BEING EVE

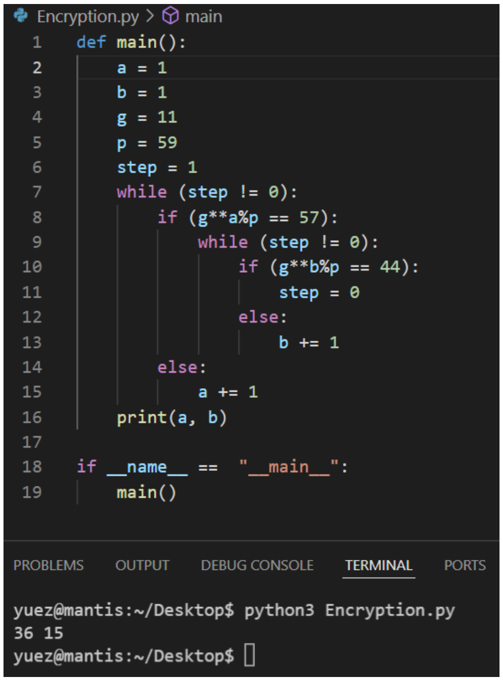
Zitian Yue

April 12, 2022

1 Diffie-Hellman key exchange

I claim that the secret shared by Alice and Bob is $K = 36$.

To figure out the shared secret, I define $A = 57$ as the message from Alice to Bob and $B = 44$ as the message from Bob to Alice. The communication between Alice and Bob at this stage is the process of agreeing on one single key for further encryption. I know that Alice and Bob will both have secret keys a and b to encrypt their messages A and B. The way of encryption are defined as $A = g^a \ mod \ p$ and $B = g^b \ mod \ p$. Therefore, I need to determine value of a and b first in order to figure out the common key Alice and Bob will be using. To do this, I wrote a python program which uses loop functions to find the first values of a and b that satisfy definitions of A and B. My evil program told me that $a = 36$ and $b = 15$. I then confirmed the accuracy of results by showing that $11^3 6 \ mod \ 59 = 57$ and $11^1 5 \ mod \ 59 = 44$. The output and codes of my program are displayed below.



```python
def main():
    a = 1
    b = 1
    g = 11
    p = 59
    step = 1
    while (step != 0):
        if (g**a%p == 57):
            while (step != 0):
                if (g**b%p == 44):
                    step = 0
                else:
                    b += 1
        else:
            a += 1
    print(a, b)

if __name__ == "__main__":
    main()
```

```
yuez@mantis:~/Desktop$ python3 Encryption.py
36 15
yuez@mantis:~/Desktop$ 
```

After derivation of a and b, I can now figure our the shared secret shared upon Alice and Bob by using $K = g^{ab} \ mod \ p = 11^{36\times 15} \ mod \ 59 = 36.K$ represents the key that Alice and Bob will use to encrypt their messages during communications in the future. I can now know anything between Alice and Bob

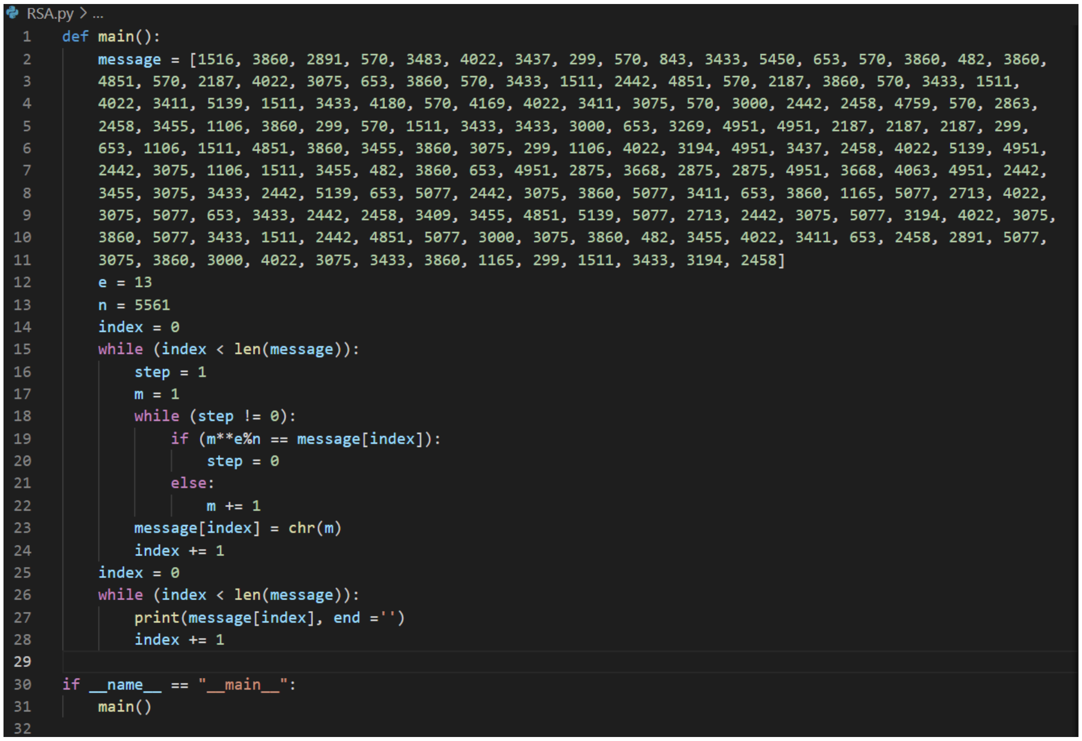
If the integers involved were much larger, my program would fail due to a tremendously large amount of operations that have to be done. For example, my computer couldn't handle the calculation when I define $g = 46516163$ and $p = 589657522$. Other than that, I may also fail to derive the correct answer since there will be more combinations of integers that satisfy definitions of A and B. It will be hard to determine which combination should be used to derive K agreed by Alice and Bob.

