Computational problems

Recall public key encryption

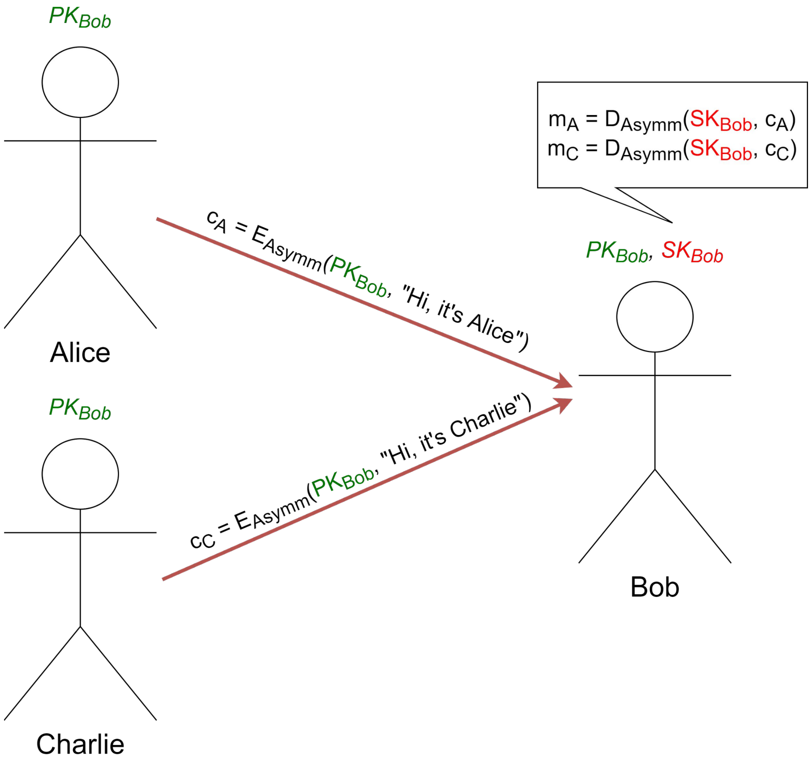
Alice and Charlie can use the public key to encrypt messages, but only Bob can decrypt ciphertexts using the secret key

How does it work? How are the keys related to each other?

We use backdoor functions.

Examples:

- Factorization problem
- Discrete logarithm problem



RSA cryptosystem

Setup

1. Generate ***p, q*** prime numbers and compute $n = p \times q$

RSA cryptosystem (cont.)

Setup

1. Generate ***p, q*** prime numbers and compute $n = p \times q$
2. Compute ***phi(n)*** $= (p-1)(q-1)$

Euler's totient function

Euler's totient function. Euler's theorem

$Z_n = \{0, 1, 2, 3, .., n\text{-}1\}$ e.g., $Z_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$Z_n^* = \{x \leftarrow Z_n \mid cmmdc(x, n) = 1\}$ e.g., $Z_{10}^* = \{1, 3, 7, 9\}$

Euler's totient function

$phi(n) = |Z_n^*|$ e.g., $phi(10) = |Z_{10}^*| = 4$

Euler's theorem

For every $a \leftarrow Z_n^*$, $a^{phi(n)} = 1 \bmod n$

1^4	3^4	7^4	9^4
1	81	2401	6561

RSA cryptosystem (cont.)

Key generation

- Generate ***p, q*** prime numbers and compute $n = p \times q$
- Compute ***phi(n)*** = *(p-1)(q-1)*
- Generate public exponent ***e*** s.t. *e* and *phi(n)* are relatively prime
- Compute private exponent ***d*** = e^{-1} *mod phi(n)*

..otherwise, *e* is not invertible mod *phi(n)*

RSA cryptosystem (cont.)

Key generation

- Generate ***p, q*** prime numbers and compute $n = p \times q$
- Compute ***phi(n)*** = *(p-1)(q-1)*
- Generate public exponent ***e*** s.t. *e* and *phi(n)* are relatively prime
- Compute private exponent ***d*** = e^{-1} *mod phi(n)*

Public key: (e, n) **Private key: (d, n)**

RSA cryptosystem (cont.)

Key generation

- Generate ***p, q*** prime numbers and compute $n = p \times q$
- Compute ***phi(n)*** $= (p-1)(q-1)$
- Generate public exponent ***e*** s.t. e and $phi(n)$ are relatively prime
- Compute private exponent ***d*** $= e^{-1} \bmod phi(n)$

Public key: (e, n) **Private key: (d, n)**

Encryption: $c = m^e \bmod n$

Decryption: $c^d = (m^e)^d = m^{1+k*phi(n)} = m * m^{k*phi(n)} = m * (\underbrace{m^{phi(n)}}_{= 1 \bmod n})^k = m * 1^k = m \bmod n$

1 mod n ← Euler's theorem

Problem

Q: What happens if two principals use the same modulus *n*?

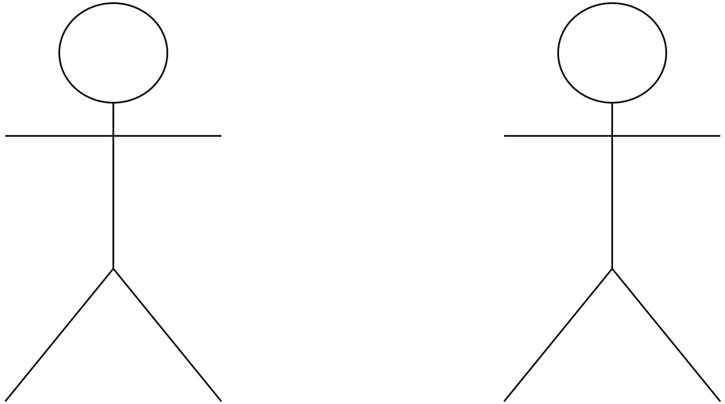
A: Each of them can recover the other principal's secret key

Let's solve the following exercise*:

$$\begin{matrix} \boldsymbol{PK_{Alice} = (e_A, n)} \\ \boldsymbol{SK_{Alice} = (d_A, n)} \end{matrix} \xrightarrow{\textit{Compute}} \begin{matrix} \boldsymbol{PK_{Bob} = (e_B, n)} \\ \boldsymbol{SK_{Bob} = (d_B, n)} \end{matrix}$$

Solved exercise in the laboratory PDF document

Alice	Bob
p, q	p, q
$n = p * q$	$n = p * q$
$phi(n) = (p-1) * (q-1)$	$phi(n) = (p-1) * (q-1)$
e_A, d_A	e_B, d_B
$e_A * d_A = 1 \bmod phi(n)$	$e_B * d_B = 1 \bmod phi(n)$



Alice

Bob

$\boldsymbol{PK_{Alice} = (e_A, n)}$
$\boldsymbol{SK_{Alice} = (d_A, n)}$

$\boldsymbol{PK_{Bob} = (e_B, n)}$
$\boldsymbol{SK_{Bob} = (d_B, n)}$

Steps

Step 1:

$(n, e_A, d_A) \rightarrow (p, q)$ i.e., factorize the modulus

Step 2:

compute $phi(n) = (p-1)*(q-1)$

Step 3:

compute $d_B = e_B^{-1} \; mod \; phi(n)$

Step 4:

Bob's secret key is (d_B, n)