資訊安全導論 HW4

B10815057 廖聖郝

產生 key

首先生成兩個在 bit 範圍內的隨機質數 p q

用 p q 算出 n 與 phi_n

公鑰 e 是隨機的從範圍(2~phi_n)內挑出一個與 phi_n 互質的數字

挑法: 1.生成隨機數 2.若為偶數 +1 變奇數 3.判斷是否互質 不互質找下一個奇數

私鑰 d 是公鑰 e 在模 phi_n 下的模反元素，因此解此式 (e*d) mod phi_n == 1，我使用 python 的 pow 函式內建的模反計算(第二個參數設為-1)

```python
#生成所有 key
# p q 為大質數
# n 為 p*q phi_n 為(p-1)*(q-1)
# e 為公鑰 d 為私鑰
def key_gen(bit_count):
    p = prime_gen(int(bit_count))
    q = prime_gen(int(bit_count))
    n = p * q
    phi_n = (p-1)*(q-1)
    public_key = random.randint(2,phi_n)
    if public_key % 2 == 0:
        public_key += 1
    while not is_coprime(public_key,phi_n):
        public_key += 2
    private_key = pow(public_key,-1,phi_n)
    print("p = {}".format(p))
    print("q = {}".format(q))
    print("n = {}".format(n))
    print("phi = {}".format(phi_n))
    print("e = {}".format(public_key))
    print("d = {}".format(private_key))
    return
```

質數生成: 1.生成隨機數 2.若為偶數 +1 變奇數 3.判斷是否為質數 不是的話往下一個奇數找

```python
#生成一個指定 bit 數範圍內的質數
def prime_gen(bit_count):
    result = random.getrandbits(bit_count)
    if result % 2 == 0:
        result += 1
    while miller_rabin(result) == False:
        result += 2
    return result
```

判斷質數: miller robin 演算法 [米勒-拉賓質數判定法 - 維基百科，自由的百科全書 (wikipedia.org)]

```python
#miller rabin 演算法
#用於判斷一個數字是否為質數
def miller_rabin(check_num):
    if check_num == 2 or check_num == 3:
        return True
    if check_num == 1 or check_num < 0 or check_num % 2 == 0:
        return False
    k = 10
    r = 0
    s = check_num-1
    while(s%2==0):
        r += 1
        s //=2
    for i in range(k):
        a = random.randrange(2,check_num-1)
        x = pow(a,s,check_num)
        if(x==1 or x == check_num-1):
            continue
        for j in range(r-1):
            x = pow(x,2,check_num)
            if(x==check_num-1):
                break
        else:
            return False
    return True
```

判斷互質: 1.判斷輸入 ab 大小 大的放第一個參數 2.求 ab 最大公因數 若為 1

則互質

```python
#求 a b 最大公因數(無遞迴版本)，大的數字要放在 a
def gcd(a,b):
    if b <= 1 or b >= a:
        return -1

    while b != 0:
        a, b = b, a % b
    return a
#確認 a b 是否互質，也就是確認 a b 的最大公因數是不是 1
#會先判斷 a b 大小，將大的放在第一個參數，以配合 gcd
def is_coprime(a, b):
    if a >= b:
        return gcd(a, b) == 1
    else:
        return gcd(b, a) == 1
```

加密

輸入一訊息，與 n 和公鑰，回傳加密後數字的 base64 編碼

```
#加密函式，輸入待加密訊息(str)與 n 和公鑰
#並將 RSA 加密後的數字透過 base64 編碼，使其能夠顯示
def encrypt(plain_text,n,public_key):
    plain_num = str2num(plain_text)
    cipher_num = RSA_encrypt(plain_num,n,public_key)
    cipher_base64 = base64.b64encode(str(cipher_num).encode('ascii'))
    cipher_text = cipher_base64.decode('ascii')
    return cipher_text
```

RSA 加密，輸入一待加密數字與 n 和公鑰，回傳加密後數字

求: (plain_num ^ 公鑰) mod n

```
#RSA 加密，輸入一數字與 n 和公鑰
#算出加密後的數字
def RSA_encrypt(plain_num,n,public_key):
    cipher_num = fast_power_mod(plain_num,public_key,n)
    return cipher_num
```

fast_power_mod 即為 square and multiply 演算法

將次方數轉為二進制，掃過所有 bit，遇 0 則 square，遇 1 則 square 與

multiply

```
#計算(base^exponent) mod mod_num
#採用 sqare and multiply algorithm 加速
def fast_power_mod(base, exponent, mod_num):
    bins = bin(exponent)
    result = 1
    for index in range(0,len(bins)):
        result = (result * result) % mod_num
        if bins[index] == '1':
            result = (result * base) % mod_num
    return result
```

