



PES UNIVERSITY, BENGALURU

VI SEM

SESSION: JANUARY-MAY, 2020

UE17CS356B –APPLIED CRYPTOGRAPHY

MINI PROJECT REPORT

On

“Implementation of One Time Pad”

Submitted by

AARYA ARUN

PES1201700009

S MANEESHA

PES1201700024

INDU RALLABHANDI

PES1201700795

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PES UNIVERSITY

(ESTABLISHED UNDER KARNATAKA ACT NO. 16 OF 2013)

100 FEET RING ROAD, BENGALURU – 560 085, KARNATAKA, INDIA

What is a One Time Pad?

The One Time Pad is an encryption technique that cannot be cracked. A one time pre-shared key is required, which has to be of the same length or greater length than the message. A key of greater length is preferred as it will not provide any hint to the outsider of the length of the plaintext thus securing the data. It must be ensured that the key is truly random and is confidential between the sender and the receiver from which uncrackable security can be achieved. This key is destroyed by the sender and the receiver after use.

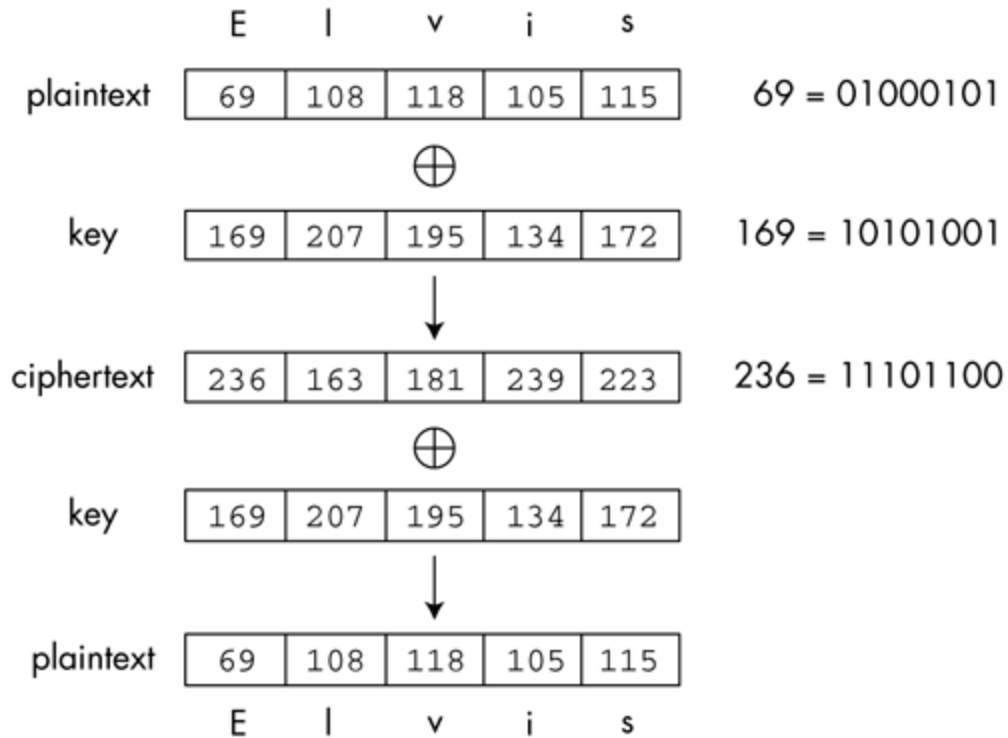
The Cipher Mechanism

Encryption:

- Every character from the plaintext and the key are converted into their 8-bit binary equivalents.
- Then, bitwise XOR operation is performed between the corresponding bits of the plaintext and key which provides a result with the number of bits equal to that of the binary equivalent of the key. This is the ciphertext.
- On grouping every 8-bits of the binary result obtained(ciphertext) and converting them back to characters we arrive at a readable form of the ciphertext.
- This step is essential in this project because the ciphertext obtained need not always have readable characters. Thus, we are adding 33 with every character obtained for the ease of readability.

