

Encryption and Decryption using RSA



Group Members:

- Muhammad Zunique (20K-0145)
- Ammar Siddiqui (20K-0177)
- Muhammad Muaz Zaffar (20K-0363)

Computer Organization and Assembly Language

Course Code: EE-2003

National University of Computer and Emerging Sciences – FAST

INTRODUCTION:

Long ago, when some person wants to deliver the message from one place to another by making sure that the message should not leak the information. So, for that different Encryption and Decryption algorithms were being introduced amongst which RSA is considered as one of most prominent as it is wholly based on prime numbers and till yet in today's world no generalize formula has been invented that can decode prime numbers logic. And this logic of prime and large arithmetic values makes it more beneficial for encryption and decryption. There public and private keys that are used for encryption and decryption purpose respectively maintaining Asymmetric property of keys and modulo function including Euclidean.

Literature Review:

The background of this topic in computer science generally referred as CRYPTOGRAPHY as network security is pretty much necessary via developing it in today's modern era where hacking other's databases private information is just quiet easy, so one have to make sure that his security system must be equally strong via developing team. So these Encryption techniques provides the general idea that how on large network and datasets we can perform operation to secure our data. Mathematical calculations is a major part for doing RSA technique as we encode message to number and do manipulation on those numbers and the decode those using defined methodology

Problem Statement:

You must have seen when you open a chat on Whatsapp, a message shows at the top that chats are end to end encrypted what's that? That is what basically Encryption and Decryption is all about, securing messages from one end to another end. The main purpose is that the idea of message should not be leaked so for that different techniques are used amongst which we chose RSA as it is one of prominent

and not that much easy to decode as if consider some commonly used such as Ceaser cipher, Shift Cipher or Hill cipher as they do not require much effort to be decrypted.

Methodology(Tools and Techniques):

- 🛠 Microsoft Visual Studio 2019
- 🛠 IRVINE MASM library
- 🛠 Relevant methods explained in code for calculating values for ciphering and Deciphering.
- 🛠 Plain text to encrypted text message
- 🛠 Encrypted Text to plain/ Decrypted Text message

Detailed Design and architecture:

First take input two prime numbers by means all which rest of values are dependent. Then the public and private keys are generated on which encoding and decoding is dependent.

PUBLIC KEY : (e,n) // $n = p * q$ and $k = (p-1)(q-1)$ and $1 < e < k$

PRIVATE KEY : (d,n) // where is inverse of e

The calculations of following components are defined in code explicitly.

$$C = m^e \bmod(n) \qquad m = C^d \bmod(n)$$

These are two formulae used for encryption and Decryption.

Implementation, Testing and Code:

```
INCLUDE Irvine32.inc
INCLUDE Macros.inc
.data
    ;/--Two prime Numbers--//
    p DWORD ?
    q DWORD ?
    ;/-----//

    n DWORD ?
    k DWORD ?
    temp2 DWORD 1

    ;/---Encription and decryption key--//
    e DWORD ? ; is the encryption key
    d DWORD ? ; is the decryption key
    ;/-----//

    temp DWORD ?
    save_eax DWORD ?
    buffer DWORD 500
    message BYTE 500 DUP(?)
    char_entered DWORD ?

    temp_array SBYTE 500 DUP(?)
    encrypted BYTE 500 DUP(?)
    decrypted BYTE lengthof encrypted DUP(?)

    is_prime PROTO, input:DWORD

;/-----MAIN PROGRAM-----//
.code
    main PROC

        ;/---clearing resgitters---//
        xor eax,eax
        xor ebx,ebx
        xor ecx,ecx
        xor edx,edx
        ;/-----//

        ;/-----Formatting of code Deisgn-----//
        menu:
```

ENCRYPTION AND

```

        mWrite"
DECRYPTION USING RSA ALGORITHM "
        call crlf
        mWrite"

```

```

===== "
        call crlf
        call crlf
        call crlf

```

```

;///----Taking INPUT----//
call take_input
mov eax,p
cmp eax,q
je invalid
;///-----//

```

```

;///----check for isPrime----//
invoke is_prime,p
cmp eax,1
je Invalid
invoke is_prime,q
cmp eax,1
je Invalid
;///-----//

```

```

;///----if isprime successful calculate follweoing-----//
call gen_n
mov n,eax
call gen_k
mov k,eax
sub eax,1
mov e,eax
;///-----//

