Public Key Cryptography

- **Chapter 7**

Public Key Cryptography

- Two keys
  - Sender uses recipient's **public key** to encrypt
  - Receiver uses his **private key** to decrypt
- Based on **trap door, one way function**
  - Easy to compute in one direction
  - Hard to compute in other direction
  - "Trap door" used to create keys
  - Example: Given p and q, product N=pq is easy to compute, but given N, it is hard to find p and q

Public Key Cryptography

- Encryption
  - Suppose we encrypt M with Bob's public key
  - Only Bob's private key can decrypt to find M
- Digital Signature
  - **Sign** by "encrypting" with private key
  - Anyone can **verify** signature by "decrypting" with public key
  - But only private key holder could have signed
  - Like a handwritten signature (and then some)

Knapsack

Knapsack Problem

- Given a set of n weights $W_0, W_1, \ldots, W_{n-1}$ and a sum S, is it possible to 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 "subset sum" problem)
- **Example**
  - Weights (62,93,26,52,166,48,91,141)
  - Problem: Find 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 into "general" knapsack (GK)
3. **Public Key:** GK
4. **Private Key:** SIK plus conversion factors

- Easy to encrypt with GK
- With private key, easy to decrypt (convert ciphertext to SIK)
- Without private key, must solve GK (???)

Knapsack Cryptosystem

- Let (2,3,7,14,30,57,120,251) be the SIK
- Choose m = 41 and n = 491 with m, n rel. prime and n greater than sum of elements of SIK
- 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,
  - $548 \cdot 12 = 193 \bmod 491$
  - 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
  - The attack uses **lattice reduction**
- "General knapsack" is not general enough!
- This special knapsack is easy to solve!

RSA

RSA

- Invented by Cocks (GCHQ), independently, by Rivest, Shamir and Adleman (MIT)
- 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 s.t. $ed = 1 \bmod (p-1)(q-1)$
- **Public key** is (N,e)
- **Private key** is d

RSA

- To encrypt message M compute
  - $C = M^e \bmod N$
- To decrypt C compute
  - $M = C^d \bmod N$
- Recall that e and N are public
- If attacker can factor N, he can use e to easily find d since $ed = 1 \bmod (p-1)(q-1)$
- Factoring the modulus breaks RSA
- It is not known whether factoring is the only way to break RSA

Does RSA Really Work?

- Given $C = M^e \bmod N$ we must show
  - $M = C^d \bmod N = M^{ed} \bmod N$
- We'll use **Euler's Theorem**
  - If x is relatively prime to n then $x^{\varphi(n)} = 1 \bmod n$
- Facts:
  - $ed = 1 \bmod (p-1)(q-1)$
  - By definition of "mod", $ed = k(p-1)(q-1) + 1$
  - $\varphi(N) = (p-1)(q-1)$
  - Then $ed - 1 = k(p-1)(q-1) = k\varphi(N)$
- $C^d = (M^e)^d = M^{(ed-1)+1} = M \cdot M^{ed-1} = M \cdot M^{k\varphi(N)}$
  $= M \cdot (M^{\varphi(N)})^k \bmod N = M \cdot 1^k \bmod N = M \bmod N$

Simple RSA Example

- Example of 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 $M = 8$
- 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

- 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 $2 = 1 \cdot 2$, $5 = 2 \cdot 2 + 1$, $10 = 2 \cdot 5$, $20 = 2 \cdot 10$
  - $5^1 = 5 \bmod 35$
  - $5^2 = (5^1)^2 = 5^2 = 25 \bmod 35$
  - $5^5 = (5^2)^2 \cdot 5^1 = 25^2 \cdot 5 = 3125 = 10 \bmod 35$
  - $5^{10} = (5^5)^2 = 10^2 = 100 = 30 \bmod 35$
  - $5^{20} = (5^{10})^2 = 30^2 = 900 = 25 \bmod 35$
- Never have to deal with huge numbers!