Decryption:

- Every character from the ciphertext and the key are converted into their 8-bit binary equivalents.
- Then, bitwise XOR operation is performed between the corresponding bits of the ciphertext and key which provides a result with the number of bits equal to that of the binary equivalent of the key. The result obtained is the plaintext that was encoded.
- On grouping every 8-bits of the binary plaintext and converting them back to characters we arrive at a readable form of the plaintext. We subtract 33 from every character for this.



Mechanism of the One Time Pad

The code

The code has been uploaded on GitHub and can be accessed through the following link:

GITHUB LINK: <https://github.com/aarya-arun/Cryptography/tree/master/One%20Time%20Pad>

*The code has been written in C++ with minimal UI, where the user inputs their text as well as the key in the terminal. The github code has been provided with comments for better understanding.

Input

- Mode: Indicates whether you are encrypting or decrypting the given message. (as there is a difference in the way the message is displayed depending on the function being called)
- Text: This could be your plaintext or ciphertext depending on the mode specified.
- Key: The key used for encryption/decryption.

Output

- Text: Outputs the given text.
- Encrypted/decrypted text: Based on the mode, the encrypted/decrypted text is printed.

Constraints

- The size of the message cannot be more than 10000 characters.
- We cannot give texts that include newline characters.

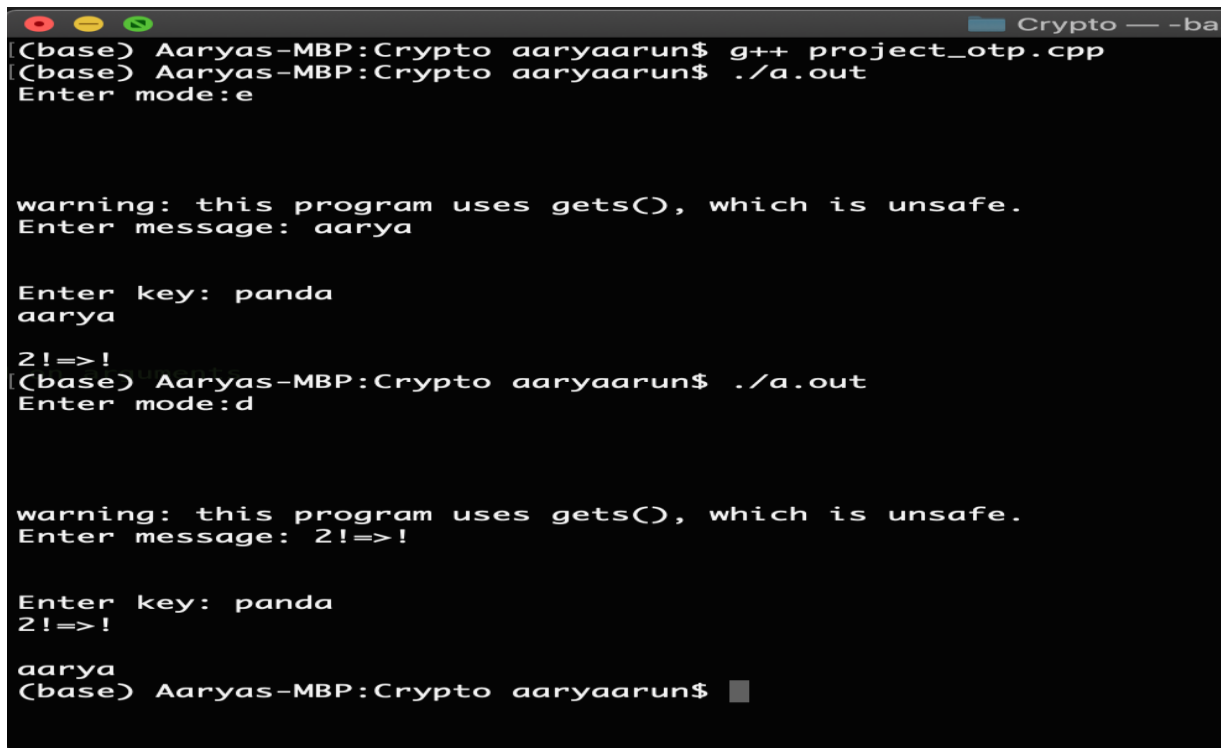
Mechanism

- The user inputs a mode ('e' for encryption mode and 'd' for decryption mode).
- The code checks whether an acceptable mode has been provided as input.
- The user provides the inputs in the form of strings (signed characters).
- The code checks whether the key length and message length are compatible (i.e. they match).
- In order to perform smooth bitwise operation, the characters are converted to unsigned characters.
- A bitwise XOR operation is performed on the key and text provided to obtain the encrypted/decrypted text.
- The output is shown as text for the ease of readability. (eliminating the possibility of having unprintable characters)

Merits

- Provides the security of a one time pad.
- Has encryption and decryption mode which is accurate given the constraints.
- Does work when special characters are included in the key or plaintext.
- Simple logic yet secure.