解密

輸入加密後的 base64 編碼，與 n 和私鑰，回傳解密後原始訊息

```python
#解密函式，輸入加密後訊息(base64 的 str)與 n 和私鑰
#解密出數字後，將數字轉回原始訊息
def decrypt(cipher_text,n,private_key):
    cipher_num = int(base64.b64decode(cipher_text).decode('ascii'))
    plain_num = RSA_decrypt(cipher_num,n,private_key)
    plain_text = num2str(plain_num)
    return plain_text
```

RSA 加密，輸入一加密後數字與 n 和私鑰，回傳原始數字

求: (cipher_num ^ 私鑰) mod n

```python
#RSA 解密，輸入一加密數字與 n 和私鑰
#算出解密後的數字
def RSA_decrypt(cipher_num,n,private_key):
    plain_num = fast_power_mod(cipher_num,private_key,n)
    return plain_num
```

CRT 解密

輸入加密後的 base64 編碼，與 p 與 q 和私鑰，回傳解密後原始訊息

```
#CRT 加速之解密函式，輸入加密後訊息(base64 的 str)與 p 與 q 和私鑰
#解密出數字後，將數字轉回原始訊息
def CRT_decrypt(cipher_text,p,q,private_key):
    cipher_num = int(base64.b64decode(cipher_text).decode('ascii'))
    plain_num = RSA_CRT_decrypt(cipher_num,p,q,private_key)
    plain_text = num2str(plain_num)
    return plain_text
```

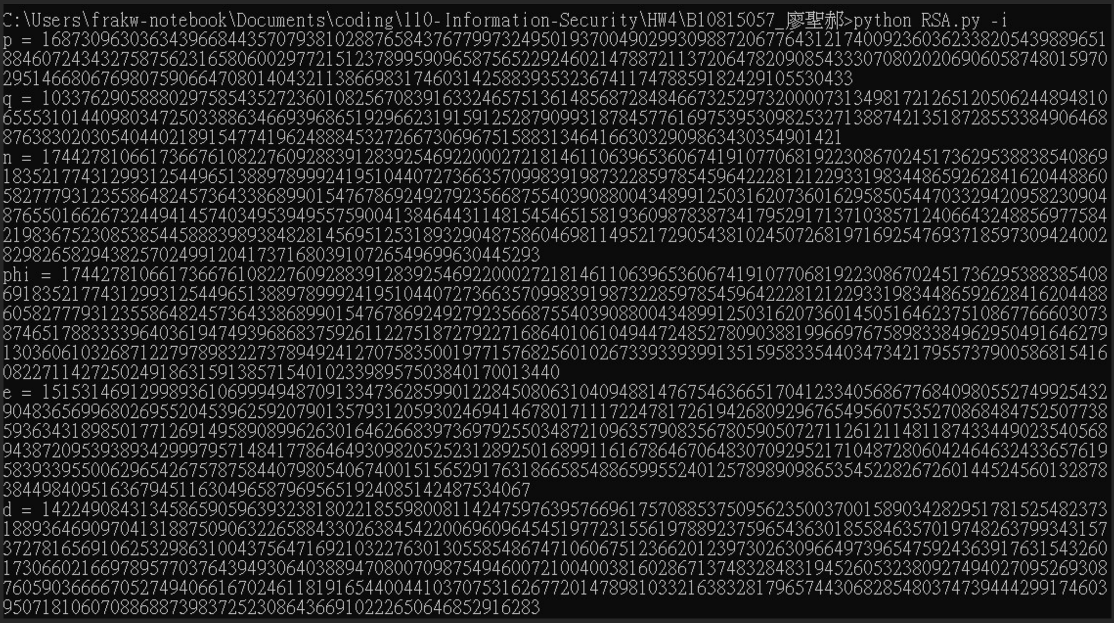
RSA 加密，輸入一加密後數字與 p 與 q 和私鑰，回傳原始數字

採用中國剩餘定理加速

```
#RSA  CRT 解密，輸入一數字與 p 與 q 和私鑰
#算出解密後的數字
def RSA_CRT_decrypt(cipher_num,p,q,private_key):
    dp = private_key % (p-1)
    dq = private_key % (q-1)
    q_inv = pow(q,-1,p)
    m1 = fast_power_mod(cipher_num,dp,p)
    m2 = fast_power_mod(cipher_num,dq,q)
    h = q_inv * (m1-m2) % p
    result = m2 + h * q
    return int(result)
```