Diffie-Hellman

Diffie-Hellman

- Invented by Williamson (GCHQ) and, independently, by D and H (Stanford)
- A "key exchange" algorithm
  - Used to establish a shared symmetric key
- Not for encrypting or signing
- Security rests on difficulty of **discrete log** problem: given g, p, and g^k mod p find k

Diffie-Hellman

- Let p be prime, let g be a **generator**
  - For any $x \in \{1,2,\ldots,p-1\}$ there is n s.t. $x = g^n \bmod p$
- Alice selects secret value a
- Bob selects secret value b
- Alice sends $g^a \bmod p$ to Bob
- Bob sends $g^b \bmod p$ to Alice
- Both compute shared secret $g^{ab} \bmod p$
- Shared secret can be used as symmetric key

Diffie-Hellman

- Suppose that Bob and Alice use $g^{ab} \bmod p$ as a symmetric key
- Trudy can see $g^a \bmod p$ and $g^b \bmod p$
- Note $g^a g^b \bmod p = g^{a+b} \bmod p \neq g^{ab} \bmod p$
- If Trudy can find a or b, system is broken
- If Trudy can solve **discrete log** problem, then she can find a or b

Diffie-Hellman

- **Public:** g and p
- **Secret:** Alice's exponent a, Bob's exponent b

$g^a \bmod p \longrightarrow$

$\longleftarrow g^b \bmod p$

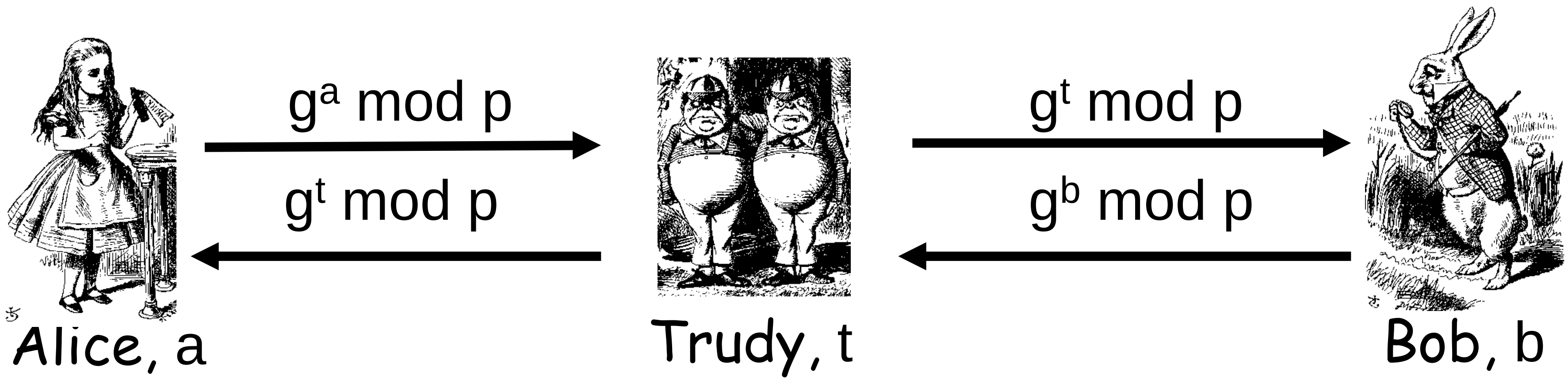
Alice, a

Bob, b

- Alice computes $(g^b)^a = g^{ba} = g^{ab} \bmod p$
- Bob computes $(g^a)^b = g^{ab} \bmod p$
- Could use $K = g^{ab} \bmod p$ as symmetric key

Diffie-Hellman

- Subject to man-in-the-middle (MiM) attack



- Trudy shares secret $g^{at} \bmod p$ with Alice
- Trudy shares secret $g^{bt} \bmod p$ with Bob
- Alice and Bob don't know Trudy exists!

Diffie-Hellman

- How to prevent MiM attack?
  - Encrypt DH exchange with symmetric key
  - Encrypt DH exchange with public key
  - Sign DH values with private key
  - Other?
- You **MUST** be aware of MiM attack on Diffie-Hellman

Elliptic Curve Cryptography

Elliptic Curve Crypto (ECC)

- "Elliptic curve" is **not** a cryptosystem
- Elliptic curves are a different way to do the math in public key system
- Elliptic curve versions of DH, RSA, etc.
- Elliptic curves may be more efficient
  - Fewer bits needed for same security
  - But the operations are more complex