```

```

;///-----Three major Functions-----//
call show_values
call encrypt
call decrypt
;///-----//

```

```

jmp endd

```

```

;///-----Labels-----//
Invalid:
        mwrite "Invalid input "

```

```

        endd:
            exit
        ;//-----//
main ENDP

;//---Check for whether the numbers entered are prime or not-----//
is_prime PROC , input:DWORD
    mov ebx,2
    check_loop:
        xor edx,edx
        mov eax,input
        div ebx
        cmp edx,0
        je n_prime
        inc ebx
        cmp ebx,input
        jne check_loop

    mov eax,0          ; 0 represents is prime
    ret
    n_prime:
        mov eax,1      ; 1 represents not prime
    ret
is_prime ENDP

;//-----//
---//

;//-----Display the calculated values-----//
show_values PROC
    call crlf
    call crlf
    mWrite"Calculated values are : "
    call crlf
    mWrite"===== "
    call crlf
    call crlf
    mov eax,p
    mwrite "The value of first prime number is: "
    call writedec
    call crlf
    mov eax,q
    mwrite "The value of second prime number is: "
    call writedec

```

```

    call crlf
    mwrite "The value of n : "
    mov eax,n
    call writedec
    call crlf
    mwrite "The value of k : "
    mov eax,k
    call writedec
    call crlf
    mwrite "The value of e : "
    mov eax,e
    call writedec
    call crlf
    call crlf

    ret
show_values ENDP
;///-----//

;///-----converting plain text to encrypted text-----//
encrypt PROC

    xor edx,edx
    xor esi,esi
    xor edi,edi

    ;///---for calculating length of string so that loop can be calculates---//
    mov ecx,char_entered

;///---Loop1 is running ascii equivalent codes of every character of string----//
l1:
    mov bl,message[esi]
    movzx ebx,bl
    sub ebx,60h    ;///--subtracting 60 as per logic--//
    mov eax,1
    mov temp,ecx
    mov ecx,e

    ;///---in LOOP2 multiplying it as per formula---//
l2:
    imul eax,ebx
    xor edx,edx
    mov save_eax,eax
    cdq

```

```

        idiv n    ;//---for getting mod calculation---//
        mov eax,save_eax
        mov eax,edx
    LOOP 12

    mov temp_array[esi],al
    add eax,60h    ;//adding that 60 again that we first subtracted so that correct encryptes string is obtained
    mov encrypted[edi],al
    inc edi
    inc esi
    mov ecx,temp
    ;//---clearing registers---//
    xor eax,eax
    xor ebx,ebx
LOOP 11

;//----show the encrypted string----//

mWrite"-----ENCRYPTED CONTEXT-----"
call crlf
call crlf
mov esi,offset encrypted
mov ecx,char_entered
mWrite"Encrypted Key values are : "
14:
    mov al,[esi]
    call writeInt
    mWrite" "
    inc esi
LOOP 14

call crlf
call crlf
mov esi,offset encrypted
mov ecx,char_entered
mwrite "The encrypted data is : "
13:
    mov al,[esi]
    call writeChar
    mWrite" "
    inc esi
loop 13
call crlf
call crlf
ret
encrypt ENDP

