RSA Algorithm Analysis

Name	Code
Abdelaziz Salah Mohammed	173041

Delivered to:

DR_ Samir Shahin

Eng. Khaled Moataz

Date: 13/4/2023

Sender Explanation:

```
This is an RSA sender program. It takes a message and a public key
    and encrypts the message using the public key. The encrypted
message
    is then sent to the receiver.
    @Algorithm is as follows:
    1. we define a map, which converts the 36 available characters to
a number.
    as follows
        [0 -> 0, 1 -> 1 ... 9 -> 9, a -> 10, b -> 11, ... z -> 35]
    2. if we find any different character, we deal with it simply as
space.
    3. Read the message from the user.
    4. before implementing the RSA, we must code the input into
numbers.
    5. our scheme is as follows:
        - Convert any extra characters to spaces as specified above.
        - Group the plaintext into sets of five characters per group
        - If the last grouping does not have exactly five
characters,then append some space to the end of the plaintext message
to fill out the last grouping
        - Convert each group into a separate number as follows
            - group one is [c4,c3,c2,c1,c0]
            - then the number = c4*37^4 + c3*37^3 + c2*37^2 + c1*37^1
+ c0*37^0
        - A decoding operation should be performed in the same way
except that the process is reversed.
          A number is converted to its character grouping by using mod
and div
```

Receiver Explanation:

```
 '''
      This file act as a receiver server
      It Accept the message coming from the sender
      Then it decrypt it using its private key
      After that it decode the message
This is done by
  1. Loop for i from 0 to 5
  2. Get the number mod 37
  3. Then apply integer division by 37
  4. Then append the character to the list in which you cascade
     letters in.

 Then we show the message.
    '''
```

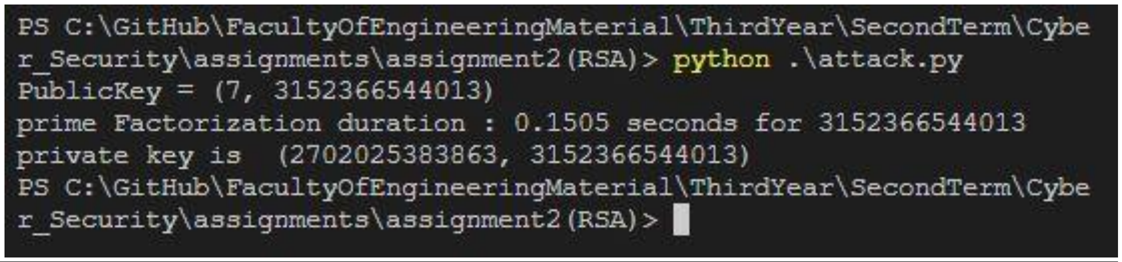
Attack Explanation:

```
 here we want to apply the attack
        so the algorithm will be as follows:
            1- read the public key of the server
            2- apply prime factorization on n
            3- calculate phi(n)
            4- calculate d
            5- decrypt the message
```

Attack notes:

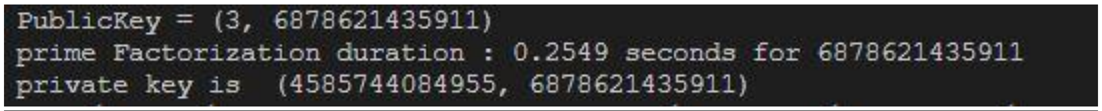
- As the number of bits used in generating the pair of keys increases, the time taken to be able to apply the attack will increase.
- The increase in the time increases exponentially.
- I have computed the time taken for apply the attack for some values of n, and here are the results:

N = 21:



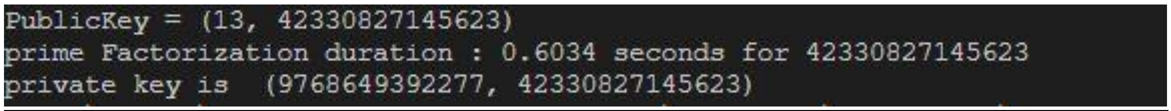
```
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTerm\Cybe
r_Security\assignments\assignment2(RSA)> python .\attack.py
PublicKey = (7, 3152366544013)
prime Factorization duration : 0.1505 seconds for 3152366544013
private key is  (2702025383863, 3152366544013)
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTerm\Cybe
r_Security\assignments\assignment2(RSA)>
```

