Encryption - Caesar cipher

1 Setup

1. Create the playground
mkdir -p caesar
echo "Hello," > README.txt
echo "All of your data has been encrypted. To recover your data, you will need to solve a cipher. To get started look for a hidden file in the caesar subdirectory." >> README.txt

--- Step-by-step shift of “In order to recover…” three places to the right
**echo "Lq rughu wr uhfryhuru llhv brx zloo qhgghu wr" **
**"hqwhu wkh iroorzqj frppdqg:\n\nopenssl dho-256-fef -sbngix2 -d -a -rq" **
**"-q S1.hqfubshgf -rxw S1.uhfryhuhg -n hwwxeurwh" **
**| tr "a-z" "d-za-c" | tr "A-Z" "D-ZA-C" **
> caesar/.leftShift3

Component	Explanation
`echo`	Command used to display a line of text or string to standard output (stdout).
`tr "a-z" "d-za-c"`	`tr` is a translate command; it performs a character-by-character substitution. Here, it shifts lowercase letters 3 places to the left (Caesar cipher decryption). `"a-z"` is the source alphabet, and `"d-za-c"` is the target alphabet, implementing a left shift of 3.
`tr "A-Z" "D-ZA-C"`	Same as above, but for uppercase letters, shifting 3 characters backward in the alphabet.
`>`	Redirection operator; writes the output of the entire piped command sequence into a file instead of displaying it on the terminal.
`caesar/.leftShift3`	Target output file where the decrypted text is saved. `caesar` is the folder and `.leftShift3` is the file name.

```
tiago-paquete@Linux:~$ mkdir -p caesar

tiago-paquete@Linux:~$ echo "Hello," > README.txt

tiago-paquete@Linux:~$ echo "All of your data has been encrypted. To
recover your data, you will need to solve a cipher. To get started look
for a hidden file in the caesar subdirectory." >> README.txt
```

```
tiago-paquete@Linux:~$ echo "Lq rughu wr uhfryhuru Ilhv brx zloo qhgghu
wr" \
"hqwhu wkh iroorzqj frppdqg:\n\nopenssl dho-256-fef -sbngix2 -d -a -rq"
\
"-q S1.hqfubshgf -rxw S1.uhfryhuhg -n hwwxeurwh" \
| tr "a-z" "d-za-c" | tr "A-Z" "D-ZA-C" \
> caesar/.leftShift3

tiago-paquete@Linux:~$ cat README.txt
=====================================================================
Hello,
All of your data has been encrypted. To recover your data, you will need
to solve a cipher. To get started look for a hidden file in the caesar
subdirectory.
=====================================================================
tiago-paquete@Linux:~$ cat caesar/.leftShift3
=====================================================================
Ot uxjkx zu xkiubkxux Loky eua corr tkjjkx zu ktzkx znk lurructm
iussgtj:\q\qrshqvvo gkr-256-ihi -veqjla2 -g -d -ut -t V1.ktixevkji -uaz
V1.xkiubkxkj -q kzzahxuzk
=====================================================================
```

--- Provide a pre-encrypted payload

echo "Cybersecurity is fun!" > secret_message.txt

openssl aes-256-cbc -pbkdf2 -a -e -in secret_message.txt -out Q1.encrypted -k ettubrute

