of it! but so far trouble had not come; and as Mr. Baggins was generous with his money, most people were willing to forgive him for his oddities and his good fortune. He remained on visiting terms with his relatives (except, of course, the sackville- bagginses), and he had many devoted admirers among the hobbits of poor and unimportant families. 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 baggins. When Bilbo was ninety-nine he adopted Frodo as his heir, and brought him 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 22nd. "You had better come and live here, Frodo my lad", said Bilbo one day; and then we can celebrate our birthday-parties comfortably together. At that time Frodo was still in his tweens, as the hobbits called the irresponsible twenties between childhood and coming of age at thirty-three.

Verdict

Among the two ciphertext, Cipher-1 was much harder to break as the cipher text was shorter compared to Cipher-2. Cipher-2 has lots of repeated structures and double-letter patterns ("well-preserved", "Baggins"), which make consonant and vowel placement easier. Proper

nouns (Bilbo, Baggins, Frodo, Bag End, September) occurred with distinctive letter shapes, that helped mappings quickly once one instance is guessed. Frequencies stabilize fast and common words/bigrams (the, and, of, to, was) appear a ton that also helped to map the characters.

Conclusion

The automatic decryption gives a plaintext that, after minimal human correction, becomes a perfectly sensible English sentence with clear meaning.