N = 22:



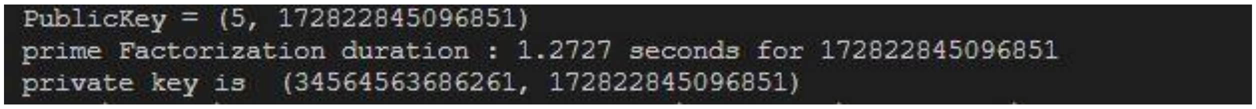
```
PublicKey = (3, 6878621435911)
prime Factorization duration : 0.2549 seconds for 6878621435911
private key is  (4585744084955, 6878621435911)
```

N = 23:



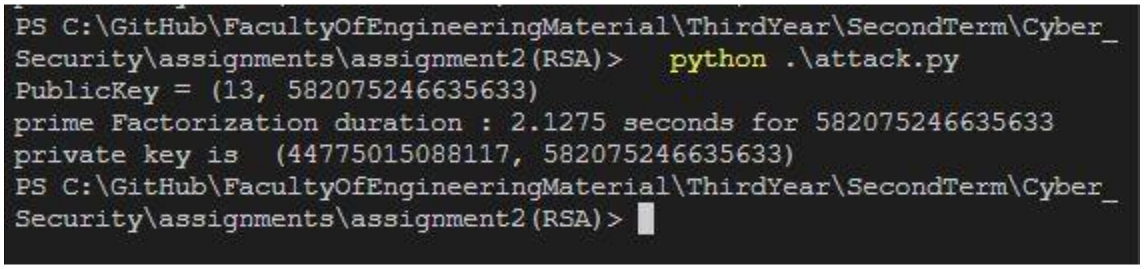
```
PublicKey = (13, 42330827145623)
prime Factorization duration : 0.6034 seconds for 42330827145623
private key is  (9768649392277, 42330827145623)
```

N = 24:



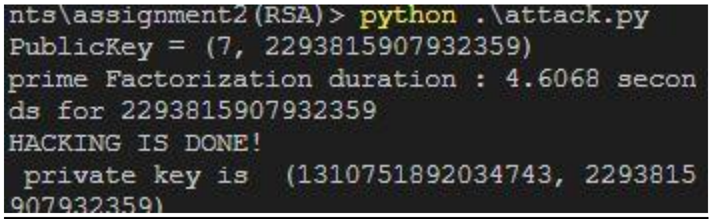
```
PublicKey = (5, 172822845096851)
prime Factorization duration : 1.2727 seconds for 172822845096851
private key is  (34564563686261, 172822845096851)
```

N = 25:



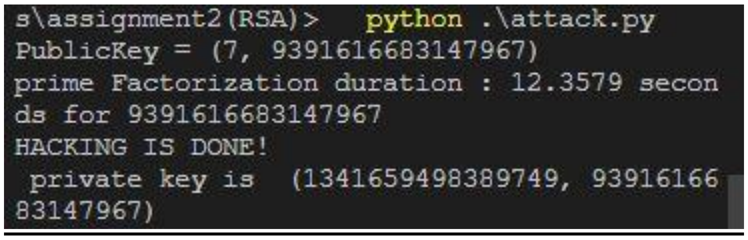
```
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTerm\Cyber_
Security\assignments\assignment2(RSA)>   python .\attack.py
PublicKey = (13, 582075246635633)
prime Factorization duration : 2.1275 seconds for 582075246635633
private key is  (44775015088117, 582075246635633)
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTerm\Cyber_
Security\assignments\assignment2(RSA)>
```

N = 26:



