Public Key Cryptography

Public Key Cryptography

- Two keys, one to encrypt, another to decrypt
  - Alice uses Bob's **public key** to encrypt
  - Only Bob's **private key** decrypts the message
- Based on "trap door, one way function"
  - "One way" means easy to compute in one direction, but hard to compute in other direction
  - Example: Given p and q, product $N = pq$ easy to compute, but hard to find p and q from N
  - "Trap door" is used when creating key pairs

Public Key Cryptography

- Encryption
  - Suppose we **encrypt** M with Bob's public key
  - Bob's private key can **decrypt** C to recover M
- Digital Signature
  - Bob **signs** by "encrypting" with his private key
  - Anyone can **verify** signature by "decrypting" with Bob's public key
  - But only Bob could have signed
  - Like a handwritten signature, but much better…

Knapsack

Knapsack Problem

- Given a set of n weights $W_0, W_1, \ldots, W_{n-1}$ and a sum S, find $a_i \in \{0,1\}$ so that

  $$S = a_0W_0 + a_1W_1 + \ldots + a_{n-1}W_{n-1}$$

  (technically, this is the *subset sum* problem)
- Example
  - Weights $(62,93,26,52,166,48,91,141)$
  - Problem: Find a subset that sums to $S = 302$
  - Answer: $62 + 26 + 166 + 48 = 302$
- The (general) knapsack is NP-complete

Knapsack Problem

- **General knapsack** (GK) is hard to solve
- But **superincreasing knapsack** (SIK) is easy
- SIK — each weight greater than the *sum of all previous weights*
- **Example**
  - Weights $(2,3,7,14,30,57,120,251)$
  - Problem: Find subset that sums to $S = 186$
  - Work from largest to smallest weight
  - Answer: $120 + 57 + 7 + 2 = 186$

Knapsack Cryptosystem

1. Generate superincreasing knapsack (SIK)
2. Convert SIK to "general" knapsack (GK)
3. **Public Key:** GK
4. **Private Key:** SIK and conversion factor

- Goal…
  - Easy to encrypt with GK
  - With private key, easy to decrypt (solve SIK)
  - Without private key, Trudy has no choice but to try to solve GK

Example

- Start with (2,3,7,14,30,57,120,251) as the SIK
- Choose $m = 41$ and $n = 491$ (m, n relatively prime, n exceeds sum of elements in SIK)
- Compute "general" knapsack

$$2 \cdot 41 \bmod 491 = 82$$
$$3 \cdot 41 \bmod 491 = 123$$
$$7 \cdot 41 \bmod 491 = 287$$
$$14 \cdot 41 \bmod 491 = 83$$
$$30 \cdot 41 \bmod 491 = 248$$
$$57 \cdot 41 \bmod 491 = 373$$
$$120 \cdot 41 \bmod 491 = 10$$
$$251 \cdot 41 \bmod 491 = 471$$

- "General" knapsack: (82,123,287,83,248,373,10,471)

Knapsack Example

- **Private key:** (2,3,7,14,30,57,120,251)

  $m^{-1} \bmod n = 41^{-1} \bmod 491 = 12$
- **Public key:** (82,123,287,83,248,373,10,471), n=491
- Example: Encrypt 10010110

  $82 + 83 + 373 + 10 = 548$
- To decrypt, use private key…
  - $548 \cdot 12 = 193 \bmod 491$
  - Solve (easy) SIK with $S = 193$
  - Obtain plaintext 10010110

Knapsack Weakness

- **Trapdoor:** Convert SIK into "general" knapsack using modular arithmetic
- **One-way:** General knapsack easy to encrypt, hard to solve; SIK easy to solve
- This knapsack cryptosystem is **insecure**
  - Broken in 1983 with Apple II computer
- "General knapsack" is not general enough!
  - This special case of knapsack is easy to break

RSA

RSA

- Invented by Clifford Cocks (GCHQ) and Rivest, Shamir, and Adleman (MIT)
  - RSA is the *gold standard* in public key crypto
- Let p and q be two large prime numbers
- Let $N = pq$ be the **modulus**
- Choose e relatively prime to $(p-1)(q-1)$
- Find d such that $ed = 1 \bmod (p-1)(q-1)$
- **Public key** is (N,e)
- **Private key** is d