```



```
;//-----//
```

```
;//-----Take Input prime Numbers-----//
```

```
take_input PROC
```

```
    mwrite"Enter any two distinct prime numbers(whose product is less than 128): "
```

```
    call readInt
```

```
    mov p,eax
```

```
    call readInt
```

```
    mov q,eax
```

```
    call crlf
```

```
    mwrite "Enter the message want to encrypt : "
```

```
    mov ecx,buffer
```

```
    mov edx,offset message
```

```
    call readString
```

```
    mov char_entered,eax
```

```
    ret
```

```
take_input ENDP
```

```
;//-----//
```

```
;//----- Generate k -----//
```

```
gen_k PROC
```

```
    mov eax,p
```

```
    mov ebx,q
```

```
    sub eax,1
```

```
    sub ebx,1
```

```
    imul eax,ebx
```

```
    ret
```

```
gen_k ENDP
```

```
;//-----//
```

```
;//----- Generate d -----//
```

```
gen_d PROC
```

```
    xor eax , eax
```

```
    xor ebx , ebx
```

```
    mov edx , 0
```

```
Lb1:
```

```
    mov eax , n    ;moving value of  n in eax
```

```
    mul temp2      ; multiplying it with  a variable defined above and taking floor value
```

```
    add eax , 1
```

```

        mov ebx , e
        div ebx      ; for getting eax -> quotient & edx ->
remainder
        cmp edx ,0 ;if remainder zero jump to label and set that particular value present in eax to 'd'
        jmp setValue
        add temp2 , 1 ; else loop until zero
        JMP Lb1

setValue:
        mov d , eax

        ret
gen_d ENDP
;///-----//

;///----- Generate n -----//
gen_n PROC
        mov eax,p
        mov ebx,q
        imul eax,ebx
        ret
gen_n ENDP
;///-----//

;///-----Decryption Function-----//
decrypt PROC

        ;///---setting values---//
        mov esi,offset temp_array
        mov edi,offset decrypted
        mov ecx,char_entered

outerloop:
        ;///--clearing registers--//
        xor eax,eax
        xor ebx,ebx

        mov al,[esi]
        movsx eax,al
        mov ebx,1
        mov temp,ecx
        mov ecx,e

```

```

        innerloop:
            xor edx,edx
            imul ebx,eax          ;k = ebx
            mov save_eax,eax
            mov eax,ebx
            cdq
            idiv n              ;quotient gets in eax and remainder in edx
            mov ebx,edx
            mov eax,save_eax
        LOOP innerloop
        mov eax,ebx
        add ebx,96
        mov [edi],bl
        mov ecx,temp
        inc esi
        inc edi
    LOOP outerloop
    call crlf

;///---Displaying Decrypted Text---//

mWrite"-----DECRYPTED CONTEXT-----"
call crlf
call crlf
mov esi,offset decrypted
mov ecx,char_entered
mwrite "The decrypted key values are : "
L1:
    mov al,[esi]
    call writeInt
    inc esi
loop L1
call crlf
call crlf

mwrite "The decrypted message is : "
mov esi,offset decrypted
mov ecx,char_entered
L5:
    mov al,[esi]
    call writechar
    inc esi
loop L5
call crlf
call crlf
ret

```

```

decrypt ENDP
;//-------//

END main

;//-------MAIN ENDED-----//

```

Results:

SAMPLE 1:

The image shows a Visual Studio environment with two windows. The top window is the application 'C:\Users\AA\source\repos\qwert\Debug\qwert.exe', which displays the title 'ENCRYPTION AND DECRYPTION USING RSA ALGORITHM'. It prompts the user to 'Enter any two distinct prime numbers (whose product is less than 128):'. The user enters '7' and '11'. Then it prompts 'Enter the message want to encrypt :', and the user enters 'Hello World Coal'.

The bottom window is the 'Microsoft Visual Studio Debug Console'. It shows the same input message 'Hello World Coal'. Below this, it displays 'Calculated values are :'. The calculated values are: 'The value of first prime number is: 7', 'The value of second prime number is: 11', 'The value of n : 77', 'The value of k : 60', and 'The value of e : 59'. It then shows the 'ENCRYPTED CONTEXT' with 'Encrypted Key values are : +35 +127 +141 +141 +132 +25 +36 +132 +126 +141 +154 +25 +88 +132 +97 +141 +25' and 'The encrypted data is : # º ï ï ä ↓ \$ ä ~ ï Ü ↓ X ä a ï ↓'. Finally, it shows the 'DECRYPTED CONTEXT' with 'The decrypted key values are : -184-155-148-148-145-224-169-145-142-148-156-224-189-145-159-148-224' and 'The decrypted message is : Hello World Coal'. At the bottom, it states 'C:\Users\AA\source\repos\qwert\Debug\qwert.exe (process 13724) exited with code 0. Press any key to close this window . . .'.

