**Data Structures and Algorithms J-Component Review-3**

**PROJECT REPORT**

Rushil Goel (19BCE0361)

Adarsh Mishra(19BCE0437)

Aashish Sarraf(19BCE0433)

**Abstract**

Cyber security is a crucial part in today’s world. Be it a multi-national company or just a regular citizen, everyone is looking for ways to safeguard their personal information on the internet. This is not limited to the internet alone; it can be applicable for any digitalized information. There are numerous needs to securing data. For example, denying access to private chats, classified documents, and passwords. This information can be secured in a variety of ways. One can be using firewalls so hackers can’t get to such information in the first place. The other very important method being Encryption. Encryption is a method of converting a normal file/text in a non-human readable way. This prevents unauthorized access to such texts. If a hacker gets into a database, they will gain access to all private information there. This could lead to substantial loss of trust and money of the affected organization.

**Keywords**

* Cyber Security
* Firewalls
* Hackers
* Encryption
* Cryptography
* Decryption
* Ransomware
* Cipher
* Information security
* Ransomware families

**Introduction**

**Problem statement:** Cryptographic Algorithms and their applications

**Main Objectives:** Implementation of encryption algorithms and comparing their Execution Time, Response Time, Popularity in ransomware families and Finally concluding which is more secure – symmetric or asymmetric cryptography algorithms

**Application:** Encrypting any highly confidential information like passwords, messages, documents and so on.

**Details about Related Work**

We started our project by defining keywords like encryption and types of encryption namely, Symmetric and Asymmetric encryption. There are various encryption and decryption algorithms and, in our project, we have implemented three of them. They vary from most common ones to the unbreakable cryptography algorithms. We implemented RSA, RC4 and OTP or one-time-pad also known as vernam cipher. We included pseudo code of the implemented algorithms to get a better understanding about their working and further implemented in common programming languages like C and Python. We included advantages and disadvantages of each of them and a brief information about them as well. We finally compared them on basis of response time, execution time and popularity in ransomware families. At the end we concluded which is more secure - symmetric or asymmetric encryption algorithms.

**Motivation on the Project**

We were motivated to take up this project because information security is a very big domain in the world of computer science and we are particularly fascinated by it because of the growing demand in the information security domain. All of us should have a basic idea of how internet security works. That will help us better defend against potential attacks and will foster a responsible behavior when it comes to online activity. Having knowledge about basic encryption algorithms will help us develop better encryption algorithms and will enable us to understand their implementation is the real world. Nowadays every organization be it a Multinational corporation or small-scale industry, everyone needs to make their data safe and secure and securing digital information is encryption. We even tried to compare the implemented algorithms on basis of their response time and execution time and graphed out the same. We even tried to conclude the popular encryption algorithms in ransomware families with help of a pie chart. Encryption finds its application in numerous places in todays world be it signing up for a course on a website or online banking or just securing the simple texts. A domain with such wide application in the industry fascinated us to take up this project and we could conclude that we learnt a lot from the same.

**Proposed Work**

We started this project by learning the basic encryption algorithms like RSA and RC4 and OTP and later expanded our research by comparing them based on various factors including Type, Response time, Execution time and finally we depicted the most popular ransomware cryptography algorithms. We finally briefed about the security level of both symmetric and asymmetric algorithms and explained in detail which is more secure. We implemented various encryption algorithm using programming languages like C and Python and differentiated between their merits and demerits. One of our implemented algorithms was OTP or One-time-pad or vernam cipher which is regarded as the truly unbreakable encryption algorithm. It was used extensively during world war – II and till today is unbreakable. Many of the implemented algorithms are still being used in today’s world and helped us a lot to learn about data encryption and its necessity in digital world.

