SE-4920: Lectures 10–11
Public key algorithms

- Reading
  - Chapter 6 (pp. 147–160, 163–170)
- Today's Outcomes
  - Perform modular arithmetic (addition, multiplication, exponentiation)
  - Apply basic theory of modular arithmetic (Totient function, Euler's theorem, …)
  - Execute and apply the RSA algorithm for encryption and digital signatures
  - Execute and apply the Diffie-Hellman algorithm for establishing a shared secret

1

Modular arithmetic

- "mod n" arithmetic
  - Like normal arithmetic, but final result is taken mod n
- mod 6 addition
  - 2 + 3 = 5
  - 4 + 5 = 3
  - 4 + 2 = 0

2

Modular arithmetic: additive inverse

- In regular arithmetic, (a) + (-a) = 0
  - -a is the additive inverse of a
- mod 6 additive inverses
  - 2 + 4 = 0, therefore -2 = 4
  - x + 2 + 4 = x

3

Modular arithmetic: multiplication

- Consider mod 10 multiplication
  - $2 \cdot 3 = 6$
  - $2 \cdot 7 = 14 = 4 \bmod 10$
  - $3 \cdot \{0, 1, \ldots, 9\} = \{0, 3, 9, 2, 5, 8, 1, 4, 7\}$
  - $8 \cdot \{0, 1, \ldots, 9\} = \{0, 8, 6, 4, 2, 0, 8, 6, 4\}$
- Multiplying by 3 is reversible, but 8 isn't
  - Big difference from integers, reals, rationals, etc.
  - 3 has a multiplicative inverse, but 8 doesn't
    - Finding the multiplicative inverse is easy
      - But we won't cover it here
      - Euclid's algorithm in chapter 7
    - $3^{-1} = 7$
    - $x \cdot 3 \cdot 7 = (x \cdot 21) \bmod 10 = ((x \bmod 10) \cdot (21 \bmod 10)) \bmod 10 = x \bmod 10$

4

Modular arithmetic: multiplicative inverses

- When does a multiplicative inverse exist?
  - When the multiplier, x, is relatively prime to the number of elements, n.
    - GCD(x, n) = 1
    - For n = 10,
      - x = {1, 3, 7, 9} are the only values that give a GCD of 1
      - Only multipliers with multiplicative inverses: {1, 7, 3, 9}
- How many numbers less than n are relatively prime to n?
  - The answer is Euler's totient function, $\varphi(n)$
    - For a prime, $\varphi(p) = p - 1$
    - For a product of 2 primes, $\varphi(pq) = pq - p - q + 1 = (p-1)(q-1)$
    - These are the 2 relevant cases for chapter 6

5

Modular arithmetic: exponentiation

- $3^7 = 2187 = 7 \bmod 10$
- Taking remainder earlier gives same result
  - $3^7 = 3^3 3^4 = 7 \cdot 1 = 7$
- Construct x^y table, with exponents from 1 to 12
  - Note that $x^n = x^{n+4} \bmod 10$
  - Fact: $x^y \bmod n = x^{y \bmod \varphi(n)} \bmod n$
    - For n prime or product of distinct primes (no p^2)
  - If $y = 1 \bmod \varphi(n)$
    - $x^y = x \bmod n$ (useful fact for RSA algorithm)

6

RSA

- Inventors Rivest, Shamir, and Adleman
- Public key algorithm
- Variable key length, commonly 512 b
- Variable block size
  - Plaintext shorter than key
  - Ciphertext same length as key
- Much slower than DES and most secret key algorithms
  - Commonly used for exchanging a secret key (*e.g.*, for DES)
    - With no pre-shared secret

7

RSA algorithm: Key generation

- Choose 2 large, secret, primes, p and q
  - Around 256 bits each
- n = pq
  - Given n, it is impractical to find p and q
- Public key <e, n>
  - Choose e (it can be small) that is relatively prime to $\varphi(n)$
    - You know $\varphi(n) = (p-1)(q-1)$
      - But others would need to do a brute force search
- Private key <d, n>
  - Find $d = e^{-1} \bmod \varphi(n)$
    - Easy only if you know $\varphi(n)$

8

RSA algorithm: encryption / decryption

- Encrypt
  - $c = m^e \bmod n$ (where $m < n$)
- Decrypt
  - $c^d = m^{de} \bmod n = m^1 \bmod n$
  - Since $de = 1 \bmod \varphi(n)$
- Sign
  - $s = m^d \bmod n$
- Verify
  - $s^e \bmod n = m^{de} \bmod n = m^1 \bmod n$

9

RSA example

- $p = 11$, $q = 17$
- $e = 3$ (verify compatible with $\varphi(n)$)
- $d = 107$
  - Easy to find with Euclid's algorithm (chapter 7)
- $m = 127$
- Calculate c (172)
- Calculate m from c (indeed 127)
  - How can this be done efficiently?
  - Do we really need to calculate 172^{107} ??!!

