Lecture 9

Rivest-Shamir-Adelman (RSA)

CS 450/650

Fundamentals of Integrated Computer Security

Euler's Totient Function

- Euler's totient function, written $\phi(n)$, is defined as the number of positive integers less than *n* and relatively prime to *n*
- By convention, $\phi(1) = 1$
- Examples: $\phi(7) = 6$, $\phi(4) = 2$
- For a prime p, $\phi(p) = p-1$
- Suppose we have two primes p and q, with p!=q. Then we have, for n=pq, $\phi(n) = \phi(pq) = \phi(p) * \phi(q) = (p-1)(q-1)$

Euler's Totient Function

$$\varphi(n) = \varphi\left(p_1{}^{k_1}\right)\varphi\left(p_2{}^{k_2}\right)\cdots\varphi\left(p_r{}^{k_r}\right)$$

$$= n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_r}\right)$$

Ø(30)=?
Ø(25)=?
Ø(100)=?

n	$\phi(n)$
1	1
2	1
3	2
4	2
5	4
6	2
7	6
8	4
9	6
10	4

n	$\phi(n)$
11	10
12	4
13	12
14	6
15	8
16	8
17	16
18	6
19	18
20	8

n	$\phi(n)$
21	12
22	10
23	22
24	8
25	20
26	12
27	18
28	12
29	28
30	8

Modular Arithmetic

- $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
  - $[(11 \bmod 10) + (12 \bmod 10)] \bmod 10 = (11 + 12) \bmod 10$
- $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
  - $[(11 \bmod 10) - (12 \bmod 10)] \bmod 10 = (11 - 12) \bmod 10$
- $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$
  - $[(11 \bmod 10) \times (12 \bmod 10)] \bmod 10 = (11 \times 12) \bmod 10$

Modular Arithmetic

- $(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$

RSA

- To encrypt message M compute
  - $c = m^e \bmod n$
- To decrypt ciphertext c compute
  - $m = c^d \bmod n$
- Parameters to decide
  - e, d, n

Key Choice

- Let p and q be two large prime numbers
- Let n = pq
- Compute $\phi(n) = \phi(p)\phi(q) = (p - 1)(q - 1)$, where ϕ is Euler's totient function. This value is kept private.

Key Choice

- Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$; i.e., e and $\phi(n)$ are coprime.
  - e is released as the encryption key.
- Determine d as $d \equiv e^{-1} \pmod{\phi(n)}$; or solve for d given $d \cdot e \equiv 1 \pmod{\phi(n)}$
  - d is the decryption key
- (e, d) is the RSA key pair

The Correctness of RSA

Given $c = m^e \bmod n$ we must show

- $m = c^d \bmod n$

- $c^d \bmod n = (m^e \bmod n)^d \bmod n$
- $(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$
- $m^{ed} = m^{1+h\varphi(n)} = m\left(m^{\varphi(n)}\right)^h \equiv m(1)^h \equiv m \pmod{n}$
  - Since $d \cdot e \equiv 1 \pmod{\phi(n)}$, we can write $ed = 1 + h\phi(n)$ for some non-negative integer h
  - *Euler's Theorem:* If x is relatively prime to n then $x^{\varphi(n)} \bmod n = 1$

Example

- Select primes p=11, q=3
- n = p* q = 11*3 = 33
- Compute $\phi(33) = \phi(11)\phi(3) = (11-1)*(3-1)=20$
- Choose e = 3

Example (cont.)

- Compute d such that

$$e * d \bmod \phi(n) = 1$$

$$3 * d \bmod 20 = 1$$

$$d = 7$$

Public key = (n, e) = (33, 3)

Private key = (d) = (7)

Example (cont.)

- Now say we want to encrypt message m = 5
- $c = m^e \bmod n = 5^3 \bmod 33 = 125 \bmod 33 = 26$
  – Hence the ciphertext c = 26
- To check decryption, we compute

$$m = c^d \bmod n = 26^7 \bmod 33 = 5$$

RSA

- Why the security of RSA depends on the hardness of big integer factoring?
  - Recall that e and n are public
  - If attacker can factor n, he can use e to easily find d
    - since ed mod (p-1)(q-1) = 1
  - Factoring the modulus breaks RSA
  - It is not known whether factoring is the only way to break RSA

RSA key-length strength

- RSA has challenges for different key-lengths
  - RSA-140 (bit size of n is 140)
    - Factored in 1 month using 200 machines in 1999
  - RSA-155
    - Factored in 3.7 months using 300 machines in 1999
  - RSA-160
    - Factored in 20 days in 2003
  - RSA-200
    - Factored in 18 month in 2005
  - RSA-210, RSA-220, RSA-232, … RSA-2048