CS 223 Lec 25

Public Key Cryptosystems

Everyone has both public and private keys

Let A be some individual

P_A = A's public key

S_A = A's private (secret) key

(P_A, S_A) need the pair

$P_A()$ ← these are 1-1 functions from ASCII strings (or bit strings) to ASCII strings

$S_A()$ ←

Example

Alice and Bob

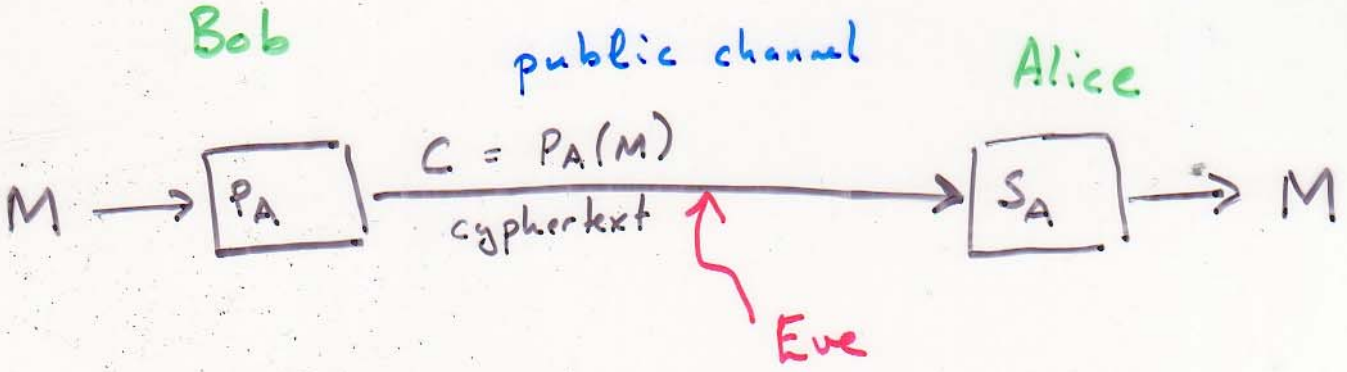
Bob has a message M that he wants to encrypt and send to Alice.

Properties of $P_A()$, $S_A()$

- each are inverses of one another

$$S_A(P_A(M)) = M$$

$$P_A(S_A(M)) = M$$



Digital Signatures

We want to "sign" a message so that the receiver trusts the sender is authentic.

Alice wants to send Bob a msg M' and wants to sign it first.

$$M' \xrightarrow{S_A} S_A(M') = \sigma$$

(σ: signature)

- sends (M', σ) using Bob's public key P_B
- So Bob receives $C = P_B(M', \sigma)$

When Bob receives C, he applies S_B ..

$$S_B(C) = S_B(P_B((M', \sigma)))$$

$$= (M', \sigma)$$

$$\sigma = S_A(M')$$

"Hi Bob, thanks for your message!

Signed Alice"

Bob knows P_A (Alice's public key)

Compute $P_A(\sigma) = P_A(S_A(M'))$

$$= M'$$

Bob checks that this really equals M'

RSA

- implements a public key cryptosystem
- believed to be secure.

1) select two large primes at random p, q (≥ 1024 bits) each

2) let $n = p \cdot q$

(Z_n^* = multiplicative group of order n)

$\phi(n) = (p-1)(q-1)$

↑ order of Z_n^*

3) select an odd, small integer e that is relatively prime to $\phi(n)$

4) find d s.t.

$$de = 1 \mod \phi(n)$$

5) Create keys

$P = (e, n)$ RSA public key

$S = (d, n)$ private key

$$P(M) = M^e \pmod n = C$$

$n \rightarrow$ length $= \beta$ bits

can be done $O(\beta^2)$ time.

$$S(C) = C^d \pmod n$$

$$\rightarrow = M$$

RSA correctness

$$M^{ed} = M \pmod n \quad \text{for all } M$$

The security of RSA rests on factoring being hard.