10

Generating RSA keys

- Can be expensive (a few minutes)
  - But must be feasible
- Choose p and q randomly and test for primality
  - Can get ***very*** high probability without much work
  - The odds are against us on any 1 number
    - Probability of random n being prime is about $1/(\ln n)$
    - Linear increase in $\ln n$ with length of prime
      - About 1/21 for numbers around 2^{30}
      - About 1/210 for numbers around 2^{300}

11

Generating RSA keys: primality tests

- Fermat's Little Theorem
  - For $a < p$, $a^{p-1} = 1 \bmod p$
  - It can happen for non-primes for some a's, but rare
    - About 1 in 10^{13} chance for hundred-digit p
      - So, try a few a's and make sure you get 1 for all
  - Carmichael numbers (the main type of pseudoprime) will give 1 for all a's, but they are not prime
    - Only 585,355 Carmichael numbers less than 10^{17}
- There are even better algorithms
  - 2006 standard is Miller and Rabin's algorithm (§6.3.4.2.1)

12

RSA: Choosing d and e

- GCD((p-1)(q-1), e) = 1
- Select e at random and test for relative primarily
- Or, Choose e and then select p and q for relative primality
    - Certain choices of e will make public key computations easier without sacrificing security

13

RSA: 3 is the smallest possible e

- 2 does not work since (p-1)(q-1) is even
- Encryption only needs 2 multiplies!
- Weaknesses (can work around)
    - Short message, $c = m^3 < n$, take $\sqrt[3]{c}$
        - Pad short messages, usually with random number
    - Encrypting to 3 recipients
        - If you know the 3 keys (they are public!) and ciphertext, can compute $m^3 \bmod n_1 n_2 n_3$
            - Method in chapter 7 (Chinese Remainder Theorem)
            - Simple cube root
        - Padding each message uniquely (even with user ID) will solve this problem
- Need GCD((p-1)(q-1), 3) = 1
    - GCD(p-1, 3) = 1 and GCD(q-1, 3) = 1
    - p and q must be 2 mod 3
        - 1 would defeat GCD
        - 0 would be non-prime

14

RSA: 65537 is another common e

- $65537 = 2^{16}+1$
    - Largest known Fermat prime ($2^{2^n}+1$)
    - Total of 17 multiplies to exponentiate
        - Expect about 384 multiplies for random acceptable 256-bit e
            - And (on the order of) that many integer divisions
- Avoids most problems with 3

15

PKCS: Public-Key Cryptography Standard

- Standards including message formats for applying RSA for encryption and signing
- Random padding for encryption
  - Built-in workarounds to problems with e = 3
- Signing
  - Done on message digest / hash
    - Generally too expensive for entire message
  - Digest type (hash function) is part of signed quantity,
    - so attacker cannot attach your signature
    - to a different (fake) message
    - using a weaker hash (*e.g.*, a compromised one)

16

Diffie-Hellman

- Oldest PK system still in use
- Agree on a shared, secret key using only public communication channels
  - Does not provide authentication

17

Diffie-Hellman

- Agree on p (large prime)
- Agree on g < p
  - g must be a "primitive root" of p
    - $g^1 \ldots g^{p-1}$ are a permutation of 1…p-1
  - There are φ(φ(p)) = φ(p-1) choices
    - 2 or 3 often works
  - No direct way to calculate
    - But there are more efficient methods than searching all primes

18

Diffie-Hellman

- Alice picks S_A and Bob picks S_B at random (512 bits)
- Alice computes T_A and Bob computes T_B
- Ts are exchanged
- Compute shared secret, K
- Everyone knows the Ts
  - Not enough to compute K
  - T → S is the "discrete logarithm problem", and mathematicians haven't solved it

$$T_A = g^{S_A} \bmod p$$
$$T_B = g^{S_B} \bmod p$$

$$K = T_B^{S_A} \bmod p = T_A^{S_B} \bmod p = g^{S_A S_B}$$

19

Man-in-the-Middle Attack

- Diffie-Hellman has no authentication
- An active attacker between Alice and Bob
  - who can intercept and rewrite the communication
  - could share a key with each of them
  - and decrypt/re-encrypt the traffic in each direction
- Still secure against passive attack

20