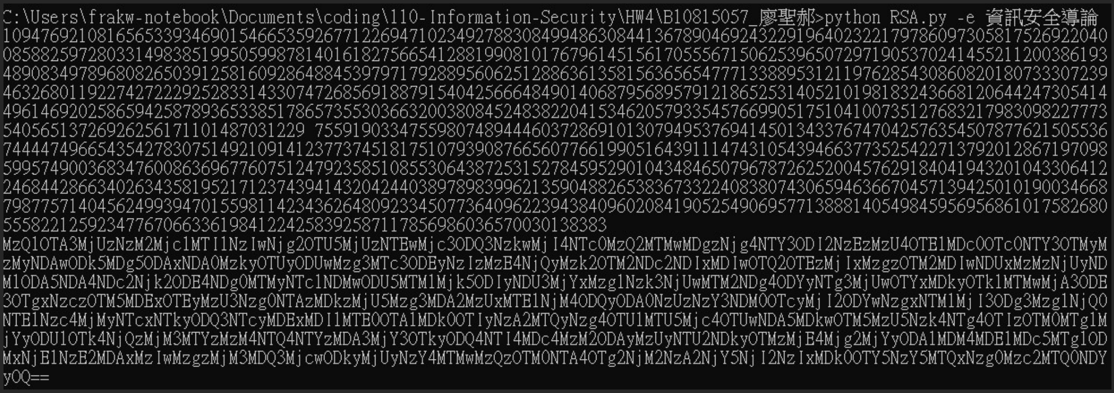
結果

Key gen:



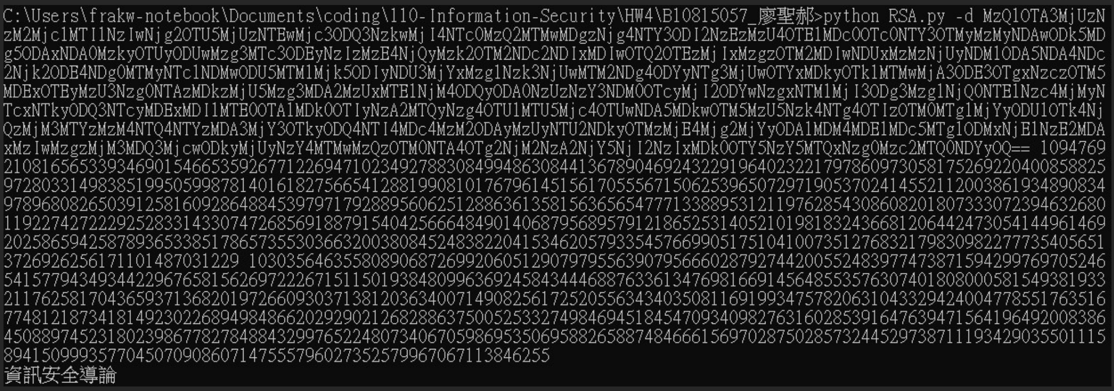
```
C:\Users\frakw-notebook\Documents\coding\110-Information-Security\HW4\B10815057_廖聖郝>python RSA.py -i
p = 1687309630363439668443570793810288765843767799732495019370049029930988720677643121740092360362338205439889651
88460724343275875623165806002977215123789959096587565229246021478872113720647820908543330708020206906058748015970
295146680676980759066470801404321138669831746031425883935323674117478859182429105530433
q = 1033762905888029758543527236010825670839163324657513614856872848466732529732000073134981721265120506244894810
65553101440980347250338863466939686519296623191591252879099318784577616975395309825327138874213518728553384906468
87638302030540440218915477419624888453272667306967515883134641663032909863430354901421
n = 1744278106617366761082276092883912839254692200027218146110639653606741910770681922308670245173629538838540869
183521774312993125449651388978999924195104407273663570998391987322859785459642228121229331983448659262841620448860
58277793123558648245736433868990154767869249279235668755403908800434899125031620736016295850544703329420958230904
87655016626732449414574034953949557590041384644311481545465158193609878387341795291713710385712406643248856977584
21983675230853854458883989384828145695125318932904875860469811495217290543810245072681971692547693718597309424002
8298265829438257024991204173716803910726549699630445293
phi = 17442781066173667610822760928839128392546922000272181461106396536067419107706819223086702451736295388385408
691835217743129931254496513889789999241951044072736635709983919873228597854596422281212293319834486592628416204488
60582777931235586482457364338689901547678692492792356687554039088004348991250316207360145051646237510867766603073
87465178833339640361947493968683759261122751872792271686401061049447248527809038819966976758983384962950491646279
13036061032687122797898322737894924127075835001977157682560102673393393991351595833544034734217955737900586815416
082271142725024918631591385715401023398957503840170013440
e = 1515314691299893610699949487091334736285990122845080631040948814767546366517041233405686776840980552749925432
90483656996802695520453962592079013579312059302469414678017111722478172619426809296765495607535270868484752507738
59363431898501771269149589089962630164626683973697925503487210963579083567805905072711261211481187433449023540568
94387209539389342999795714841778646493098205252312892501689911616786467064830709295217104872806042464632433657619
58393395500629654267578758440798054067400151565291763186658548865995524012578989098653545228267260144524560132878
384498409516367945116304965879695651924085142487534067
d = 1422490843134586590596393238180221855980081142475976395766961757088537509562350037001589034282951781525482373
18893646909704131887509063226588433026384542200696096454519772315561978892375965436301855846357019748263799343157
37278165691062532986310043756471692103227630130558548674710606751236620123973026309664973965475924363917631543260
17306602166978957703764394930640388947080070987549460072100400381602867137483284831945260532380927494027095269308
76059036666705274940661670246118191654400441037075316267720147898103321638328179657443068285480374739444299174603
9507181060708868873983725230864366910222650646852916283
```