SAMPLE 2:

```
Microsoft Visual Studio Debug Console

Enter any two distinct prime numbers (whose product is less than 128): 7
11

Enter the message want to encrypt : Fast Nuces

Calculated values are :
=====

The value of first prime number is: 7
The value of second prime number is: 11
The value of n : 77
The value of k : 60
The value of e : 59

-----ENCRYPTED CONTEXT-----

Encrypted Key values are : +93 +97 +169 +123 +25 +66 +117 +122 +127 +169

The encrypted data is : ] a r { ↓ B u z Δ r

-----DECRYPTED CONTEXT-----

The decrypted key values are : +70+97+115+116+32+78+117+99+101+115

The decrypted message is : Fast Nuces

Microsoft Visual Studio D...
```

SAMPLE 3:

```
Microsoft Visual Studio Debug Console

Enter the message want to encrypt : Coal Project

Calculated values are :
=====

The value of first prime number is: 7
The value of second prime number is: 11
The value of n : 77
The value of k : 60
The value of e : 59

-----ENCRYPTED CONTEXT-----

Encrypted Key values are : +88 +132 +97 +141 +25 +43 +126 +132 +150 +127 +122 +123 +25

The encrypted data is : X ä a î ↓ + ~ ä û Δ z { ↓

-----DECRYPTED CONTEXT-----

The decrypted key values are : -189-145-159-148-224-176-142-145-150-155-157-140-224

The decrypted message is : Coal Project

C:\Users\AA\source\repos\qwert\Debug\qwert.exe (process 15036) exited with code 0.
Press any key to close this window . . .
```

SAMPLE 4:

```
Microsoft Visual Studio Debug Console
Enter any two distinct prime numbers (whose product is less than 128): 7
13

Enter the message want to encrypt : Do you know me bruhh ?

Calculated values are :
=====

The value of first prime number is: 7
The value of second prime number is: 13
The value of n : 91
The value of k : 72
The value of e : 71

-----ENCRYPTED CONTEXT-----

Encrypted Key values are : +89 +181 +32 +147 +181 +166 +32 +154 +110 +181 +100 +32 +109 +169 +32 +142 +182 +166 +153 +15
3 +32 +16

The encrypted data is : Y ¼ ½ ð n ¼ d m - Ä ½ ð Ö Ö ►

-----DECRYPTED CONTEXT-----

The decrypted key values are : -188-145-224-135-145-139-224-149-146-145-137-224-147-155-224-158-142-139-152-152-224-193

The decrypted message is : Do you know me bruhh ?
```

SAMPLE 5:

```
Microsoft Visual Studio Debug Console

Calculated values are :
=====

The value of first prime number is: 7
The value of second prime number is: 13
The value of n : 91
The value of k : 72
The value of e : 71

-----ENCRYPTED CONTEXT-----

Encrypted Key values are : +10 +166 +157 +134 +169 +97 +182 +32 +8 +181 +109 +142 +32 +44 +181 +182 +109 +166 +134 +97 +
32 +142 +137 +97 +181 +32 +25 +25

The encrypted data is :
¾ Å - a ½ ¼ m Ä , ¼ ½ m ð Å a Ä ë a ¼ ↓ ↓

-----DECRYPTED CONTEXT-----

The decrypted key values are : -178-139-157-148-155-159-142-224-190-145-147-158-224-186-145-142-147-139-148-159-224-158-
140-159-145-224-210-210

The decrypted message is : Nuclear Bomb Formula btao ..

C:\Users\AA\source\repos\qwert\Debug\qwert.exe (process 14404) exited with code 0.
Press any key to close this window . . .
```

SAMPLE 6:

```
Select Microsoft Visual Studio Debug Console
Enter any two distinct prime numbers (whose product is less than 128): 7
11
Enter the message want to encrypt : Do not ask about my GPA

Calculated values are :
=====
The value of first prime number is: 7
The value of second prime number is: 11
The value of n : 77
The value of k : 60
The value of e : 59

-----ENCRYPTED CONTEXT-----
Encrypted Key values are : +61 +132 +25 +166 +132 +123 +25 +97 +169 +140 +25 +97 +135 +132 +117 +123 +25 +102 +133 +25 +
59 +43 +91
The encrypted data is : = ä ↓ ã { ↓ a ¯ î ↓ a ç ä u { ↓ f à ↓ ; + [

-----DECRYPTED CONTEXT-----
The decrypted key values are : -188-145-224-146-145-140-224-159-141-149-224-159-158-145-139-140-224-147-135-224-185-176-
191
The decrypted message is : Do not ask about my GPA
```

Conclusion:

Our main goal in the statement was that we should secure the message from being its get stolen by means of ciphering and provide the key to only the person we want to give or provide the formula of decryption. So that the message should remain end to end encrypted and cipher everyone can see / read but unable to understand the meaning of it until properly decoded into plain text.