Component	Explanation
`openssl`	The command-line tool for using the OpenSSL cryptography library.
`aes-256-cbc`	Specifies the cipher algorithm to use: AES (Advanced Encryption Standard) with a 256-bit key in CBC (Cipher Block Chaining) mode.
`-pbkdf2`	Enables PBKDF2 (Password-Based Key Derivation Function 2), a secure key derivation function that replaces the older, less secure method.
`-a`	Tells OpenSSL to base64-encode the output (or decode it during decryption). This is useful for making binary data ASCII-readable.
`-e`	Stands for "encrypt". This tells OpenSSL to perform encryption (as opposed to `-d` for decryption).
`-in secret_message.txt`	Specifies the input file to be encrypted, which in this case is `secret_message.txt`.
`-out Q1.encrypted`	Specifies the output file, where the encrypted data will be written. Here, it's `Q1.encrypted`.
`-k ettubrute`	Specifies the password to use for key derivation. In this case, the password is `ettubrute`. **Note**: Using `-k` directly is not recommended in scripts.

```
tiago-paquete@Linux:~$ echo "Cybersecurity is fun!" > secret_message.txt
tiago-paquete@Linux:~$ openssl aes-256-cbc -pbkdf2 -a -e -in
secret_message.txt -out Q1.encrypted -k attubrute

tiago-paquete@Linux:~$ ls -la | grep -E ' secret| README| Q1'
==========================================================================
-rw-rw-r--  1 tiago-paquete tiago-paquete    65 May  5 11:54
Q1.encrypted
-rw-rw-r--  1 tiago-paquete tiago-paquete   165 May  5 11:45 README.txt
-rw-rw-r--  1 tiago-paquete tiago-paquete    22 May  5 11:50
secret_message.txt
==========================================================================
```

```
#  --- Prep two nearly-identical files for hashing
echo 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-
FILE!$H+H*' > file1.txt
cp file1.txt file2.txt

tiago-paquete@Linux:~$ echo 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-
ANTIVIRUS-TEST-FILE!$H+H*' > file1.txt
tiago-paquete@Linux:~$ cp file1.txt file2.txt

# Subtly corrupt file2.txt so hashes will differ
echo '#' >> file2.txt

tiago-paquete@Linux:~$ echo '#' >> file2.txt


tiago-paquete@Linux:~$ cat file1.txt
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
tiago-paquete@Linux:~$ cat file2.txt
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
#
```

2 Hands-On Tasks

Task 2.1 Reconnaissance and orientation

List top-level items

```
tiago-paquete@Linux:~$ ls -l | grep -E “Q1|READ|file|file|caesar"
==========================================================================
drwxrwxr-x 2 tiago-paquete tiago-paquete 4096 May  5 11:45 caesar
-rw-rw-r-- 1 tiago-paquete tiago-paquete   69 May  5 12:01 file1.txt
-rw-rw-r-- 1 tiago-paquete tiago-paquete   71 May  5 12:02 file2.txt
-rw-rw-r-- 1 tiago-paquete tiago-paquete   65 May  5 11:54 Q1.encrypted
-rw-rw-r-- 1 tiago-paquete tiago-paquete  165 May  5 11:45 README.txt
==========================================================================
```

Read the ransom note

cat README.txt

```
tiago-paquete@Linux:~$ cat README.txt
==========================================================================
Hello,
All of your data has been encrypted. To recover your data, you will need
to solve a cipher. To get started look for a hidden file in the caesar
subdirectory.
==========================================================================
```

Task 2.2 Hunt the hidden clue

Change into the sub-directory and show all entries, even dot-file:

```
tiago-paquete@Linux:~$ cd caesar
tiago-paquete@Linux:~/caesar$ ls -la
==========================================================================
total 12
drwxrwxr-x  2 tiago-paquete tiago-paquete 4096 May  5 11:45 .
drwxr-x--- 19 tiago-paquete tiago-paquete 4096 May  5 12:02 ..
-rw-rw-r--  1 tiago-paquete tiago-paquete  163 May  5 11:45 .leftShift3
==========================================================================
```

View the encrypted clue:
cat .leftShift3

```
tiago-paquete@Linux:~/caesar$ cat .leftShift3
========================================================================
Ot uxjkx zu xkiubkxux Loky eua corr tkjjkx zu ktzkx znk lurructm
iussgtj:\q\qrshqvvo gkr-256-ihi -veqjla2 -g -d -ut -t V1.ktixevkji -uaz
V1.xkiubkxkj -q kzzahxuzk
========================================================================
```

