ITC8240 Cryptography. Test2

Urmas Pitsi, 10.dec.2020

1. Given a public RSA exponent 105, find suitable prime factors of the public modulus.

Answer: p=3, q=5
Solution:
We need to find primes *p* and *q* such that:

$$n = p \times q$$

$$y = x^{105} \; mod \; n$$

We observe that the prime factors of 105 are: 3, 5 and 7. In order to qualify as suitable factors for RSA exponent 105, $\Phi(n)$ must be coprime with 105. Where $\Phi(n) = (p-1) \times (q-1)$, Euler's totient function. Eg we could choose p=3 and q=5, then n = 3 * 5 = 15 and

$$\Phi(15) = (3-1) \times (5-1) = 8$$

As 8 is coprime to e = 105, the prime factors 3 and 5 are suitable for public exponent e=105.

2. Show that RSA is not secure against chosen plaintext attack.

Answer:
In public key cryptosystem (like RSA) it is always possible for the attacker to encrypt any plaintext, because attacker can use public encryption exponent of the target to encrypt the message. Now, if the plaintext space is small, then it would be trivial to brute-force: encrypt all plaintexts and compare them to target ciphertext.

$$c = m^e \; mod \; N$$

where *c*=cipher text, *m*=message, *e* and *N* are publicly available numbers.

Example in the lecture slides about "Semantic security" where attacker knows possible plaintexts (inputs): eg votes. Attacker encrypts all possible votes and compares results with ciphertext he wants to decrypt.

3. Find non-trivial square roots of Z_{77} .

Answer: 1, 76, 43, 34
Solution:
Prime factors of 77 are: 7 and 11. 77 = 7 * 11.
First find Bezout coefficients a,b in Z, such that 7a + 11b = 1. We find that: a = -3 and b = 2.
Using $Z_{77} = Z_7 \times Z_{11}$, we solve the equation:
$(u,v)^2 = (1,1)$ in $Z_7 \times Z_{11}$ that is equivalent to solving the equations:

$$u^2 \equiv 1 \ (mod\ 7)$$
$$v^2 \equiv 1 \ (mod\ 11)$$

The solutions are $u \in \{1,6\}$ and $v \in \{1,10\}$, and the solutions of
$(u,v)^2 = (1,1)$ in $Z_7 \times Z_{11}$ are the pairs (1, 1), (6, 10), (1, 10), (6, 1)

We use the map $Z_7 \times Z_{11} \rightarrow Z_{77}$:
$g(u,v) = 7 \times a \times v + 11 \times b \times u \ (mod\ 77) = 22u - 21v \ (mod\ 77)$, to map the solutions into Z_{77}.

Solutions are: 1, 76, 43, 34

4. How many iterations of Miller–Rabin is required to achieve confidence level of 99.9999%?

Answer: We need 10 iterations.
Solution: We need to find *k*, such that: $4^{-k} \leq 0.0001\%$. (ie 100% - 99.9999%)

```
p = 0.999999

for i in range(2, 12):
    conf = 1 - 4**(-i)
    print(f'k={i}, confidence={round(conf * 100, 6)}%, Is confidence greater than 99.9999%: {conf > p}')
```

```
k=2, confidence=93.75%, Is confidence greater than 99.9999%: False
k=3, confidence=98.4375%, Is confidence greater than 99.9999%: False
k=4, confidence=99.609375%, Is confidence greater than 99.9999%: False
k=5, confidence=99.902344%, Is confidence greater than 99.9999%: False
k=6, confidence=99.975586%, Is confidence greater than 99.9999%: False
k=7, confidence=99.993896%, Is confidence greater than 99.9999%: False
k=8, confidence=99.998474%, Is confidence greater than 99.9999%: False
k=9, confidence=99.999619%, Is confidence greater than 99.9999%: False
k=10, confidence=99.999905%, Is confidence greater than 99.9999%: True
k=11, confidence=99.999976%, Is confidence greater than 99.9999%: True
```

Source:
https://en.wikipedia.org/wiki/Miller%E2%80%93Rabin_primality_test

The error made by the primality test is measured by the probability for a composite number to be declared probably prime. The more bases *a* are tried, the better the accuracy of the test. It can be shown that if *n* is composite, then at most $1/4$ of the bases *a* are strong liars for *n*.[2][6] As a consequence, if *n* is composite then running *k* iterations of the Miller–Rabin test will declare *n* probably prime with a probability at most 4^{-k}.

5. Suppose an RSA cryptogram c1 = 537 was encrypted using public key (e = 18, n = 943), cryptogram c2 = 285 was encrypted using public key (e = 19, n = 943). Adversary Carol knows that the same message was encrypted to both of the recepients. What is m?

Answer: m = 717
Solution: brute-force solution. Assume search space is small, we iterate through possible messages, encrypt them and check whether they match known ciphertexts.

```
e1, n1, c1 = 18, 943, 537
e2, n2, c2 = 19, 943, 285

def rsa_encrypt(msg, e, n):
    return msg**e % n

def find_msg():
    for i in range(1, 1000):
        rsa1 = rsa_encrypt(i, e1, n1)
        rsa2 = rsa_encrypt(i, e2, n2)
        if rsa1 == c1 and rsa2 == c2:
            print(f'Found solution: message={i}')
            return i
    print('Solution NOT FOUND!')
    return None

find_msg()
```

```
Found solution: message=717

717
```

6. Suppose an RSA cryptogram c1 = 330 was encrypted using public key (e = 3, n = 377), cryptogram c2 = 34 was encrypted using public key (e = 3, n = 391), cryptogram c3 = 419 was encrypted using public key (e = 3, n = 589). Adversary Carol knows that the same message was encrypted to both of the recepients. What is m?

Answer: m = 102
Solution: Same as in previous task: brute-force.

```
e1, n1, c1 = 3, 377, 330
e2, n2, c2 = 3, 391, 34
e3, n3, c3 = 3, 589, 419

def rsa_encrypt(msg, e, n):
    return msg**e % n

def find_msg():
    for i in range(1, 1000):
        rsa1 = rsa_encrypt(i, e1, n1)
        rsa2 = rsa_encrypt(i, e2, n2)
        rsa3 = rsa_encrypt(i, e3, n3)
        if rsa1 == c1 and rsa2 == c2 and rsa3 == c3:
            print(f'Found solution: message={i}')
            return i
    print('Solution NOT FOUND!')
    return None

find_msg()
```

```
Found solution: message=102

102
```