



Faculty Of Engineering and Technology
Electrical And Computer Engineering Department

Linux Lab

ENCS3130

Shell Scripting Project Report

Students:

Saif Aldeen battah_1170986

Shaymaa dawabsha _1191708

Dr: Mohammad Jubran

TA: Ibrahim injas

Sections: 3

Date: 11/8/2022

Abstract

The aim of this project is to be more familiar with shell script programming by building a shell script that does a simple program which does some operations in Text Message Encryption and Decryption

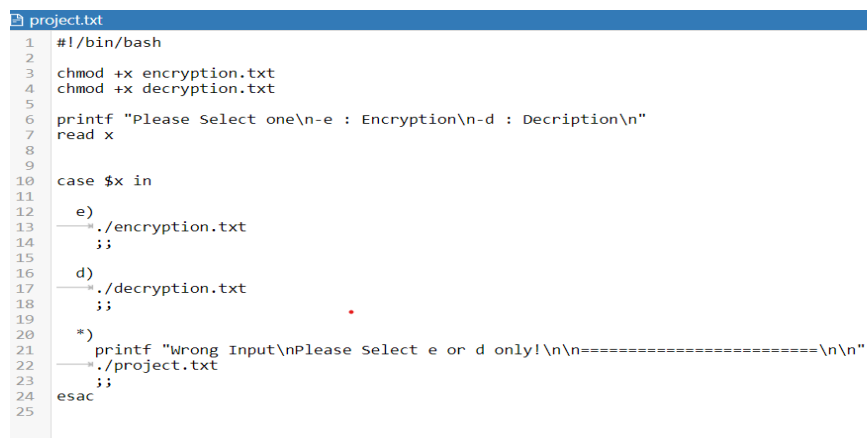
Table of Content

General idea	4
Proses	6
Procurer	11
Conclution	16

General idea

At first in this project , we wrote a shell script code that asks the user to choose between d and e and any other character that outputs an error message. We created a plain file to write the text with, and a cipher file for Outlook

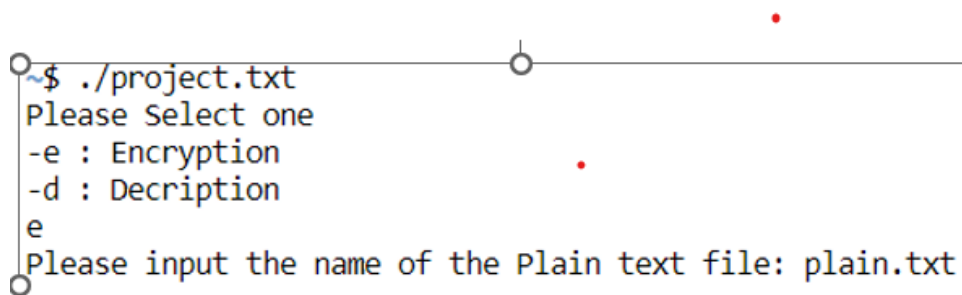
The code is completely

A screenshot of a text editor showing a shell script file named 'project.txt'. The script starts with a shebang line '#!/bin/bash' and sets permissions for 'encryption.txt' and 'decryption.txt'. It then prompts the user to select between encryption ('e') and decryption ('d'). If 'e' is selected, it runs './encryption.txt'. If 'd' is selected, it runs './decryption.txt'. If any other character is entered, it prints an error message and runs './project.txt' again. The script ends with 'esac' and a newline character.

```
1 #!/bin/bash
2
3 chmod +x encryption.txt
4 chmod +x decryption.txt
5
6 printf "Please Select one\n-e : Encryption\n-d : Decryption\n"
7 read x
8
9
10 case $x in
11     e)
12     ./encryption.txt
13     ;;
14     d)
15     ./decryption.txt
16     ;;
17     *)
18     printf "Wrong Input\nPlease Select e or d only!\n\n=====\\n\\n"
19     ./project.txt
20     ;;
21 esac
22
23
24
25
```

fig1.1

result for choosing e

A screenshot of a terminal window showing the execution of the script. The user enters './project.txt' at the prompt. The script outputs 'Please Select one' followed by '-e : Encryption' and '-d : Decryption'. The user enters 'e'. The script then prompts 'Please input the name of the Plain text file: plain.txt'.

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
e
Please input the name of the Plain text file: plain.txt
```

Fig1.2

result for choosing d

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
d
Please input the name of the Cipher text file: cipher.txt
```

Fig1.3

result for choosing other character

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
y
Wrong Input
Please Select e or d only!

=====

Please Select one
-e : Encryption
-d : Decryption
[
Wrong Input
Please Select e or d only!

