

COMP3441/9441 Week 3
Public Key Cryptography, RSA Public Key System

- A *public key cryptosystem* is
- a pair of efficiently computable encryption and decryption functions E, D (parameterized by a key K)
 - an efficient way of generating a pair of keys K, K^{-1} so that
 1. $D_{K^{-1}}(E_K(M)) = M$ for all messages M , and
 2. it is hard to compute K^{-1} from K .

Given a public key cryptosystem, A can securely send M to B by
B generates K, K^{-1}
B \longrightarrow A: K
A \longrightarrow B: $E_K(M)$
B knows K^{-1} , $E_K(M)$, and computes $D_{K^{-1}}(E_K(M)) = M$
 E does not know K^{-1} so cannot easily do this.

- Comments
1. A public key cryptosystem must be able to withstand *adaptive chosen plaintext* cryptanalysis
Eve learns K so can compute
$$E_K(M_1), E_K(M_2), \dots$$
for messages M_1, M_2, \dots of Eve's choosing.
 2. If the message space is small, Eve can mount a brute force attack:
Fix: pad messages with a long random string N :
B \longrightarrow A: K
A \longrightarrow B: $E_K(M, N)$

Some constructions of public key cryptosystems

- Merkle (1975): a weak system requiring $O(n)$ messages and having an attack of $O(n^2)$
- Merkle and Hellman (1978): a system based on *knapsacks*, since cracked
- Rivest Shamir and Adleman (1978): most widely adopted system

More Number Theory

n and m are *relatively prime* if their only common divisor is 1.

Theorem: if a and n are relatively prime then there exists a unique number b mod n such that $ab \equiv 1 \pmod{n}$.

Fermat's Little Theorem: If p is prime and $1 \leq a \leq p - 1$ then $a^{p-1} \equiv 1 \pmod{p}$

The *reduced set of residues* mod n is the set of numbers $x \in \{1, \dots, n - 1\}$ such that x and n are relatively prime.

Euler's totient function: $\phi(n)$ is the size of the reduced set of residues mod n

Euler's generalization of Fermat's little theorem: If a and n are relatively prime, then $a^{\phi(n)} \equiv 1 \pmod{n}$.

Fact: If $n = p \cdot q$ where p and q are prime, then $\phi(n) = (p - 1) \cdot (q - 1)$

RSA Public Keys

- Widely used, e.g., in web browsers
- standard PKS for finance applications in Australia, France (not USA)
- patented in US, expired Sept 20, 2000
- not *proved* secure, but has withstood extensive cryptanalysis

1. Choose two random primes p and q , Define $n = p \cdot q$.
2. Choose e such that e and $(p-1) \cdot (q-1)$ are relatively prime.
3. Let d be a solution of

$$ed \equiv 1 \pmod{(p-1) \cdot (q-1)}$$

Discard p, q .

Public key $K = (e, n)$

Private key $K^{-1} = (d, n)$

$$E_K(M) = m^e \pmod n$$

$$E_{K^{-1}}(M) = m^d \pmod n$$

(where M is represented as a number $< n$)

Note

$$\begin{aligned} M^{ed} \pmod p &= M^{k \cdot (p-1) \cdot (q-1) + 1} \pmod p \\ &= M \cdot (M^{k(q-1)})^{(p-1)} \pmod p \\ &= M \pmod p \end{aligned}$$

(by Fermat's little theorem for $M \neq 0 \pmod p$, and calculation for $M = 0 \pmod p$)

Thus, p divides $M^{ed} - M$. Similarly q divides $M^{ed} - M$. Since p and q are different primes, $n = pq$ divides $M^{ed} - M$. I.e., $M^{ed} \pmod n = M$.

Thus $D_{K^{-1}}(E_K(M)) = (M^e \pmod n)^d \pmod n = M^{ed} \pmod n = M$.

Computing Exponentiation

Remark: we can compute $a^{2^k} \pmod n$ in k multiplications:

$$\begin{aligned} a^2 \pmod n &= a \cdot a \pmod n \\ a^4 \pmod n &= a^2 \cdot a^2 \pmod n \\ &\vdots \\ a^{2^k} \pmod n &= a^{2^{k-1}} \cdot a^{2^{k-1}} \pmod n \end{aligned}$$

If $x = x_0 + x_1 \cdot 2 + x_2 \cdot 2^2 + \dots + x_k \cdot 2^k$ is the binary decomposition of x then

$$a^x \pmod n = \prod_{i=1..k: x_i=1} a^{2^i}$$

So $k + \log_2 k$ multiplications suffice to compute $a^x \pmod n$

Security of RSA

RSA has withstood cryptanalysis, but

- it has not been proved to be secure
- factoring n would suffice to compute d . Factorization is thought not to be NP-complete.
- in theory factorization can be done efficiently on a quantum computer
- it is not known that factorization is the only way to crack RSA.

