SUB: Information Security

AY 2023-24 (Semester-V)

Experiments No: 3

D - 12

60009210105 **Amitesh Sawarkar**

Aim: Design and implement Encryption and Decryption Algorithm for Caesar Cipher, Shift Cipher. Also Perform Brute Force Attack on Ciphers.

Theory:

1. Caesar Cipher.

The Caesar cipher is the simplest and oldest method of cryptography. The Caesar cipher method is based on a mono-alphabetic cipher and is also called a shift cipher or additive cipher. Julius Caesar used the shift cipher (additive cipher) technique to communicate with his officers. For this reason, the shift cipher technique is called the Caesar cipher. The Caesar cipher is a kind of replacement (substitution) cipher, where all letter of plain text is replaced by another letter.

Let's take an example to understand the Caesar cipher, suppose we are shifting with 1, then A will be replaced by B, B will be replaced by C, C will be replaced by D, D will be replaced by C, and this process continues until the entire plain text is finished.

Caesar ciphers is a weak method of cryptography. It can be easily hacked. It means the message encrypted by this method can be easily decrypted.

Plaintext: It is a simple message written by the user.

Ciphertext: It is an encrypted message after applying some technique.

SUB: Information Security

2. **Shift Cipher.**

The Caesar Cipher is a type of **shift cipher**. Shift Ciphers work by using the modulo operator to encrypt and decrypt messages. The Shift Cipher has a **key K**, which is an **integer from 0 to 25**. We will only share this key with people that we want to see our message.

For every letter in the message **M** :

1. Convert the letter into the number that matches its order in the alphabet starting from 0, and call this number **X**.
(A=0, B=1, C=2, ...,Y=24, Z=25)

2. Calculate: **Y** = **(X + K)** mod **26**

3. Convert the number **Y** into a letter that matches its order in the alphabet starting from 0.

(A=0, B=1, C=2, ...,Y=24, Z=25)

3. **Brute Force Attack on Ciphers**

A brute force attack is a method of attempting to gain unauthorized access to encrypted data or a secured system by systematically trying all possible combinations of keys or passwords until the correct one is found. When it comes to ciphers, such attacks typically involve trying every possible decryption key until the plaintext message is obtained.

Here's a basic overview of how a brute force attack on ciphers works:

1. **Understand the Cipher**: The attacker needs to have knowledge of the cipher algorithm being used, including the encryption method and any parameters such as the key length.

SUB: Information Security

2. **Key Space Analysis**: The attacker calculates the size of the key space, which is the total number of possible keys. For example, with a simple Caesar cipher, where each letter is shifted by a fixed amount, the key space is small because there are only 26 possible shifts (for the English alphabet). In contrast, modern encryption algorithms like AES use much larger key spaces.

3. **Generate Keys**: The attacker systematically generates keys according to the cipher's parameters. This can involve trying all possible combinations of characters for a password or trying all possible values for a symmetric encryption key.

4. **Encrypt and Compare**: For each generated key, the attacker encrypts the ciphertext using that key and checks if the resulting plaintext is meaningful. If it is, the correct key has been found, and the attacker gains access to the decrypted data.

5. **Time and Resources**: The time and resources required for a successful brute force attack depend on the size of the key space. Larger key spaces make brute force attacks infeasible within a reasonable timeframe, especially if the attacker lacks significant computational power.

6. **Countermeasures**: To protect against brute force attacks, strong encryption algorithms with large key spaces should be used. Additionally, enforcing password policies, implementing rate limiting for login attempts, and using multi-factor authentication (MFA) can help mitigate the risk of brute force attacks.

It's important to note that modern encryption algorithms, when used with sufficiently long and complex keys, are designed to resist brute force attacks. The use of proper key management and security practices is essential for safeguarding encrypted data and systems. As computing power continues to advance, it's important to regularly assess and update encryption methods and key lengths to stay ahead of potential threats.

SUB: Information Security

Example:

1) ATTACK K=3

SUB: Information Security

```
string = input("Enter string: ")
encrypted = ''
key = 3
for x in string:
  if(ord(x)==32):
    encrypted += " "
  else:
    encrypted += chr(((ord(x) + key - ord('a'))%26 + ord('a')))
print(encrypted)
```

```
Enter string: attack
dwwdfn
```

```
string = input("Enter string: ")
decrypted = ''
key = -3
for x in string:
  if(ord(x)==32):
    decrypted += " "
  else:
    decrypted += chr(((ord(x) + key - ord('a'))%26 + ord('a')))
print(decrypted)
```

```
Enter string: dwwdfn
attack
```

SUB: Information Security

```python
#Brute force method
string = input("Enter string: ")
for i in range(1, 26):
    decrypted = ''
    key = -i
    for x in string:
        if(ord(x)==32):
            decrypted += " "
        else:
            decrypted += chr(((ord(x) + key - ord('a'))%26 + ord('a')))
    print(decrypted)
```

```
Enter string: dwwdfn
cvvcem
buubdl
attack
zsszbj
yrryai
xqqxzh
wppwyg
voovxf
unnuwe
tmmtvd
sllsuc
rkkrtb
qjjqsa
piiprz
ohhoqy
nggnpx
mffmow
leelnv
kddkmu
jccjlt
ibbiks
haahjr
gzzgiq
fyyfhp
exxego
```

SUB: Information Security

2) ACADEMY K=25

SUB: Information Security

```
string = input("Enter string: ")
encrypted = ''
key = 3
for x in string:
    if(ord(x)==32):
        encrypted += " "
    else:
        encrypted += chr(((ord(x) + key - ord('a'))%26 + ord('a')))
print(encrypted)
```

✓ 3.9s

```
dfdghpb
```

```
string = input("Enter string: ")
decrypted = ''
key = -3
for x in string:
    if(ord(x)==32):
        decrypted += " "
    else:
        decrypted += chr(((ord(x) + key - ord('a'))%26 + ord('a')))
print(decrypted)
```

✓ 2.1s

```
academy
```

```
#Brute force method
string = input("Enter string: ")
for i in range(1, 26):
    decrypted = ''
    key = -i
    for x in string:
        if(ord(x)==32):
            decrypted += " "
        else:
            decrypted += chr(((ord(x) + key - ord('a'))%26 + ord('a')))
    print(decrypted)
```

✓ 2.4s

SUB: Information Security

```
cecfgoa
bdbefnz
academy
zbzcdlx
yaybckw
xzxabjv
wywzaiu
vxvyzht
uwuxygs
tvtwxfr
susvweq
rtruvdp
qsqtuco
prpstbn
oqorsam
npnqrzl
mompqyk
lnlopxj
kmknowi
jljmnvh
ikilmug
hjhkltf
gigjkse
fhfijrd
egehiqc
```

Conclusion:

1. Caesar Cipher is a simple and easily breakable encryption method.
2. The brute force attack tries all possible keys (0 to 25) to decrypt the ciphertext.
3. In the example provided, the key "3" was used for encryption, and the brute force attack successfully revealed the original message by trying all possible keys.
4. Caesar Cipher should not be used for securing sensitive information because it's vulnerable to brute force attacks and frequency analysis.
5. Modern encryption algorithms with strong key management should be used for real-world applications.