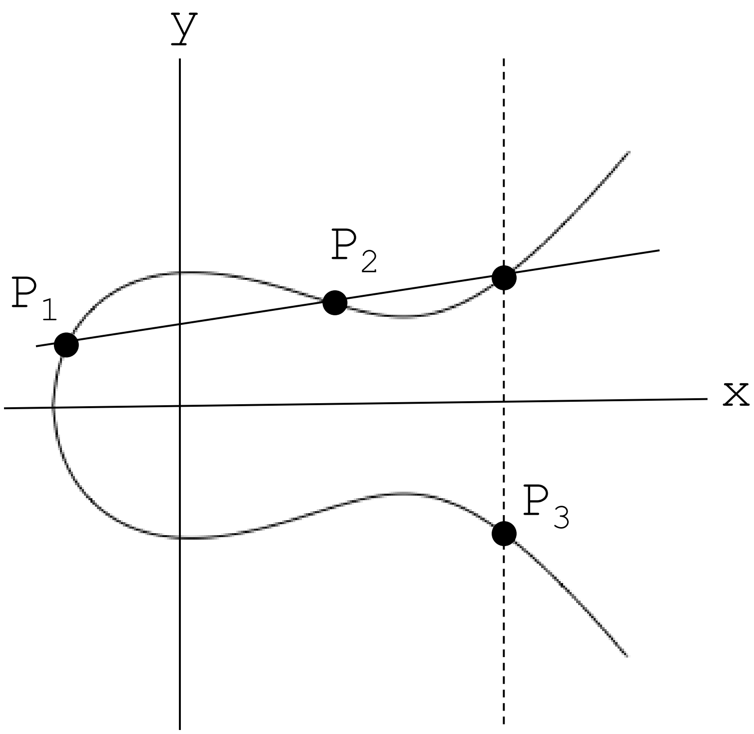
What is an Elliptic Curve?

- An elliptic curve E is the graph of an equation of the form

$$y^2 = x^3 + ax + b$$

- Also includes a "point at infinity"
- What do elliptic curves look like?
- See the next slide!

Elliptic Curve Picture



- Consider elliptic curve

$$E: y^2 = x^3 - x + 1$$

- If P_1 and P_2 are on E, we can define

$$P_3 = P_1 + P_2$$

as shown in picture

- Addition is all we need

Points on Elliptic Curve

- Consider $y^2 = x^3 + 2x + 3 \pmod 5$

  $x = 0 \Rightarrow y^2 = 3 \Rightarrow \text{no solution} \pmod 5$

  $x = 1 \Rightarrow y^2 = 6 = 1 \Rightarrow y = 1,4 \pmod 5$

  $x = 2 \Rightarrow y^2 = 15 = 0 \Rightarrow y = 0 \pmod 5$

  $x = 3 \Rightarrow y^2 = 36 = 1 \Rightarrow y = 1,4 \pmod 5$

  $x = 4 \Rightarrow y^2 = 75 = 0 \Rightarrow y = 0 \pmod 5$

- Then points on the elliptic curve are

  $(1,1)\ (1,4)\ (2,0)\ (3,1)\ (3,4)\ (4,0)$

  and the point at infinity: ∞

Elliptic Curve Math

- Addition on: $y^2 = x^3 + ax + b \pmod p$

  $P_1=(x_1,y_1), P_2=(x_2,y_2)$

  $P_1 + P_2 = P_3 = (x_3,y_3)$ where

  $x_3 = m^2 - x_1 - x_2 \pmod p$

  $y_3 = m(x_1 - x_3) - y_1 \pmod p$

  And $m = (y_2-y_1)*(x_2-x_1)^{-1} \bmod p$, if $P_1 \neq P_2$

  $m = (3x_1{}^2+a)*(2y_1)^{-1} \bmod p$, if $P_1 = P_2$

  Special cases: If m is infinite, $P_3 = \infty$, and

  $\infty + P = P$ for all P

Elliptic Curve Addition

- Consider $y^2 = x^3 + 2x + 3 \pmod 5$. Points on the curve are $(1,1)$ $(1,4)$ $(2,0)$ $(3,1)$ $(3,4)$ $(4,0)$ and ∞
- What is $(1,4) + (3,1) = P_3 = (x_3,y_3)$?

$$m = (1-4)*(3-1)^{-1} = -3*2^{-1}$$
$$= 2(3) = 6 = 1 \pmod 5$$
$$x_3 = 1 - 1 - 3 = 2 \pmod 5$$
$$y_3 = 1(1-2) - 4 = 0 \pmod 5$$