Break the Caesar cipher (shift 3 → left):
cat .leftShift3 | tr 'd-za-cD-ZA-C' 'a-zA-Z'

```
tiago-paquete@Linux:~/caesar$ cat .leftShift3 | tr 'd-za-cD-ZA-C' ‘a-zA-Z'
========================================================================
Lq rughu wr uhfryhuru Ilhv brx zloo qhgghu wr hqwhu wkh iroorzqj
frppdqg:\n\nopenssl dho-256-fef -sbngix2 -d -a -rq -q S1.hqfubshgf -rxw
S1.uhfryhuhg -n hwwxeurwh
========================================================================
```

Copy the plaintext command that appears.
Return to your home directory for the next task:

cd ~

Task 2.3 Recover the hostage file

Execute exactly what the clue told you (no edits!):

openssl aes-256-cbc -pbkdf2 -a -d -in Q1.encrypted -out Q1.recovered -k attubrute

Component	Explanation
`openssl`	The OpenSSL command-line tool used for cryptographic operations like encryption, decryption, key generation, certificate handling, etc.
`aes-256-cbc`	Specifies the cipher algorithm: AES (Advanced Encryption Standard) with 256-bit key size in CBC (Cipher Block Chaining) mode.
`-pbkdf2`	Enables the use of PBKDF2 (Password-Based Key Derivation Function 2), a more secure key derivation method than the default (which is considered weaker).
`-a`	Tells OpenSSL to expect input/output in base64 format. It is used for encoding (or decoding) the binary ciphertext in a textual format.
`-d`	Decryption mode. Tells OpenSSL to decrypt the input data.
`-in Q1.encrypted`	Specifies the input file (`Q1.encrypted`) that contains the encrypted data to be decrypted.
`-out Q1.recovered`	Specifies the output file (`Q1.recovered`) where the decrypted data will be written.
`-k ettubrute`	Provides the passphrase (`ettubrute`) directly on the command line to derive the encryption key. Not recommended for secure usage (use `-pass` instead).

tiago-paquete@Linux:~$ openssl aes-256-cbc -pbkdf2 -a -d -in Q1.encrypted -out Q1.recovered -k attubrute

Verify success:
cat Q1.recovered

```
tiago-paquete@Linux:~$ cat Q1.recovered
==========================================================================
Cybersecurity is fun!
==========================================================================
```

Task 2.4 Integrity detective work

Hash each file.

sha256sum file1.txt
sha256sum file2.txt

Note the digests differ even though the files once looked identical.

```
tiago-paquete@Linux:~$ sha256sum file1.txt
========================================================================
131f95c51cc819465fa1797f6ccacf9d494aaaff46fa3eac73ae63ffbdfd8267
file1.txt
========================================================================

tiago-paquete@Linux:~$ sha256sum file2.txt
========================================================================
6765bd1477b6411b130a0d705122f744c5171e47ed271101e43a23e397677b54
file2.txt
========================================================================
```

Persist the hashes for later forensics.

sha256sum file1.txt >> file1hash
sha256sum file2.txt >> file2hash

tiago-paquete@Linux:~$ sha256sum file1.txt >> file1hash
tiago-paquete@Linux:~$ sha256sum file2.txt >> file2hash

Display them side-by-side:

cat file1hash
cat file2hash

tiago-paquete@Linux:~$ cat file1hash
==========
131f95c51cc819465fa1797f6ccacf9d494aaaff46fa3eac73ae63ffbdfd8267 file1.txt
==========

tiago-paquete@Linux:~$ cat file2hash
==========
6765bd1477b6411b130a0d705122f744c5171e47ed271101e43a23e397677b54
file2.txt
==========

Pinpoint the first byte difference automatically:

cmp file1hash file2hash

```
tiago-paquete@Linux:~$ cmp file1hash file2hash
==========
file1hash file2hash differ: byte 1, line 1
==========
```