=====
```

```

string=$(cat $PN)
if [[ ! $string =~ $pat ]] #check that file only contains characters
then
    printf "Error: file contains non-alphabet characters!\n"
    exit
fi
len=`expr length "$string"` #gets the length of the sentence to be encrypted

```

Fig1.4 code of this

process

To calculate the key

We read the blain file and it split the word letter by letter and makes sure that the character index is the order of the character in the English alphabet. I.e., The character index of “A” or “a” is 1, and the character index of “W” or “w” is 23. Calculate sum for each word then mod 256 we take the max and convert to 8 binary bits.

```

if [ $chhar -le 90 ]
then
    val=$(( $chhar - 64 ))
    #echo $val
else
    val=$(( $chhar - 96 ))
    #echo $val
fi

if [ $val != -32 ]
then
    sum=$(( $sum + $val ))
    result=$(( $sum % 256 ))
else
    #echo "sum= $sum"
    #echo "result = $result"
    echo "$result" >> results.txt
    sum=0
fi
done

sort -n results.txt > sorted.txt

MAX=$(tail -1 sorted.txt)
#echo "Max = $MAX"
bin=`echo ${D2B[$MAX]}`
echo "Key: Decimal = $MAX | Binary = $bin"
printf "\n"

```

Fig2.1

Encryption

Read the file and convert the character to ASCII code then to binary code and for each character in the text file compute the XOR between the key generated and the ASCII code of the character. The result will be 8-binary digit. and each 8 bits make flip between the first and last 4 bit and store in cipher code After finish sentences make flip between first and last 4 bit and store it as character in the last in cipher file .

```

sum=0
pat="^[a-z A-Z]*[a-z A-Z]$"
D2B=({0..1}{0..1}{0..1}{0..1}{0..1}{0..1}{0..1}{0..1})
read -p "Please input the name of the Plain text file: " PN
if [ -f "$PN" ]; then
    printf "\n"
    string=$(cat $PN)
    if [[ ! $string =~ $pat ]] #check that file only contains characters
    then
        printf "Error: file contains non-alphabet characters!\n"
        exit
    fi
    len=`expr length "$string"` #gets the length of the sentence to be encrypted

    for (( i=0; i<$len; i++ ))
    do
        chhar=$( printf "%d\n" "${string:i:1}" )

        if [ $chhar -le 90 ] # for characters only in ASCII table
        then
            val=$(( $chhar - 64 )) # if Capital letter subtract 64 from ascii value to get the needed value
            #echo $val
        else
            val=$(( $chhar - 96 )) # if Small letter subtract 96 from ascii value to get the needed value
            #echo $val

            #-----
            read -p "Please input the name of the Cipher text file: " cipher_file
            touch $cipher_file

            for (( j=0; j<$len; j++ ))
            do
                Ascii=$( printf "%d\n" "${string:j:1}" ) #get the decimal ascii value of character
                #echo "Decimal = $Ascii | Binary = $Ascbin"
                xor_res=$(( $Ascii ^ $MAX )) # xor between character and key
                xor_bin=`echo ${D2B[$xor_res]}` # transfere result of xor to binary 8 bit
                #echo "XOR Result = $xor_bin"
                swapped=$( printf "${xor_bin:4:4}${xor_bin:0:4}\n" ) # swap the first and last 4 bits of character
                #echo "XOR Result = $xor_bin | Swapped = $swapped"
                echo "$swapped" >> temp.txt # append result to temp file
            done
            printf "\n"
            #transfere new lines to spaces
            cat temp.txt | tr "\12" "x" > temp_2.txt # transfere new line to x
            sed 's/x//g' temp_2.txt > $cipher_file # transfere x to space and append to cipher file
            "

```

Fig2.3

Decryption process

We open the file and check if each character represented by 8 bits

How?

The number of digits is a multiple of 8

We read the length of the file in the cipher file and make mod 8 if the result zero the encryption is true and we can make decryption if the result not equal to zero there is an error.

```
#!/bin/bash

rm plain.txt
pat="^[01]*[01]$"

read -p "Please input the name of the Cipher text file: " CN
if [ -f "$CN" ]; then
    string=$(cat $CN)
    if [[ ! $string =~ $pat ]]
    then
        printf "Error: file contains something wrong"
        exit
    fi
    len=`expr length "$string"`
    res=$((len%8))
    #echo "len = $len | res = $res"
    if [[ $res != 0 ]]
    then
        echo "Not Divided well, Cant be Decrypted!"
        exit
    else
        key=$( cat $CN | grep -o '.....$')
        original_key=$( printf "${key:4:4}${key:0:4}" )
        dec_key=$(echo "$((2#$original_key))")
        echo "key: Decimal = $dec_key | Binary = $original_key"
        loop_size=$(( (len-8)/8 ))
        read -p "Please input the name of the Plain text file: " plain_file
        touch $plain_file
    fi
fi
```

Fig2.4

After we check we convert the key from binary to decimal
get key (the last character in the encryption file) and swap the
first 4-bit with the last four bit

for each character in the encrypted file, swap the first 4-bit with
the last four bit

Do the XOR between the key and each character from the
encrypted file. The result from binary to ascii

