PKC algorithm

RSA

RSA & Diffie-Hellman

- RSA - Ron Rivest, Adi Shamir and Len Adleman at MIT, in 1977.
  - RSA is a block cipher
  - The most widely implemented
- Diffie-Hellman
  - Exchange a secret key securely

RSA

- best known & widely used public-key scheme
- can be used to provide both secrecy & digital signatures
- security due to cost of factoring large numbers

Key Lengths

SKE length	PKC length
56 bits	384 bits
64 bits	512 bits
80 bits	768 bits
112 bits	1792 bits
128 bits	2304 bits

- certification authorities use 4096 bits

RSA Key Setup

- each user generates a public/private key pair by
  - select two large distinct primes at random - p, q.
  - compute their system modulus $n=p.q$
    - note $\phi(n)=(p-1)(q-1)$
  - select at random the encryption key e
    - where $1<e<\phi(n)$, $\gcd(e,\phi(n))=1$
  - solve the following equation to find decryption key d
    - $e.d=1 \bmod \phi(n)$ and $0\le d\le n$. How ?
  - publish the public encryption key: PU={e,n}
  - keep secret private decryption key: PR={d,n}
- It is critically important that the factors p & q of the modulus n are kept secret

A hacker problem

- If public key = (31,3599), then what is the private key?
  - From the problem e = 31, n = 3599
  - From this find p and q easily
  - Finding p and q, find ∅(n)
  - Having found out ∅(n), apply Extended Euclidean to find d

RSA Use

- to encrypt a message M the sender:
  - obtains public key of recipient **PU={e,n}**
  - computes: $c = m^e \bmod n$, where $0 \le m < n$
- to decrypt the ciphertext C the owner:
  - uses their private key **PR={d,n}**
  - computes: $m = c^d \bmod n$
- note that the message m must be smaller than the modulus n

RSA Example - Key Setup

- Select primes
  p=17 & q=11
- Compute n
  $n = pq$ =17 x 11=187
- Compute phi value
  $\varnothing(n)=(p-1)(q-1)=16 \times 10=160$
- Select encryption parameter
  e: gcd(e,160)=1; choose e=7
- Determine decryption parameter
  d: de=1 mod 160 and $d < 160$
  Value is d=23?
- Publish public key PU={7,187}
- Keep secret private key PR={23,187}

RSA Example - En/Decryption

- The sample RSA private/public operations are:
- Given message **M = 88** (note that **88<187**)
- Encryption is

$$\begin{aligned} C &= 88^7 \bmod 187 \\ &= 88^{(3+3+1)} \bmod 187 \\ &= (88^3 \bmod 187)(88^3 \bmod 187)(88 \bmod 187) \bmod 187 \\ &= (44 * 44 * 88) \bmod 187 \\ &= 11 \end{aligned}$$

- Decryption is

$$M = 11^{23} \bmod 187 = 88$$

RSA Example - Key Setup

- Select primes
  p=11 & q=3
- Compute n
  $n = pq = 11 \times 3 = 33$
- Compute phi value
  $ø(n)=(p-1)(q-1)=10 \times 2 = 20$
- Select encryption parameter
  gcd(e: gcd(e,20)=1; choose **$e = 3$**
- Determine decryption paramter
  d: de=1 mod 20 and **$d < 20$**
  Value is **d=7** since **7X3=21=20x1+1**
- Publish public key **PU={3,20}**
- Keep secret private key **PR={7,20}**

RSA Example - En/Decryption

- The sample RSA private/public operations are:
  - Given message **M = 7** (note that 7<33)
  - Encryption is