Screenshots



```
Crypto — -ba
(base) Aaryas-MBP:Crypto aaryaarun$ g++ project_otp.cpp
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: aarya

Enter key: panda
aarya

Z!=>!
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d

warning: this program uses gets(), which is unsafe.
Enter message: Z!=>!

Enter key: panda
Z!=>!

aarya
(base) Aaryas-MBP:Crypto aaryaarun$
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: aarya

Enter key: @bcd$
aarya

B$2>f
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d

warning: this program uses gets(), which is unsafe.
Enter message: B$2>f

Enter key: @bcd$
B$2>f

aarya
(base) Aaryas-MBP:Crypto aaryaarun$ █
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: aarya1

Enter key: indiaa
aarya1
15th
)071!q
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d 15th of text. Please try again.);

warning: this program uses gets(), which is unsafe.
Enter message: )071!q

Enter key: indiaa
)071!q

aarya1
(base) Aaryas-MBP:Crypto aaryaarun$ █
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: 121212

Enter key: ababab
121212
qqqqqq
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d

warning: this program uses gets(), which is unsafe.
Enter message: qqqqqq

Enter key: ababab
qqqqqq
121212
(base) Aaryas-MBP:Crypto aaryaarun$
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: $%^&*

Enter key: panda
$%^&*
ueQc1
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d

warning: this program uses gets(), which is unsafe.
Enter message: ueQc1

Enter key: panda
ueQc1
$%^&*
(base) Aaryas-MBP:Crypto aaryaarun$
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: MS Dhoni

Enter key: InduAnuhya
MS Dhoni
no length
%^eRJ"<"
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d 9th of text. Please try again.");

warning: this program uses gets(), which is unsafe.
Enter message: %^eRJ"<"

Enter key: InduAnuhya
%^eRJ"<"

MS Dhoni
(base) Aaryas-MBP:Crypto aaryaarun$
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: Maneesha

Enter key: indu aarya
Maneesha
no length
E0+1f3*4
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d 9th of text. Please try again.");

warning: this program uses gets(), which is unsafe.
Enter message: E0+1f3*4

Enter key: indu aarya
E0+1f3*4

Maneesha
(base) Aaryas-MBP:Crypto aaryaarun$
```



```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: indu

Enter key: 1234
indu
me length
y}xb
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d
length of text. Please try again.");

warning: this program uses gets(), which is unsafe.
Enter message: y}xb

Enter key: 1234
y}xb

indu
(base) Aaryas-MBP:Crypto aaryaarun$
```

```
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: aarya

Enter key: cake1
aarya
me length
#!:=q
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d
length of text. Please try again.");

warning: this program uses gets(), which is unsafe.
Enter message: #!:=q

Enter key: cake1
#!:=q

aarya
(base) Aaryas-MBP:Crypto aaryaarun$
```

```

(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:e

warning: this program uses gets(), which is unsafe.
Enter message: aarya arun

Enter key: cake123456789
aarya arun
#!:=q3sgay
(base) Aaryas-MBP:Crypto aaryaarun$ ./a.out
Enter mode:d

warning: this program uses gets(), which is unsafe.
Enter message: #!:=q3sgay

Enter key: cake123456789
#!:=q3sgay
aarya arun
(base) Aaryas-MBP:Crypto aaryaarun$

```

Further scope of project:

- To allow input in the form of a text file and compute cipher text even if the message contains newline characters and large message sizes.

Contributions:

| | |
|------------------------------|-----------------------|
| Code implementation | Aarya, Maneesha, Indu |
| Code execution and debugging | Aarya, Maneesha, Indu |
| Report | Aarya, Maneesha, Indu |
| PPT | Aarya, Maneesha, Indu |
