

CNS ASSIGNMENT REPORT

Blind Signature

Nandu Krishnan MG - 45

Nikhil Joji - 46

Nikhil Scaria - 47

Rasla KK - 48

S7 CSE

1 Digital Signature

A digital signature is a mathematical scheme for verifying the authenticity of digital messages or documents. A valid digital signature provides Authentication and integrity. A digital signature gives the receiver reason to believe the message was sent by the claimed sender.

2 Blind Signature

A Blind Signature is a form of digital signature in which the content of a message is disguised (blinded) before it is signed. The resulting blind signature can be publicly verified against the original. Blind signatures are typically employed in privacy-related protocols where the signer and message author are different parties.

Blind signature schemes can be implemented using a number of common public key signing schemes, for instance RSA and DSA. To perform such a signature, the message is first "blinded", typically by combining it in some way with a random "blinding factor". The blinded message is passed to a signer, who then signs it using a standard signing algorithm. The resulting message, along with the blinding factor, can be later verified against the signer's public key.

3 Blind RSA Signature

One of the simplest blind signature schemes is based on RSA signing. The blind version uses a random value r , such that r is relatively prime to N (i.e. $\gcd(r, N) = 1$) then,

$$m' = mr^e \pmod{N}$$

and sends the resulting value m' to the signing authority. Because r is a random value m' does not leak any information about m . The signing authority then calculates the blinded signature s' as:

$$s' = (m')^d \pmod{N}$$

s' is sent back to the author of the message, who can then remove the blinding factor to reveal s , the valid RSA signature of m :

$$s = s'r^{-1} \pmod{N}$$

This works because RSA keys satisfy the equation

$$r^{ed} = r(\text{mod} N)$$

and thus

$$s \equiv s' \cdot r^{-1} \equiv (m')^d r^{-1} \equiv m^d r^{ed} r^{-1} \equiv m^d r r^{-1} \equiv m^d \pmod{N}$$

hence s is indeed the signature of m.

4 Program

4.1 Signer (Server)

```
import socket
from decimal import Decimal

#Declaration
mysocket = socket.socket()
host = "127.0.0.1"
port = 4000

#Prevent socket.error: [Errno 98] Address already in use
mysocket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

mysocket.bind((host, port))

mysocket.listen(5)

c, addr = mysocket.accept()

def gcd(a,b):
    if b==0:
        return a
    else:
        return gcd(b,a%b)

# p = int(input('Enter the value of p = '))
p = 53
```

```

# q = int(input('Enter the value of q = '))
q = 59
# no = int(input('Enter the value of text = '))
n = p*q
t = (p-1)*(q-1)

#calculating e which is public key
for e in range(2,t):
    if gcd(e,t)== 1:
        break
print "Public Key (e,n): (" ,e, ', ',n, ')'"

#calculating d which is private key
for i in range(1,10):
    x = 1 + i*t
    if x % e == 0:
        d = int(x/e)
        break
print "Private Key (d,n): (" ,d, ', ',n, ')'"

#Wait for 'Okay'
data = c.recv(1024)
if (data == "Okay"):
    c.send("PublicKey"+str(e))

#wait until data is received
data = c.recv(1024)
data = data.replace("BlindedMsg", "")
print "Blinded message: ",data
blindedMsg = int(data)
# break

signedMsg = pow(blindedMsg,d,n)
print "Signed Msg: ",signedMsg
c.send("SignedMsg"+str(signedMsg))
c.close()

```

4.2 Encrypter (Client)

```
import socket
from decimal import Decimal

server = socket.socket()
host = "127.0.0.1"
port = 4000

server.connect((host, port))

def gcd(a,b):
    if b==0:
        return a
    else:
        return gcd(b,a%b)

# p = int(input('Enter the value of p = '))
p = 53
# q = int(input('Enter the value of q = '))
q = 59
no = int(input('Enter the value of text = '))
n = p*q
t = (p-1)*(q-1)

#generate r which is relatively prime to n to blind
for r in range(2,n):
    if gcd(r,n)== 1:
        break

print "Random Co-prime no. (r) is: ",r

server.sendall("Okay")

resp = server.recv(1024)
resp = resp.replace("PublicKey","")
e = int(resp) #publicKey

#blinding
ctt = pow(r,e,n)
blindedMsg = ctt*no
```