加密:



```
C:\Users\frakw-notebook\Documents\coding\110-Information-Security\HW4\B10815057_廖聖郝>python RSA.py -e 資訊安全導論
109476921081656533934690154665359267712269471023492788308499486308441367890469243229196402322179786097305817526922040
085882597280331498385199505998781401618275665412881990810176796145156170555671506253965072971905370241455211200386193
489083497896808265039125816092864884539797179288956062512886361358156365654777133889531211976285430860820180733307239
463268011922742722292528331433074726856918879154042566648490140687956895791218652531405210198183243668120644247305414
496146920258659425878936533851786573553036632003808452483822041534620579335457669905175104100735127683217983098227773
54056513726926256171101487031229 755919033475598074894446037286910130794953769414501343376747042576354507877621505536
744447496654354278307514921091412377374518175107939087665607766199051643911147431054394663773525422713792012867197098
599574900368347600863696776075124792358510855306438725315278459529010434846507967872625200457629184041943201043306412
246844286634026343581952171237439414320424403897898399621359048826538367332240838074306594636670457139425010190034668
798775714045624993947015598114234362648092334507736409622394384096020841905254906957713888140549845956956861017582680
5558221259234776706633619841224258392587117856986036570030138383
MzQ1OTA3MjUzNzM2Mjc1MTI1NzIwNjg2OTU5MjUzNTEwMjc3ODQ3NzkwMjI4NTc0MzQ2MTMwMDgzNjg4NTY3ODI2NzEzMzU4OTE1MDc0OTc0NTY3OTMyM
zMyNDAwODk5MDg5ODAxNDA0MzkyOTUyODUwMzg3MTc3ODEyNzIzMzE4NjQyMzk2OTM2NDc2NDIxMDIwOTQ2OTEzMjIxMzgzOTM2MDIwNDUxMzMzNjUyND
M1ODA5NDA4NDc2Njk2ODE4NDg0MTMyNTc1NDMwODU5MTM1Mjk5ODIyNDU3MjYxMzg1Nzk3NjUwMTM2NDg4ODYyNTg3MjUwOTYxMDkyOTk1MTMwMjA3ODE
3OTgxNzczOTM5MDExOTEyMzU3Nzg0NTAzMDkzMjU5Mzg3MDA2MzUxMTE1NjM4ODQyODA0NzUzNzY3NDM0OTcyMjI2ODYwNzgxNTM1MjI3ODg3Mzg1NjQ0
NTE1Nzc4MjMyNTcxNTkyODQ3NTcyMDExMDI1MTE0OTA1MDk0OTIyNzA2MTQyNzg4OTU1MTU5Mjc4OTUwNDA5MDkwOTM5MzU5Nzk4NTg4OTIzOTM0MTg1M
jYyODU1OTk4NjQzMjM3MTYzMzM4NTQ4NTYzMDA3MjY3OTkyODQ4NTI4MDc4MzM2ODAyMzUyNTU2NDkyOTMzMjE4Mjg2MjYyODA1MDM4MDE1MDc5MTg1OD
MxNjE1NzE2MDAxMzIwMzgzMjM3MDQ3MjcwODkyMjUyNzY4MTMwMzQzOTM0NTA4OTg2NjM2NzA2NjY5NjI2NzIxMDk0OTY5NzY5MTQxNzg0Mzc2MTQ0NDY
yOQ==
```