```
else
    key=$( cat $CN | grep -o '.....$')
    original_key=$( printf "${key:4:4}${key:0:4}" )
    dec_key=$(echo "$((2#$original_key))")
    echo "key: Decimal = $dec_key | Binary = $original_key"
    loop_size=$(( (len-8)/8 ))
    read -p "Please input the name of the Plain text file: " plain_file
    touch $plain_file
    for (( k=0; k<$loop_size; k++ ))
    do
        chars=$(echo "${string:k*8:8}")
        original_chars=$( printf "${chars:4:4}${chars:0:4}" )
        dec_chars=$(echo "$((2#$original_chars))")
        xor_res=$(( $dec_chars ^ $dec_key ))
        echo $xor_res | awk '{ printf("%c", $0); }' >> $plain_file
    done
fi
echo "Decryption Process Done Successfully!"
else
    echo "$CN Doesn't Exist!!!"
fi
```

Fig 2.5

Procedure:

EX1

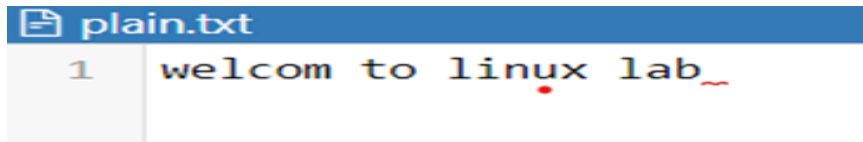


Fig2.1

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
e
Please input the name of the Plain text file: plain.txt
Key: Decimal = 80 | Binary = 01010000
Please input the name of the Cipher text file: cipher.txt
Encryption Done Successfully!
```

Fig2.2

1. The program will ask user to choose between encryption and decryption (e.g. e for encryption and d for decryption)

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
```

2. If the user enters 'e':

a. The program should print on the screen "Please input the name of the plain text file"

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
e
Please input the name of the Plain text file: plain.txt
```

b. The program should raise an error if the file contains any non-alphabet characters

```
plain.txt
1 welcom to linux lab 6
```

```
Please Select one
-e : Encryption
-d : Decryption
e
Please input the name of the Plain text file: plain.txt

Error: file contains non-alphabet characters!
~$
```

c. After that, the program must print the value of the key in decimal and binary

```
Key: Decimal = 80 | Binary = 01010000
```

d. Ask user to input the name of the cipher text file

```
Please input the name of the Cipher text file: cipher.txt
```

e. The program will write the generated cipher text on the cipher file

```
01110010010100111100001100110011111100111101001100000111010000100000010000001111100011100100111110001101010010100000100000011111000011000100110010001100000101
```

```
Encryption Done Successfully!
```

3. If the user enters 'd':

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
d
Please input the name of the Cipher text file: cipher.txt
key: Decimal = 80 | Binary = 01010000
Please input the name of the Plain text file: plain.txt
Decryption Proccess Done Successfully!
```

a. The program should print on the screen “Please input the name of the cipher text file”

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
d
Please input the name of the Cipher text file: cipher.txt
```

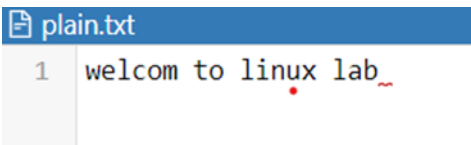
b. After that, the program must print value of the key

```
d
Please input the name of the Cipher text file: cipher.txt
key: Decimal = 80 | Binary = 01010000
```

c. Ask user to input the name of the plain text file

```
Please input the name of the Plain text file: plain.txt
```

d. The program will write the generated plain text on the plain text file



```
plain.txt
1 welcom to linux lab
```

```
Decryption Proccess Done Successfully!
```

EX2

```
MY code IS run
```

The code of encryption

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
e
Please input the name of the Plain text file: plain.txt

Key: Decimal = 53 | Binary = 00110101

Please input the name of the Cipher text file: cipher.txt

Encryption Done Successfully!
```

Result

```
100001111110001100101000101100101101001010001010100000101010100011100011101100110010100010111010000000100101101010101000101010011
```

The code of decryption

```
~$ ./project.txt
Please Select one
-e : Encryption
-d : Decryption
d
Please input the name of the Cipher text file: cipher.txt
key: Decimal = 53 | Binary = 00110101
Please input the name of the Plain text file: plain.txt
Decryption Process Done Successfully!
~$
```

Result

MY code IS run

We put the number

plain.txt

1 MY code3 IS run

Result

Please Select one

-e : Encryption

-d : Decryption

e

Please input the name of the Plain text file: plain.txt

Error: file contains non-alphabet characters!

~\$

Conclusion

In this project, we have learnt how to do simple program that does simple encryption/decryption algorithm for text messages with only alphabet characters. Also we got to know basics of using shell script in a way to get benefit of this in our lives in general and in many sides in special.