RSA

- Message M is treated as a number
- To encrypt M we compute
  $C = M^e \bmod N$
- To decrypt ciphertext C, we compute
  $M = C^d \bmod N$
- Recall that e and N are public
- If Trudy can factor $N = pq$, she can use e to easily find d since $ed = 1 \bmod (p-1)(q-1)$
- So, **factoring the modulus breaks RSA**
  - Is factoring the only way to break RSA?

Does RSA Really Work?

- Given $C = M^e \bmod N$ we want to show that $M = C^d \bmod N = M^{ed} \bmod N$
- We'll need **Euler's Theorem:** https://en.wikipedia.org/wiki/Euler%27s_theorem
  If x is relatively prime to n then $x^{\varphi(n)} = 1 \bmod n$
- Facts:
  1) $ed = 1 \bmod (p - 1)(q - 1)$
  2) By definition of "mod", $ed = k(p - 1)(q - 1) + 1$
  3) $\varphi(N) = (p - 1)(q - 1)$
- Then $ed - 1 = k(p - 1)(q - 1) = k\varphi(N)$
- So, $C^d = M^{ed} = M^{(ed - 1) + 1} = M \cdot M^{ed - 1} = M \cdot M^{k\varphi(N)}$
  $= M \cdot (M^{\varphi(N)})^k \bmod N = M \cdot 1^k \bmod N = M \bmod N$

Simple RSA Example

- Example of ***textbook*** RSA
  - Select "large" primes $p = 11, q = 3$
  - Then $N = pq = 33$ and $(p - 1)(q - 1) = 20$
  - Choose $e = 3$ (relatively prime to 20)
  - Find d such that $ed = 1 \bmod 20$
    - We find that $d = 7$ works
- **Public key:** $(N, e) = (33, 3)$
- **Private key:** $d = 7$

Simple RSA Example

- **Public key:** $(N, e) = (33, 3)$
- **Private key:** $d = 7$
- Suppose message to encrypt is $M = 8$
- Ciphertext C is computed as

$$C = M^e \bmod N = 8^3 = 512 = 17 \bmod 33$$

- Decrypt C to recover the message M by

$$M = C^d \bmod N = 17^7 = 410{,}338{,}673$$
$$= 12{,}434{,}505 * 33 + 8 = 8 \bmod 33$$

More Efficient RSA (1)

- Modular exponentiation example
  - $5^{20} = 95367431640625 = 25 \bmod 35$
- A better way: **repeated squaring**
  - $20 = 10100$ base 2
  - $(1, 10, 101, 1010, 10100) = (1, 2, 5, 10, 20)$
  - Note that $2 = 1 \cdot 2$, $5 = 2 \cdot 2 + 1$, $10 = 2 \cdot 5$, $20 = 2 \cdot 10$
  - $5^1 = 5 \bmod 35$
  - $5^2 = (5^1)^2 = 5^2 = 25 \bmod 35$
  - $5^5 = (5^2)^2 \cdot 5^1 = 25^2 \cdot 5 = 3125 = 10 \bmod 35$
  - $5^{10} = (5^5)^2 = 10^2 = 100 = 30 \bmod 35$
  - $5^{20} = (5^{10})^2 = 30^2 = 900 = 25 \bmod 35$
- No huge numbers and it's efficient!

More Efficient RSA (2)

- Use $e = 3$ for all users (but not same N or d)
  - + Public key operations only require 2 multiplies
  - o Private key operations remain expensive
  - - If $M < N^{1/3}$ then $C = M^e = M^3$ and cube root attack
  - - For any M, if C_1, C_2, C_3 sent to 3 users, cube root attack works (uses Chinese Remainder Theorem)

https://en.wikipedia.org/wiki/Chinese_remainder_theorem

- Can prevent cube root attack by padding message with random bits
- Note: $e = 2^{16} + 1$ also used ("better" than $e = 3$)