2 RSA

The encrypted message sent from Alice to Bob is:

Hey Bob. It's even worse than we thought! Your pal, Alice. https://www.schneier.com/blog/archives/2022/04/airtags-are-used-for-stalking-far-more-than-previously-reported.html

I know that every integer in this list may represent an encrypted character by employing the RSA method. Since only Bob's public key is the know parameter, Alice will have to use this key to encrypt her message. Define every element in the list as E, I conclude that the process of encryption works like $M^{e_{Bob}}\ mod\ n_{Bob} = E$ where M indicates original information. Then I wrote a program in python in order to reverse the encryption. The codes of my program are displayed below.

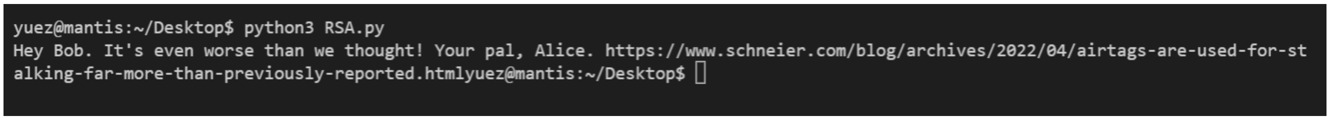


```python
def main():
    message = [1516, 3860, 2891, 570, 3483, 4022, 3437, 299, 570, 843, 3433, 5450, 653, 570, 3860, 482, 3860,
    4851, 570, 2187, 4022, 3075, 653, 3860, 570, 3433, 1511, 2442, 4851, 570, 2187, 3860, 570, 3433, 1511,
    4022, 3411, 5139, 1511, 3433, 4180, 570, 4169, 4022, 3411, 3075, 570, 3000, 2442, 2458, 4759, 570, 2863,
    2458, 3455, 1106, 3860, 299, 570, 1511, 3433, 3433, 3000, 653, 3269, 4951, 4951, 2187, 2187, 2187, 299,
    653, 1106, 1511, 4851, 3860, 3455, 3860, 3075, 299, 1106, 4022, 3194, 4951, 3437, 2458, 4022, 5139, 4951,
    2442, 3075, 1106, 1511, 3455, 482, 3860, 653, 4951, 2875, 3668, 2875, 2875, 4951, 3668, 4063, 4951, 2442,
    3455, 3075, 3433, 2442, 5139, 653, 5077, 2442, 3075, 3860, 5077, 3411, 653, 3860, 1165, 5077, 2713, 4022,
    3075, 5077, 653, 3433, 2442, 2458, 3409, 3455, 4851, 5139, 5077, 2713, 2442, 3075, 5077, 3194, 4022, 3075,
    3860, 5077, 3433, 1511, 2442, 4851, 5077, 3000, 3075, 3860, 482, 3455, 4022, 3411, 653, 2458, 2891, 5077,
    3075, 3860, 3000, 4022, 3075, 3433, 3860, 1165, 299, 1511, 3433, 3194, 2458]
    e = 13
    n = 5561
    index = 0
    while (index < len(message)):
        step = 1
        m = 1
        while (step != 0):
            if (m**e%n == message[index]):
                step = 0
            else:
                m += 1
        message[index] = chr(m)
        index += 1
    index = 0
    while (index < len(message)):
        print(message[index], end ='')
        index += 1

if __name__ == "__main__":
    main()

```

This program loops through the list of information. For each element, it uses another loop function to find the value of M that satisfies $M^{e_{Bob}}\ mod\ n_{Bob} = E$. After M is determined, I converted the integer value to a char value based on ASCII table through chr() function and inserted that char value back into a list. Finally, I printed out the actual message sent from Alice to Bob by looping through the list of the plaintext. The output of my program is displayed as below.



```
yuez@mantis:~/Desktop$ python3 RSA.py
Hey Bob. It's even worse than we thought! Your pal, Alice. https://www.schneier.com/blog/archives/2022/04/airtags-are-used-for-st
alking-far-more-than-previously-reported.htmlyuez@mantis:~/Desktop$ 
```

If the integers involved were much larger, my program mail fail due to the large amount of calculations entailed in the process of decryption. However, the message of Alice may still be insecure even if Bob's keys involved larger integers. Notice that there are many repetitive integers in the list of encrypted message since they represent the same character. The Eve can utilize this characteristic and classify indications of integers to slowly figure out what the original message is. Or the Eve can just make a plan of acquiring Bob's secret key. In that way, any message sent from Alice to Bob will become transparent.