

# Homework 4

## AES Encryption

### Generating large primes (p and q):

The Rabin–Miller is primality test: this algorithm determines whether any given number is prime or not.

Mode a number p with all the numbers starting from 2 up to the sqrt of the number is how our is\_prime() function make sure that requested number is prime or not. A number will be prime when it never leaves a remainder 0 when this number is mode by a other number other than its factors.

Any given number n we need to check if this number is prime or not. the Rabin-Miller primality algorithm works as follow:

Input Number 1:  $n > 3$ , an odd number checked for primality using this algorithm

Input Number 2: k, the counter, testing rounds

Output: “composite number” if number n is composite otherwise “probably prime”.

### RabinMiller Algo:

write down 'n' as  $\text{pow}(2, r \cdot d + 1)$  with d being odd

KtimeLoop: repeat the process k times:

KtimeLoop: for all numbers a in the range from  $[2, \min(n-2, \lfloor 2(\ln n)^2 \rfloor)]$ :

$x\_num \leftarrow \text{pow}(a, d) \bmod n$

if  $x\_num = 1$  or  $x\_num = n - 1$  then

continue KtimeLoop

repeat\_this r - 1 times:

$x\_num \leftarrow \text{pow}(x, 2) \bmod n$

if  $x\_num = n - 1$  then

continue KtimeLoop

return “composite\_number”

return “prime\_number”

def generate\_private\_public\_key

while True:

$e = \text{random.randrange}(2^{**}(\text{default\_key\_size} - 1), 2^{**}(\text{default\_key\_size}))$

if  $\text{gcd}(e, (p - 1) * (q - 1)) == 1$ :

break

$\text{print}(\backslash n \text{ ***** calculating d for private key that is mod inverse of e... ..})$

$d = \text{find\_modular\_inverse}(e, (p - 1) * (q - 1))$

$\text{rsa\_public\_key} = (n, e)$

$\text{rsa\_private\_key} = (n, d)$

return (rsa\_public\_key, rsa\_private\_key)

The `generate_large_prime_of_keySize()` function returns an int which is prime. It comes up with large random number and store then in number variable and then check primality with `is_Prime()` function. The function `is_prime()` internally uses a modified version of Rabin miller algorithm

when the current number `p` is prime, then line we return the current num. Otherwise we run infinite loop with random number and this process runs all over again.

Once we generate `N,e` and `N,d` we store `(N, e)` as `user.pub` key and `(N,d)` as `user.prv` key which we will be using in `crypt.py` for securing AES key.

### **Crypt.py**

Take input from user and based on `-e` or `-d`, we decide whether we need to encrypt or decrypt.

Encrypt:

We first get random 128 bits from `random.getrandbits()` function which we'll be using as key for our AES encryption and decryption.

Then use MD5 to hash our key using `hashlib` library as follows:

We create an AES instance with ECB mode as follow:

```
cipher = AES.new(self.key, AES.MODE_ECB)
encrypted = cipher.encrypt(raw_message)
encoded = base64.b64encode(encrypted)
```

Since AES encrypted message is in bytes so we need to used base64 encoder with any encoding format.

Once we receive the encrypted cipher from AES, we store cipher and encrypted AES key with a special separator.

While decrypting, we read the `message.cip` from file and split the content which result in first part as our encrypted message and second part as encrypted AES key.

We first need to decrypt the AES key from RSA decryptor. For this purpose, we read RSA key file of the mentioned user and get the `(d, N)`

Decrypt the AES key. Here if we have more than 128 bits key, we divide the key 128 bits blocks and then do encryption and decryption. Since we chose 128 bits, we'll have only one block is AES key to encrypt and decrypt.

```
cipher = AES.new(key, AES.MODE_ECB)
decrypted = cipher.decrypt(base64.b64decode(enc))
```

After we read `user.prv` and decrypt the AES key be performing RSA decryption, we pass same decrypted key to AES decryption instance to decrypt our `message.cip` file.

Uses:

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ python genkeys.py bob
***** generating p prime.....
***** generating q prime.....
Storing..... bob.pub
Storing..... bob.prv
```

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ python genkeys.py alice
***** generating p prime.....
***** generating q prime.....
Storing..... alice.pub
Storing..... alice.prv
```

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ cat message.txt
This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is
Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh
This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is
Mukesh
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ python crypt.py -e bob.pub
message.txt message.cip
Encryption of message.txt file is done. cipher text is present in message.cip file
Process is done
```

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ python crypt.py -d bob.prv
message.cip message.txt
plaintext This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh
This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is
Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh
This is Mukesh
Decryption of message.cip file is done. plain text is in message.txt file
Process is done
```

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ cat message.txt
This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is
Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh
This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is Mukesh This is
Mukesh
```

### Using Alice Key with message2.txt

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ cat message2.txt
I have strong grasp of technology as well as business skills. Since in my previous role i had interacted
with business leads and the country manager of our Asia pacific business as well as their technology
teams.
```

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ python crypt.py -e alice.pub
message2.txt message2.cip
Encryption of message.txt file is done. cipher text is present in message.cip file
Process is done
```

```
mukesh@localhost in ~/Downloads/CSCI531_HW_3 on master [!?$]$ python crypt.py -d alice.prv
message2.cip message2.txt
```

Decryption of message2.cip file is done. plain text is in message.txt file  
Process is done

I have strong grasp of technology as well as business skills. Since in my previous role i had interacted with business leads and the country manager of our Asia pacific business as well as their technology teams.

[illegible]

[illegible]