**Coding**

* **RSA**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<math.h>

#include<string.h>

long int p,q,n,t,flag,e[100],d[100],temp[100],j,m[100],en[100],i;

char msg[100];

int prime(long int);

void ce();

long int cd(long int);

void encrypt();

void decrypt();

void main() {

printf("\nENTER FIRST PRIME NUMBER\n");

scanf("%d",&p);

flag=prime(p);

if(flag==0) {

printf("\nWRONG INPUT\n");

getch();

exit(1);

}

printf("\nENTER ANOTHER PRIME NUMBER\n");

scanf("%d",&q);

flag=prime(q);

if(flag==0||p==q) {

printf("\nWRONG INPUT\n");

getch();

exit(1);

}

printf("\nENTER MESSAGE\n");

fflush(stdin);

scanf("%s",msg);

for (i=0;msg[i]!=NULL;i++)

m[i]=msg[i];

n=p\*q;

t=(p-1)\*(q-1);

ce();

printf("\nPOSSIBLE VALUES OF e AND d ARE\n");

for (i=0;i<j-1;i++)

printf("\n%ld\t%ld",e[i],d[i]);

encrypt();

decrypt();

getch();

}

int prime(long int pr) {

int i;

j=sqrt(pr);

for (i=2;i<=j;i++) {

if(pr%i==0)

return 0;

}

return 1;

} void ce() {

int k;

k=0;

for (i=2;i<t;i++) {

if(t%i==0)

continue;

flag=prime(i);

if(flag==1&&i!=p&&i!=q) {

e[k]=i;

flag=cd(e[k]);

if(flag>0) {

d[k]=flag;

k++;

}

if(k==99)

break;

}

}

}

long int cd(long int x) {

long int k=1;

while(1) {

k=k+t;

if(k%x==0)

return(k/x);

}

} void encrypt() {

long int pt,ct,key=e[0],k,len;

i=0;

len=strlen(msg);

while(i!=len) {

pt=m[i];

pt=pt-96;

k=1;

for (j=0;j<key;j++) {

k=k\*pt;

k=k%n;

}

temp[i]=k;

ct=k+96;

en[i]=ct;

i++;

}

en[i]=-1;

printf("\nTHE ENCRYPTED MESSAGE IS\n");

for (i=0;en[i]!=-1;i++)

printf("%c",en[i]);

}

void decrypt() {

long int pt,ct,key=d[0],k;

i=0;

while(en[i]!=-1) {

ct=temp[i];

k=1;

for (j=0;j<key;j++) {

k=k\*ct;

k=k%n;

}

pt=k+96;

m[i]=pt;

i++;

}

m[i]=-1;

printf("\nTHE DECRYPTED MESSAGE IS\n");

for (i=0;m[i]!=-1;i++)

printf("%c",m[i]); }

* **RC4**

import sys

import binascii

def key\_to\_bytes(key):

keyinput = list(range(len(key)))

for c in range(0, len(key)):

keyinput[c] = ord(key[c])

return keyinput

def make\_stream\_source(key):

keylen = len(key)

stream = list(range(256))

j = 0

for i in range(0, 256):

j = (j + stream[i] + key[i % keylen]) % 256

temp = stream[i]

stream[i] = stream[j]

stream[j] = temp

return stream

def gen\_stream(key, source):

i = 0

j = 0

while True:

i = (i + 1) % 256

j = (j + source[i]) % 256

temp = source[i]

source[i] = source[j]

source[j] = temp

k = source[(source[i] + source[j]) % 256]

yield k

def main():

if len(sys.argv) == 2:

keyinput = sys.argv[1]

else:

keyinput = input('enter a key to use: ')

plaintext = input('enter a string for encrypting: ')

if len(keyinput) < 5 or len(keyinput) > 32:

print("Key must be between 5 bytes and 32 bytes long")

exit()

keyinput = key\_to\_bytes(keyinput)

print('key:', keyinput)

stream\_source = make\_stream\_source(keyinput)

print('stream source:', stream\_source)

stream = gen\_stream(keyinput, stream\_source)

for char in plaintext:

sys.stdout.write("%02X" % (ord(char) ^ next(stream)))

print()

if \_\_name\_\_ == '\_\_main\_\_':

main()