    $C = 7^3 \bmod 33$

    $= 343 \bmod 33$

    $= 13$
  - Decryption is

    $M = 13^7 \bmod 33$

    $= 13^{(3+3+1)} \bmod 33$

    $= (13^3 \bmod 33)(13^3 \bmod 33)(13 \bmod 33) \bmod 33$

    $= (2197 \bmod 33)(2197 \bmod 33)(13) \bmod 33$

    $= 19.19.13 \bmod 33 = 4693 \bmod 33$

    $= 7$

Another Example

- Consider the text grouping in the groups of three i.e.
  - **ATTACKXATXSEVEN = ATT ACK XAT XSE VEN**
- Represent the blocks in base 26 using A=0, B=1, C=2…..
  - ATT $= 0 * 26^2 + 19 * 26^1 + 19 = 513$
  - ACK $= 0 * 26^2 + 2 * 26^1 + 10 = 62$
  - XAT$= 23 * 26^2 + 0 * 26^1 + 19 = 15567$
  - XSE$= 23 * 26^2 + 18 * 26^1 + 4 = 16020$
  - VEN $= 21 * 26^2 + 4 * 26^1 + 13 = 14313$
- Next issue is designing the cryptosystem – selecting the parameters.
- What should be the value of n ?
- The value of n should be greater than 17575. How & why ?
- Let $p = 137$ and $q = 131$; so that $n = pq = 17947$

Another Example – Key Setup

- Compute phi value

  $\varnothing(n)=(p-1)(q-1)=136 \times 130 = 17680$

- Select encryption parameter

  gcd(e: gcd(e,17680)=1; choose ***e* = 3**

- Determine decryption parameter

  d: ***de*=1 mod 17680** and ***d* < 17680**

  Value is **d=11787**

- Publish public key **PU={**3,17947**}**
- Keep secret private key **PR={**11787,17947**}**

Another Example – En/Decryption

- The sample RSA private/public operations are:
- Given message **M = ATT = 513**
  - Encryption is

    $C = 513^3 \bmod 17947$

    $= 8363$
  - Decryption is

    $M = 8363^{11787} \bmod 17947$

    $= 513$
- Overall the plaintext is represented as the set of integers m

  {513,62,15567,16020,14313}
- Overall the ciphertext is represented as the set of integers c

  {8363,5017,11884,9546,13366}

Speed of RSA

- In H/W, the speed of RSA is 1000 times slower than DES
- In S/W, the speed of RSA is 100 times slower than DES
- It is assumed, the difference will remain….
  - even with the advancement in technology
- How to speed up the RSA operations ?

RSA Key Generation

- The users of RSA must
  - determine two primes at random - **`p, q`**
  - select either **`e`** or **`d`** and compute the other
- primes **`p,q`** must not be easily derived from modulus **`n=p.q`**
  - means must be sufficiently large
  - typically guess and use probabilistic test

RSA Key Generation (contd)

- exponents **e, d** are inverses, so use inverse algorithm to compute the other
- So, the basic operation involved, in either case, is
  - modular exponentiation
  - hence need to optimize the same
- Use square-and-multiply method

Computational complexity: Encryption

- encryption uses exponentiation to power e
- hence if e small, this will be faster
  - the most common choices are 3, 17 and 65537
    - X.509 recommends 65537
    - PEM recommends 3 while
    - PKCS#1 recommends either 3 or 65537
- but if e too small (eg e=3) security attack is possible
- if e fixed one must ensure **gcd(e,ø(n))=1**
  - i.e. reject any p or q not relatively prime to e

Computational complexity: Decryption

- decryption uses exponentiation to power d
  - this is likely to be large and insecure if not
- can use the Chinese Remainder Theorem (CRT)
  - to compute mod p & q separately and then combine to get the desired answer
  - approx 4 times faster than doing directly
- only owner of private key who knows values of p & q can use this technique

RSA Security

- RSA has been extensively analyzed for vulnerabilities by many researchers.
- After thirty years, one finds interesting attacks but,
  - none of them is critical
  - they mostly illustrate the dangers of improper usage of RSA
  - hence, securely implementing RSA is a nontrivial task.
- At an outset, the security depends on two mathematical problems
    - the problem of factoring large numbers
    - the RSA problem
  - both the above problems are considered to be hard
- How do we define the RSA problem?
- How do we define the RSA function ?
- What is breaking the RSA security?
- The most promising approach to solve the RSA problem appears to be
  - being able to factor n into p and q not knowing them a priori
- How to use the factors then, to break the security?