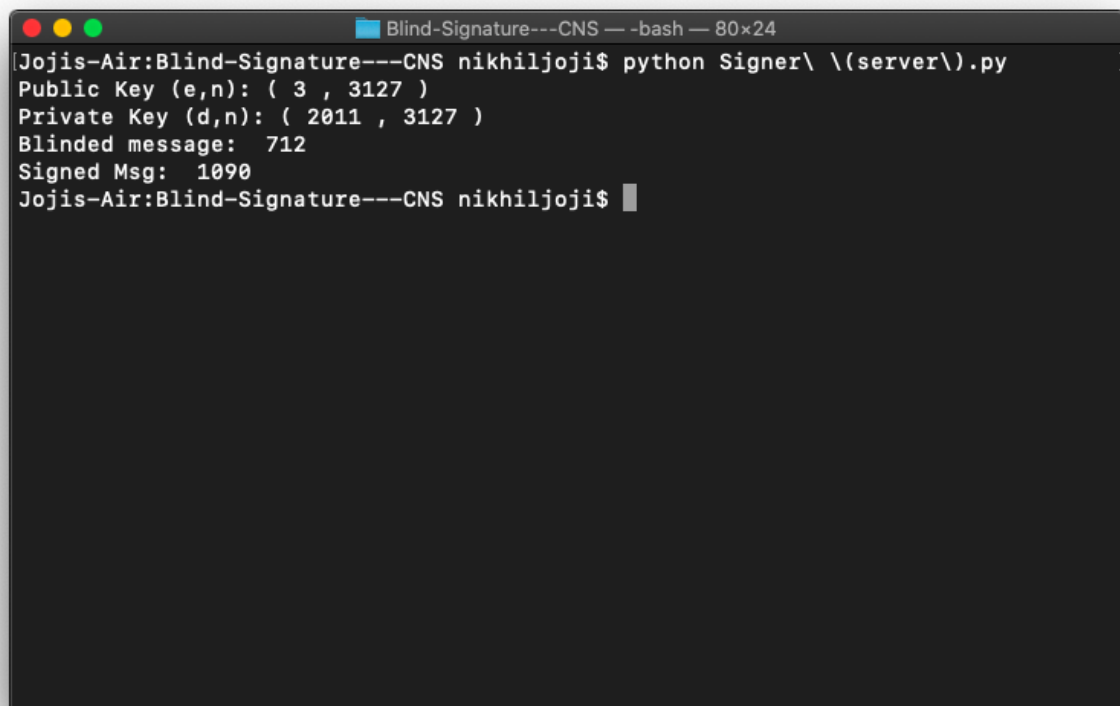
```
# blindedMsg now holds the blinded msg
print "Blinded Msg: ",blindedMsg
server.sendall("BlindedMsg"+str(blindedMsg))

server_string = server.recv(1024)
server_string = server_string.replace("SignedMsg","")
signedMsg = int(server_string)
print "Signed Msg: ",signedMsg

og = pow((signedMsg/r),e,n)
print "original msg: ",og
```

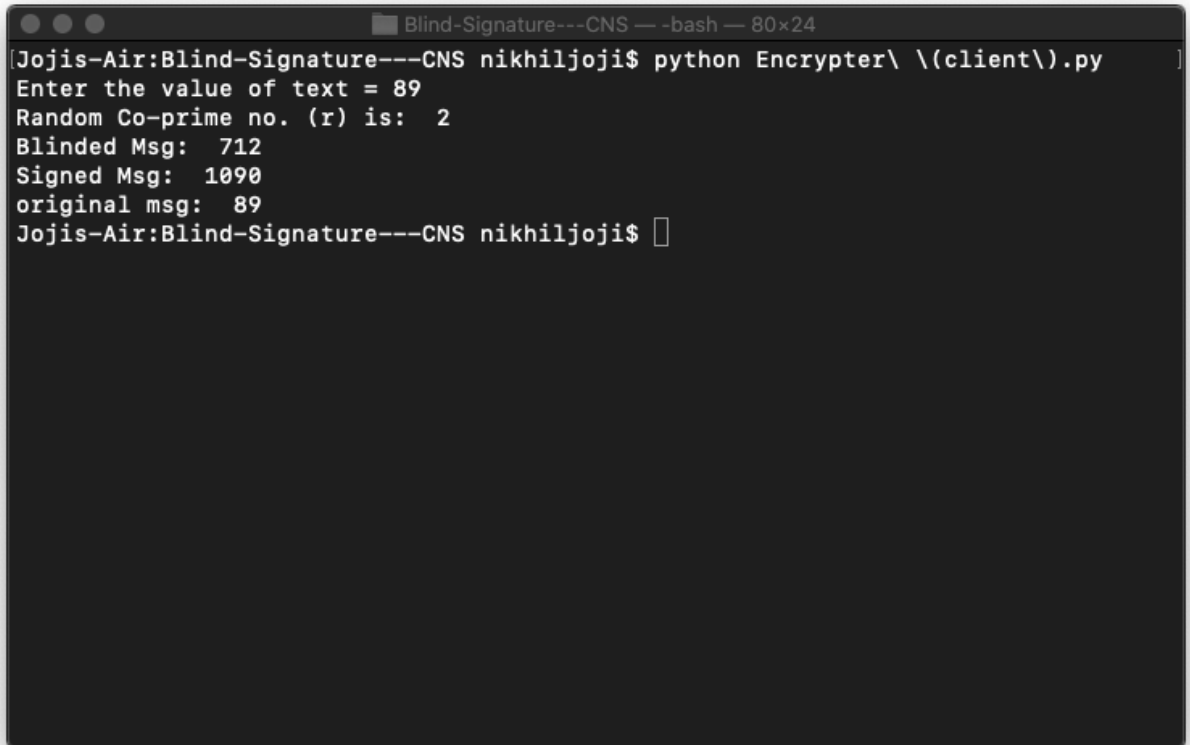
5 Output

5.1 Signer (Server)



```
Blind-Signature---CNS — -bash — 80x24
[Jojis-Air:Blind-Signature---CNS nikhiljoji$ python Signer\ \((server\).py
Public Key (e,n): ( 3 , 3127 )
Private Key (d,n): ( 2011 , 3127 )
Blinded message: 712
Signed Msg: 1090
Jojis-Air:Blind-Signature---CNS nikhiljoji$
```

5.2 Encrypter (Client)



```
Blind-Signature---CNS — -bash — 80x24
[Jojis-Air:Blind-Signature---CNS nikhiljoji$ python Encrypter\ \(client\).py
Enter the value of text = 89
Random Co-prime no. (r) is: 2
Blinded Msg: 712
Signed Msg: 1090
original msg: 89
Jojis-Air:Blind-Signature---CNS nikhiljoji$
```