* **OTP**

// ENCRYPTION

#include<stdio.h>

#include<string.h>

#include<ctype.h>

main() { //All the text which ever entered is converted to upper and without spaces

int i,j,len1,len2,numstr[100],numkey[100],numcipher[100];

char str[100],key[100],cipher[100];

printf("Enter a string text to encrypt\n");

gets(str);

for(i=0,j=0;i<strlen(str);i++)

{

if(str[i]!=' ')

{

str[j]=toupper(str[i]);

j++;

}

}

str[j]='\0';

//obtaining numerical plain text ex A-0,B-1,C-2

for(i=0;i<strlen(str);i++)

{

numstr[i]=str[i]-'A';

}

printf("Enter key string of random text\n");

gets(key);

for(i=0,j=0;i<strlen(key);i++)

{

if(key[i]!=' ')

{

key[j]=toupper(key[i]);

j++;

}

}

key[j]='\0';

//obtaining numerical one time pad(OTP) or key

for(i=0;i<strlen(key);i++)

{

numkey[i]=key[i]-'A';

}

for(i=0;i<strlen(str);i++)

{

numcipher[i]=numstr[i]+numkey[i];

}

//To loop the number within 25 i.e if addition of numstr and numkey is 27 then numcipher should be 1

for(i=0;i<strlen(str);i++)

{

if(numcipher[i]>25)

{

numcipher[i]=numcipher[i]-26;

}

}

printf("One Time Pad Cipher text is\n");

for(i=0;i<strlen(str);i++)

{

printf("%c",(numcipher[i]+'A'));

}

printf("\n"); }

// DECRYPTION

#include<stdio.h>

#include<string.h>

#include<ctype.h>

main()

{

//All the text which ever entered is converted to upper and without spaces

int i,j,len1,len2,numstr[100],numkey[100],numcipher[100];

char str[100],key[100],cipher[100];

printf("Enter an Encrypted string text to Decrypt\n");

gets(str);

for(i=0,j=0;i<strlen(str);i++)

{

if(str[i]!=' ')

{

str[j]=toupper(str[i]);

j++;

}

}

str[j]='\0';

//obtaining numerical plain text ex A-0,B-1,C-2

for(i=0;i<strlen(str);i++)

{

numstr[i]=str[i]-'A';

}

printf("Enter key string of random text\n");

gets(key);

for(i=0,j=0;i<strlen(key);i++)

{

if(key[i]!=' ')

{

key[j]=toupper(key[i]);

j++;

}

}

key[j]='\0';

//obtaining numerical one time pad(OTP) or key

for(i=0;i<strlen(key);i++)

{

numkey[i]=key[i]-'A';

}

for(i=0;i<strlen(str);i++)

{

numcipher[i]=numstr[i]-numkey[i];

//changed from + to - for decryption

if(numcipher[i]<0)

{

numcipher[i]=numcipher[i]+26; //If cipher is negative we have to add 26

}

numcipher[i]=numcipher[i]%26;//To loop within 1 to 26 for alphabets from A-Z

}

printf("Decrypted One Time Pad Cipher text is\n");

for(i=0;i<strlen(str);i++)

{

printf("%c",(numcipher[i]+'A'));

}

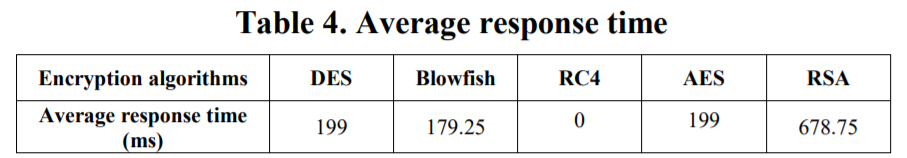
printf("\n");

