

PUBLIC KEY ALGORITHMS

RSA

RSA

- Method
- Security strength
- Finding primes
- Choosing public/private keys
- Pitfalls
- Public-Key Cryptography Standard (PKCS)

RSA – Modular Exponentiation

- Normal exponentiation, then take remainder (e.g. $2^{10} = 4 \pmod{10}$)
- Exponentiation repeats itself
- i.e. $x^y \pmod{n} = x^{y \pmod{\Phi(n)}} \pmod{n}$
- e.g. $2^2 \pmod{10} = 4 = 2^6 \pmod{10} = 2^{10} \pmod{10}$
- Exponentiation with large numbers (256 bit) computationally intensive – efficient techniques must be used

RSA Overview

- Rivest, Shamir and Adleman
- Encryption/decryption and signatures
- Key length variable – typically 512 bits
- Message block size variable (< key length)
- Ciphertext block is length of key
- RSA used in key management

RSA Algorithm (I)

- Choose 2 large (≈ 256 bits) primes p and q
- $n = p \times q$
- Choose public key e relatively prime to $\Phi(n)$ ($\Phi(n) = (p - 1) \cdot (q - 1)$)
- private key d is multiplicative inverse of $e \bmod \Phi(n)$ (i.e. $d \times e = 1 \bmod \Phi(n)$)

RSA Algorithm (II)

- To encrypt message m ($m < n$)
ciphertext $c = m^e \bmod n$
- To decrypt ciphertext
message $m = c^d \bmod n$
- To sign a message m ($m < n$)
 $s = m^d \bmod n$
- To verify signature show that
 $m = s^e \bmod n$

RSA Algorithm (III)

- $de = 1 \bmod \Phi(n)$
- $x^{de} \bmod n = x^{de \bmod \Phi(n)} \bmod n = x \bmod n$
- Encrypting $c = m^e \bmod n$
- Decrypting $x = c^d \bmod n = m^{de} \bmod n = m$

Finding Primes

- Simple approach, for any n (≈ 512 bits) divide by all numbers $< \sqrt{n}$ and check for even division
- Use Euler/Fermat
 $a^{\Phi(n)} = a^{n-1} = 1 \bmod n$ if n prime
pick n as candidate prime and for any $a < n$ calculate a^{n-1}
if $a^{n-1} = 1$ prob that n is not prime is 10^{-13}
- Miller – Rabin test to remove doubt

Choosing Public/Private Key Pairs

- Method 1 – choose p , q and select e at random, check e is relatively prime to $\Phi(n) = (p - 1)(q - 1)$ and, if not, choose another e
- Method 2 – choose e and select primes p , q so that $(p-1)$ and $(q-1)$ are relatively prime to e
- For Method 2 may use small e (e.g. 3 or 65537) to improve performance

RSA Pitfalls (I)

- Limited number of messages m
 - attack by computing $m_i^e \bmod n$
 - protect by adding random padding to m_i
- Short message cube root attack
 - if $e = 3$ and $m < \sqrt[3]{n}$
 - $c = m^e \bmod n = m^3$
 - decrypt by taking cube root
 - protect by random padding

RSA Pitfalls (II)

- Chinese Remainder Cube Root Attack

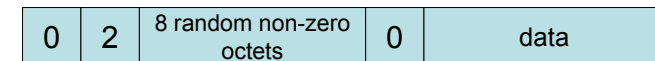
if $e = 3$ for 3 or more locations receiving same message and knowing c_1, c_2, c_3 and $(3, n_1), (3, n_2), (3, n_3)$ then from $m^3 \bmod n_i$ find $m^3 \bmod n_1 n_2 n_3$ and take cube root

Forged Signatures

any x is a signature of $x^e \bmod n$ but is x a valid message?

 - protect by specific padding

PKCS



First zero guarantees $m < n$
 2 denotes encryption and protects against short message cube root attack
 8 random non-zero octets protects against Chinese Remainder Theorem attack



1 denotes signature