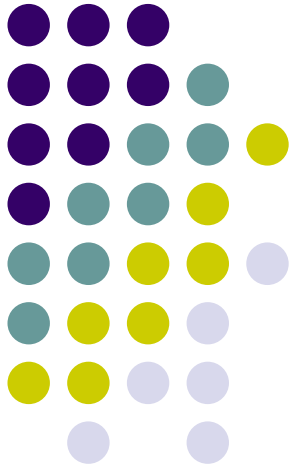
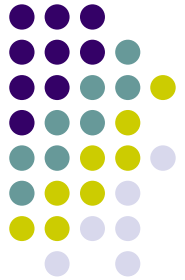
A Method for Obtaining Digital Signatures and Public-Key Cryptosystems – Part 2

Authors: R.L. Rivest, A. Shamir, and L. Adleman

Presented by Bing-Rong Lin



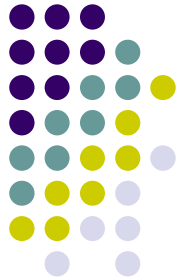
Scope



- The focus is on "Efficient Algorithm" and "Security" of RSA

- Ignore the proof of the referenced theorems, but
  (a) Theorems are given
  (b) Welcome to discuss with me after the class

Outline



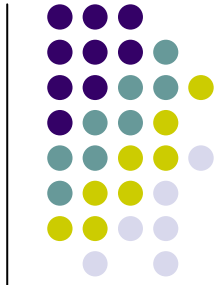
- Efficient algorithm
  - Encrypt & Decrypt
  - Find Large Prime Numbers (Primality Test)
  - Determine d & e
- Security of the Method
  - Factor n
  - Compute Φ(n) without (a)
  - Compute d without (a) & (b)
  - Compute D in some other way

Encrypt & Decrypt

- $D(M)=M^d \pmod n$
  - $d=d_k d_{k-1} \ldots d_1$ (binary representation)
    - $M^d = ((\ldots((A_k)^2 \times A_{k-1})^2 \times \ldots \times A_1$
      - If (d_k==1) A_k=M else A_k=1
      - Ex. $Y^{11}=Y^{1011}=(((Y)^2 \times 1)^2 \times Y)^2 \times Y$
  - $M^d \pmod n$

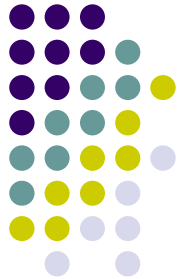
    $= ((\ldots((A_k \pmod n)^2 \times A_{k-1} \pmod n)^2 \times \ldots \times A_1 \pmod n)$
      - Hint: a*b mod c = (a mod c)*b mod C

Primality test



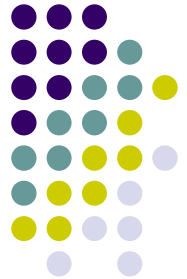
- Gcd(a,b)==1 & J(a,b)== $a^{(b-1)/2}$
  - If b is prime, it is always TURE
  - 0 < a < b
  - Euler's criterion
    - *p* is an odd prime, *a* is coprime to *p*
      - if *a* is quadratic residue modulo *p* (i.e. there exists a number *k* such that $k^2 \equiv a \pmod{p}$), then $\underline{\boldsymbol{a^{(p-1)/2} \equiv 1 \pmod{p}}}$
      - Else $\underline{\boldsymbol{a^{(p-1)/2} \equiv -1 \pmod{p}}}$
  - Jacobi symbol – J(a,b)
    - =0, if b divides a
    - =1, if *a* is quadratic residue modulo *b*
    - =-1, others

Primality test (cont.)



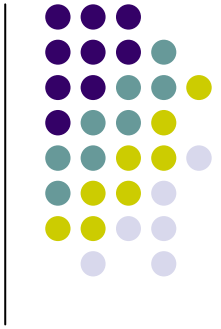
- J(a,b)
  - =1, if a=1
  - $=J(a/2,b)*(-1)^{(b*b-1)/8}$, if a is even
  - $= J(b(\mathrm{mod}\ a) , a)*(-1)^{(a-1)(b-1)/4}$, others
  - Hint: quadratic reciprocity

Determine d & e



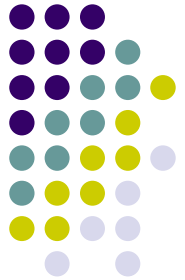
- Choose d
  - Relatively prime to $\phi(n)$
  - d should be chosen from a large enough set
- Compute $e \equiv d^{-1} \bmod \phi(n)$
  - Above equation is identical to $a \cdot \phi(n) + e \cdot d = 1$
  - Based on Euclid's algorithm
    - $X_0 = \phi(n)$, $X_1 = d$
      - $X_{i+1} \equiv X_{i-1} \pmod{X_i}$
      - Compute a_i and b_i such that $X_i = a_i \times X_0 + b_i \times X_1$
        - If $X_k = 1$ then $e = b_k$

Example



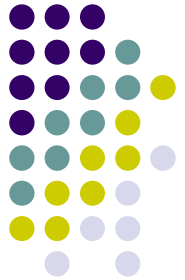
- $X_0 = \phi(n) = 2668$, $X_1 = d = 157$
  - $X_{i+1} \equiv X_{i-1} \pmod{X_i}$
    - $X_2 = 2668 \bmod 157 = 156$
    - $X_3 = 157 \bmod 156 = 1$
  - $X_i = a_i \times X_0 + b_i \times X_1$
    - $X_2 = 156 = 2668 - 16 * 157 = X_0 + 16*X_1$
    - $X_3 = 1 = 157 - 1 * 156 = X_1 + 1*X_2$
      $= X_1 + 1*(X_0 + 16*X_1)$
      $= X_0 + 17*X_1$
    - $e = 17$

Security of the Method



- No techniques exist to prove that an encryption scheme is secure
- Factoring large number is a well-know problem that has been worked on for 300 years.
- Show breaking the system is equivalent to factoring problem.

How to break?



- Input
  - n, e
- Possible breaking approaches
  - (a) Factor n
    - Well-known problem, hard
  - (b) Compute $\phi(n)$ without (a)
    - Equivalent to factoring
  - (c) Compute d/d' without (a)&(b)
    - Equivalent to factoring
  - (d) Compute D in some other way
    - "May" equivalent to factoring

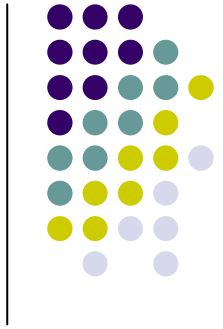
Compute $\phi(n)$ without (a)

- If it is possible, then n can be easily factored
  - $\phi(n) = (p-1)(q-1) = n - (p+q) + 1$
  - $(p-q)^2 = (p+q)^2 - 4n$
  - $q = (p+q) + (p-q)$

Compute d/d’ without (a)&(b)

- d
  - If it is possible, then n can be easily factored
    - $e*d - 1 = k * \phi(n)$
    - [6] “Reimann’s hypothesis and tests for primality” shows n can be factored by using any multiple of $\phi(n)$
- d’ (find a key equivalent to d)
  - If it is possible, then n can be easily factored
    - $d' = d + k*\phi(n)$
    - Finding one enables n to be factored.

Conclusions



- The authors show the efficient algorithms of
  - encrypt & decrypt
  - Primality test -- has been largely superseded by the Miller-Rabin primality test
  - Compute e
- The security of the method is based on factoring problem