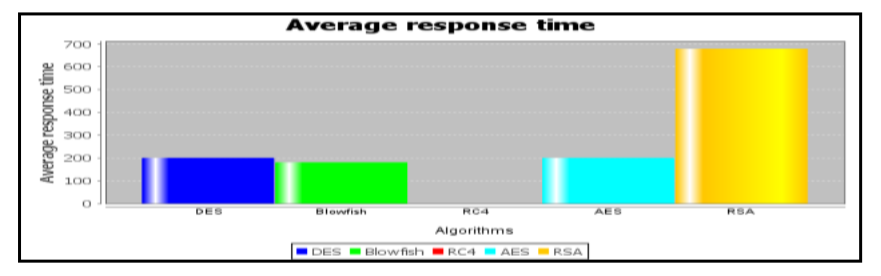
}

**Conclusion**

* **Response Time Comparison**

The results presented in Figure show that RC4 has better behavior compared to the other four algorithms namely, DES, Blowfish, RC4, AES and RSA. From the Table, we can estimate an average gain for the RC4 to 100% compared to DES, Blowfish, AES and RSA. .The efficiency of the RC4 algorithm in terms of response time, space memory and security becomes very high once we change the size of the rules base. While RSA records highest average record time which is attributed to its complex mathematical calculations and functions, AES, DES and Blowfish had almost same response time.

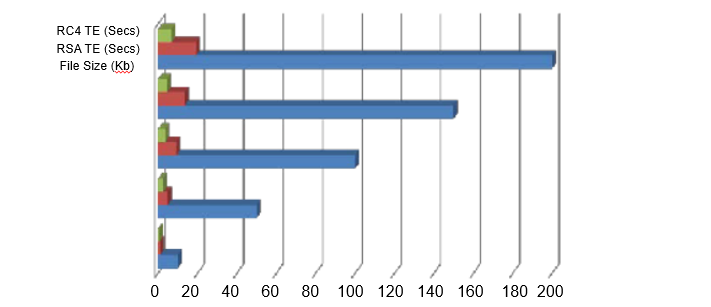




* **Execution Time Comparison**

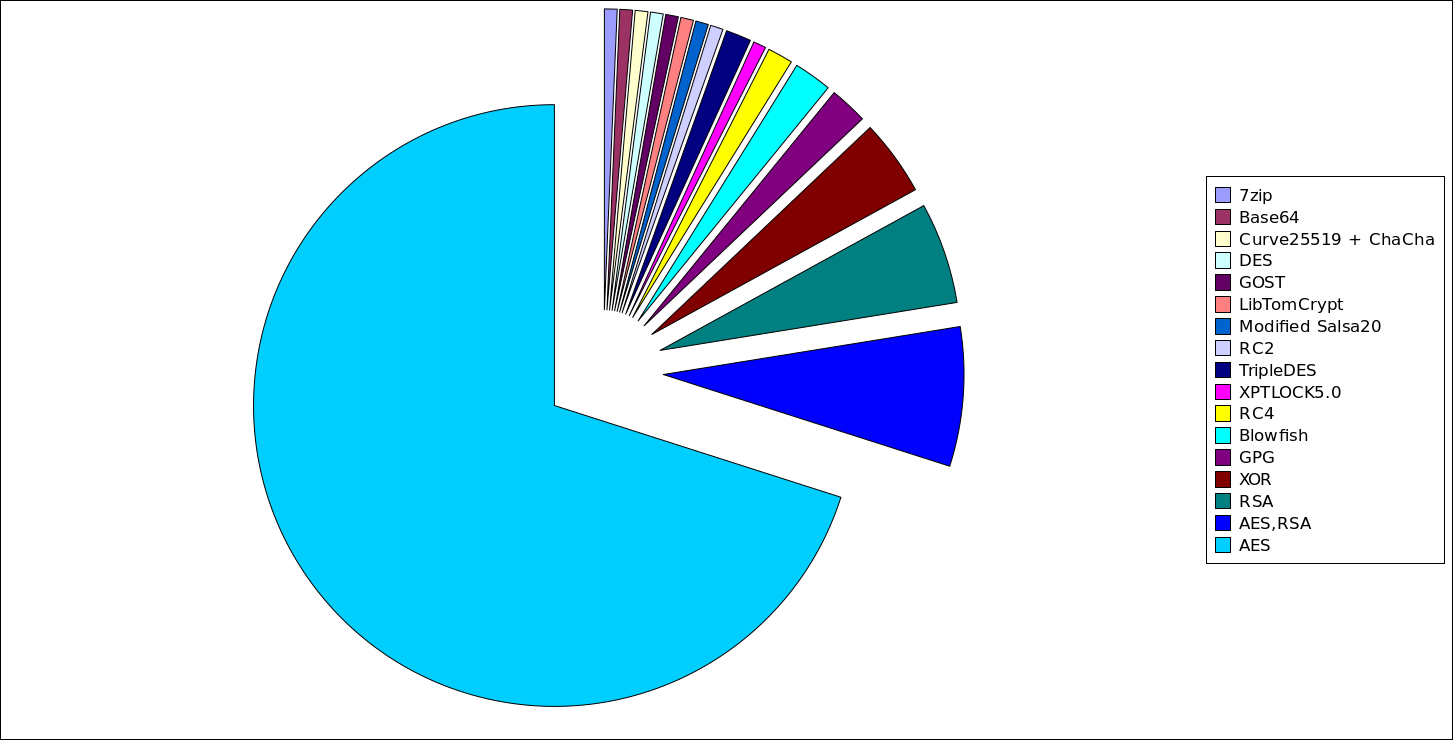
RC4 using five text files and five graphic files of different sizes of 10, 50, 100, 150, and 200 kilobyte respectively. The major factor considered for measuring the performance of the algorithms (RSA and RC4) is the speed of execution using time of execution (TE) as parameter for the evaluation. The RSA TE and RC4 TE were measured for both text file, and the graphic file.