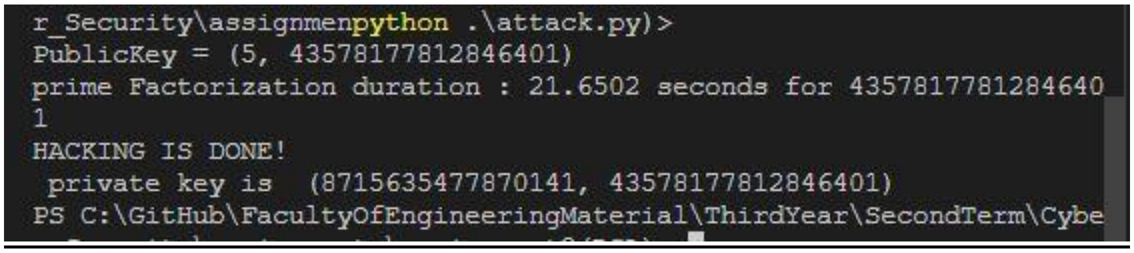
```
nts\assignment2(RSA)> python .\attack.py
PublicKey = (7, 2293815907932359)
prime Factorization duration : 4.6068 secon
ds for 2293815907932359
HACKING IS DONE!
 private key is  (1310751892034743, 2293815
907932359)
```

N = 27:



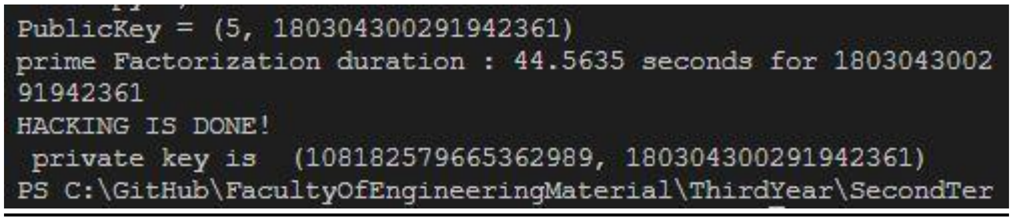
```
s\assignment2(RSA)>   python .\attack.py
PublicKey = (7, 9391616683147967)
prime Factorization duration : 12.3579 secon
ds for 9391616683147967
HACKING IS DONE!
 private key is  (1341659498389749, 93916166
83147967)
```

N = 28:



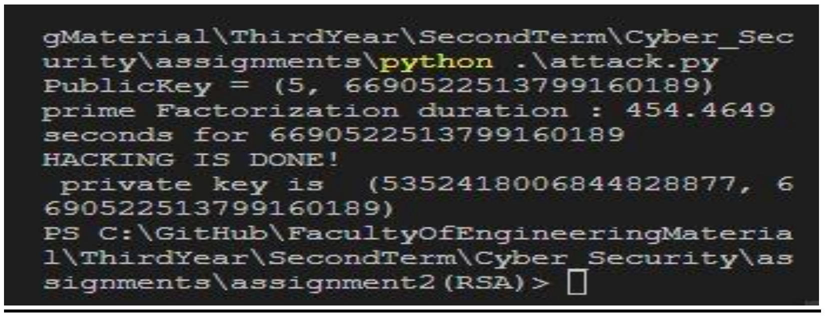
```
r_Security\assignmenpython .\attack.py)>
PublicKey = (5, 43578177812846401)
prime Factorization duration : 21.6502 seconds for 4357817781284640
1
HACKING IS DONE!
 private key is  (8715635477870141, 43578177812846401)
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTerm\Cybe
```

N = 29:



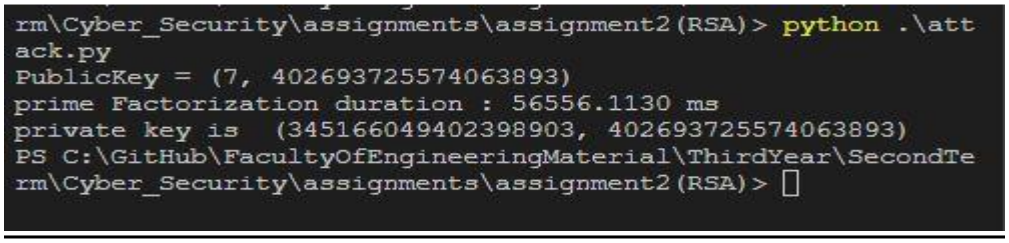
```
PublicKey = (5, 180304300291942361)
prime Factorization duration : 44.5635 seconds for 1803043002
91942361
HACKING IS DONE!
 private key is  (108182579665362989, 180304300291942361)
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTer
```

N = 30:



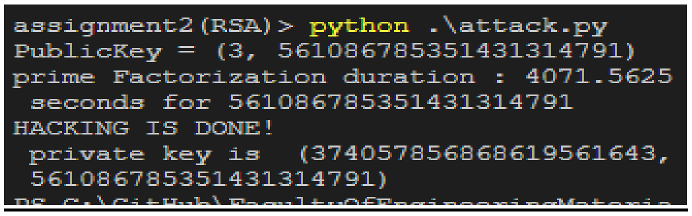
```
gMaterial\ThirdYear\SecondTerm\Cyber_Sec
urity\assignments\python .\attack.py
PublicKey = (5, 669052251379916O189)
prime Factorization duration : 454.4649
seconds for 669052251379916O189
HACKING IS DONE!
 private key is  (535241800684482887 7, 6
69052251379916O189)
PS C:\GitHub\FacultyOfEngineeringMateria
l\ThirdYear\SecondTerm\Cyber_Security\as
signments\assignment2(RSA)> 
```

N = 32:



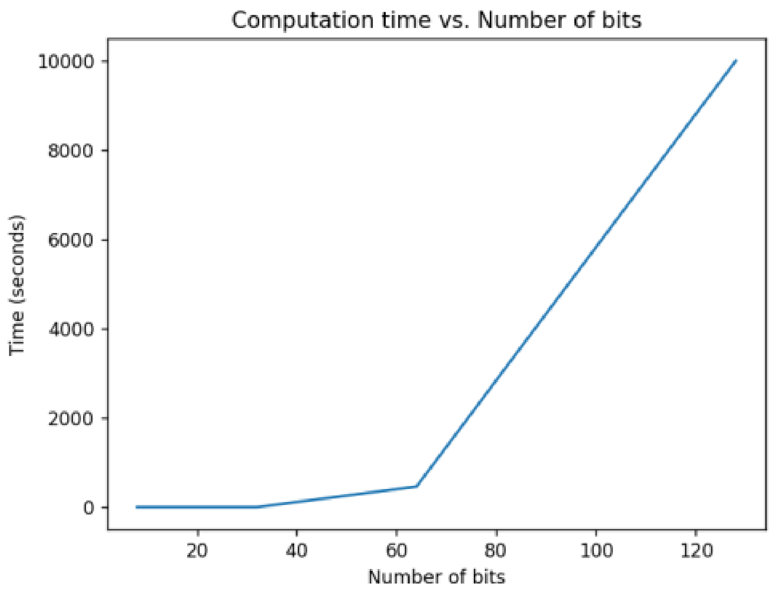
```
rm\Cyber_Security\assignments\assignment2(RSA)> python .\att
ack.py
PublicKey = (7, 402693725574063893)
prime Factorization duration : 56556.1130 ms
private key is  (345166049402398903, 402693725574063893)
PS C:\GitHub\FacultyOfEngineeringMaterial\ThirdYear\SecondTe
rm\Cyber_Security\assignments\assignment2(RSA)> 
```

N = 35:



```
assignment2(RSA)> python .\attack.py
PublicKey = (3, 561086785351431314791)
prime Factorization duration : 4071.5625
 seconds for 561086785351431314791
HACKING IS DONE!
 private key is  (374057856868619561643,
 561086785351431314791)
PS C:\GitHub\FacultyOfEngineeringMateria
```

Time vs number of bits:


Computation time vs. Number of bits
Time (seconds)
Number of bits