Key Size Issues

How large should the modulus n be in bits?

Computers & factorization algorithms keep getting faster.

US restricted export to modulus 512 bits until Jan 2000

Aug 1999 - 512 bit modulus cracked in 7 months using 300 workstations

Exercise: what is the modulus size in your browser?

Digital Signatures

Properties of Paper Signatures

- **Authentication:** convinces the recipient about who signed the document
- **Unforgeability:** only the signer can produce the signature
- **not reusable:** signature can't be transferred to another document
- **unalterability:** of signed document
- **non-repudiability:** signer can't repudiate signature

Notation for Signature Schemes

$S_K(M)$: message M , signed with key K

$V_K(M)$: verification of the signed message M

When K_s is a signature key and K_v is the corresponding verification key, we want

$$V_{K_v}(S_{K_s}(M)) = M$$

Ideally, if M' is not a message signed by K_s , we want

$$V_{K_v}(M') = \text{error}$$

A shared key signature scheme

Trent is an arbitrator trusted by both Alice and Bob.

Trent and Alice share a secret symmetric key $K_{A,T}$ // Trent and Bob share a secret symmetric key $K_{B,T}$

To send a signed message M to Bob,

$$A \longrightarrow T: E_{K_{A,T}}(M)$$

$$T \text{ computes } D_{K_{A,T}}(E_{K_{A,T}}(M)) = M$$

$$T \longrightarrow B: E_{K_{B,T}}(\text{"Alice sent } M'")$$

$$B \text{ computes } D_{K_{B,T}}(E_{K_{B,T}}(\text{"Alice sent } M'")) = \text{"Alice sent } M'$$

Disadvantages:

1. Trent needs to be trusted not to change M , or make up messages
2. Trent is a bottleneck

Signatures Using Public Key Cryptosystems

Alice has a public key K_A and a private key K_A^{-1} such that

$$D_{K_A}(E_{K_A^{-1}}(M)) = M$$

$$S_{K_A^{-1}}(M) = (M, E_{K_A^{-1}}(M))$$

$$V_{K_A}(M, sig) = \begin{cases} M & \text{if } D_{K_A}(sig) = M \\ \text{error} & \text{otherwise} \end{cases}$$

- If only A possesses K_A^{-1} , only A could have created $E_{K_A^{-1}}(M)$.
- K_A is public, so anyone can verify the signature

Note: usually you encrypt with public key K_A and decrypt with private key K_A^{-1} , and the equation is the reverse

$$D_{K_A^{-1}}(E_{K_A}(M)) = M$$

In case of RSA, public keys/private keys and encryption/decryption are symmetric, so either the "public key" $K_A = (e, n)$ or the "private key" $K_A^{-1} = (d, n)$ can be used as the secret signature key, and the other made public

$$\begin{aligned} D_{K_A}(E_{K_A^{-1}}(M)) &= (M^d)^e \bmod n \\ &= (M^e)^d \bmod n \\ &= D_{K_A^{-1}}(E_{K_A}(M)) \end{aligned}$$

Properties of Public Key Signatures

Provided A can be trusted to keep K_A^{-1} secret, this has the following properties:

- Authentication, unforgeability: Only A possesses K_A^{-1} , so only A can (easily) create $E_{K_A^{-1}}(M)$ from M
- document can't be altered, & signature is not reusable: if $M' \neq M$ then

$$V_{K_A}(M', E_{K_A^{-1}}(M)) = \text{error}$$

$$\text{since } M' \neq D_{K_A}(E_{K_A^{-1}}(M)) = M$$

Disanalogies

- unlike signed paper documents, digitally signed messages can be reused at will

So can't e.g. use this for digitally signed cheques unless add timestamps, sequence numbers.

- documents signed with private keys can be repudiated. E.g.
 1. Alice signs a contract to buy a house using K_A^{-1} , sends to the vendor Bob
 2. Alice changes her mind
 3. Alice anonymously posts K_A^{-1} to a hacker bulletin board
 4. When Bob tries to enforce the contract, Alice says: "I didn't sign that, a hacker stole my key!"