CONCLUSION - From the evaluations of both RSA and RC4 algorithms and the comparison presented, it was concluded that between RC4 and RSA, RSA is the most reliably secure algorithm. However, the RC4 seems to be faster in encryption and decryption process but rather less secure, as it can be broken using a relatively inexpensive device in a short time when compare with RSA.



* **Most Popular cipher in ransomware families**

Clearly AES is the most widely used cipher in ransomware families. **AES** data encryption is a more mathematically efficient and elegant cryptographic algorithm, but its main **strength** rests in the option for various key lengths. **AES** allows you to choose a 128-bit, 192-bit or 256-bit key, making it exponentially stronger than the 56-bit key of DES. Originally adopted by the federal government, **AES encryption** has become the industry standard for data security. RSA has next big chunk in the popularity chart. This could be due to its existence in cipher world since beginning and its simplicity. It works on mathematical operations and still is very difficult to break into. RC4 with a relatively smaller chunk is next and reasons of its decreased popularity could be because it finds its application in software encryption and is not regarded is very secure. Although it is worth mentioning that it is a very fast encryption algorithm and have efficient response time and execution time.



* **Which is more secure – Symmetric or Asymmetric encryption?**

In this Project we implemented 3 cryptography algorithms namely, RSA, RC4 and OTP. While RSA was asymmetric cryptography algorithm, RC4 and OTP are symmetric cryptography algorithm. After understanding their implementation and application in real world, We are now capable of concluding Which is more secure – Symmetric or Asymmetric encryption. This is a difficult question to answer. Most people believe that asymmetric encryption is more secure since it has both a public and a private key. But comparing the strength and resistance to attack of symmetric and asymmetric encryption isn’t that easy. What is important here is the context. Symmetric encryption is better used when trying to share information between a smaller number of people. It is easier to use and understand, so there are lower chances of the information being misinterpreted. On the other hand, asymmetric encryption work way better on large groups of people (such as the internet). This makes it hard to say that “asymmetric is better than symmetric” or vice-versa. While symmetric encryption might be the best fit for certain situations, in other cases asymmetric encryption might be the better choice.

Most of today’s systems (such as SSL or TLS) use a combination of both symmetric and asymmetric encryption, as well as other algorithms.

**Referred Papers**

* <https://www.researchgate.net/publication/280804925_RSA_and_RC4_Cryptosystem_Performance_Evaluation_Using_Image_and_Text_File>
* <https://dl.acm.org/doi/pdf/10.1145/2816839.2816919?download=true>