解密:



```
C:\Users\frakw-notebook\Documents\coding\110-Information-Security\HW4\B10815057_廖聖郝>python RSA.py -d MzQ1OTA3MjUzN
zM2Mjc1MTI1NzIwNjg2OTU5MjUzNTEwMjc3ODQ3NzkwMjI4NTc0MzQ2MTMwMDgzNjg4NTY3ODI2NzEzMzU4OTE1MDc0OTc0NTY3OTMyMzMyNDAwODk5MD
g5ODAxNDA0MzkyOTUyODUwMzg3MTc3ODEyNzIzMzE4NjQyMzk2OTM2NDc2NDIxMDIwOTQ2OTEzMjIxMzgzOTM2MDIwNDUxMzMzNjUyNDM1ODA5NDA4NDc
2Njk2ODE4NDg0MTMyNTc1NDMwODU5MTM1Mjk5ODIyNDU3MjYxMzg1Nzk3NjUwMTM2NDg4ODYyNTg3MjUwOTYxMDkyOTk1MTMwMjA3ODE3OTgxNzczOTM5
MDExOTEyMzU3Nzg0NTAzMDkzMjU5Mzg3MDA2MzUxMTE1NjM4ODQyODA0NzUzNzY3NDM0OTcyMjI2ODYwNzgxNTM1MjI3ODg3Mzg1NjQ0NTE1Nzc4MjMyN
TcxNTkyODQ3NTcyMDExMDI1MTE0OTA1MDk0OTIyNzA2MTQyNzg4OTU1MTU5Mjc4OTUwNDA5MDkwOTM5MzU5Nzk4NTg4OTIzOTM0MTg1MjYyODU1OTk4Nj
QzMjM3MTYzMzM4NTQ4NTYzMDA3MjY3OTkyODQ4NTI4MDc4MzM2ODAyMzUyNTU2NDkyOTMzMjE4Mjg2MjYyODA1MDM4MDE1MDc5MTg1ODMxNjE1NzE2MDA
xMzIwMzgzMjM3MDQ3MjcwODkyMjUyNzY4MTMwMzQzOTM0NTA4OTg2NjM2NzA2NjY5NjI2NzIxMDk0OTY5NzY5MTQxNzg0Mzc2MTQ0NDYyOQ== 1094769
210816565339346901546653592677122694710234927883084994863084413678904692432291964023221797860973058175269220400858825
972803314983851995059987814016182756654128819908101767961451561705556715062539650729719053702414552112003861934890834
978968082650391258160928648845397971792889560625128863613581563656547771338895312119762854308608201807333072394632680
119227427222925283314330747268569188791540425666484901406879568957912186525314052101981832436681206442473054144961469
202586594258789365338517865735530366320038084524838220415346205793354576699051751041007351276832179830982277735405651
3726926256171101487031229 1030356463558089068726992060512907979556390795666028792744200552483977473871594299769705246
541577943493442296765815626972226715115019384809963692458434446887633613476981669145648553576307401808000581549381933
211762581704365937136820197266093037138120363400714908256172520556343403508116919934757820631043329424004778551763516
774812187341814923022689498486620292902126828863750052533274984694518454709340982763160285391647639471564196492008386
450889745231802398677827848843299765224807340670598695350695882658874846661569702875028573244529738711193429035501115
8941509993577045070908607147555796027352579967067113846255
資訊安全導論
```