- On this curve, $(1,4) + (3,1) = (2,0)$

ECC Diffie-Hellman

- **Public:** Elliptic curve and point (x,y) on curve
- **Secret:** Alice's A and Bob's B

A(x,y) →

← B(x,y)

Alice, A

Bob, B

- Alice computes A(B(x,y))
- Bob computes B(A(x,y))
- These are the same since AB = BA

ECC Diffie-Hellman

- **Public:** Curve $y^2 = x^3 + 7x + b \pmod{37}$ and point $(2,5) \Rightarrow b = 3$
- ***Alice's secret:*** $A = 4$
- **Bob's secret:** $B = 7$
- Alice sends Bob: $4(2,5) = (7,32)$
- Bob sends Alice: $7(2,5) = (18,35)$
- Alice computes: $4(18,35) = (22,1)$
- Bob computes: $7(7,32) = (22,1)$

Uses for Public Key Crypto

Uses for Public Key Crypto

- Confidentiality
  - Transmitting data over insecure channel
  - Secure storage on insecure media
- Authentication (later)
- Digital signature provides integrity and **non-repudiation**
  - No non-repudiation with symmetric keys

Non-non-repudiation

- Alice orders 100 shares of stock from Bob
- Alice computes **MAC** using symmetric key
- Stock drops, Alice claims she did not order
- Can Bob prove that Alice placed the order?
- **No!** Since Bob also knows symmetric key, he could have forged message
- **Problem:** Bob knows Alice placed the order, but he can't prove it

Non-repudiation

- Alice orders 100 shares of stock from Bob
- Alice **signs** order with her private key
- Stock drops, Alice claims she did not order
- Can Bob prove that Alice placed the order?
- **Yes!** Only someone with Alice's private key could have signed the order
- This assumes Alice's private key is not stolen (revocation problem)

Sign and Encrypt
vs
Encrypt and Sign

Public Key Notation

- **Sign** message M with *Alice's* **private key:** $[M]_{Alice}$
- **Encrypt** message M with *Alice's* **public key:** $\{M\}_{Alice}$
- Then

$$\{[M]_{Alice}\}_{Alice} = M$$

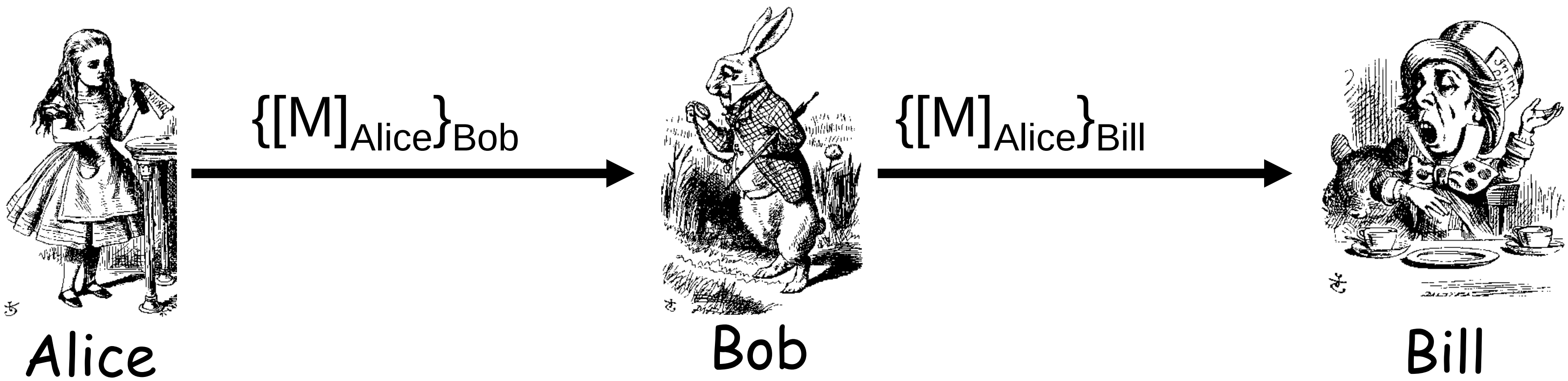
$$[\{M\}_{Alice}]_{Alice} = M$$

Confidentiality and Non-repudiation

- Suppose that we want confidentiality and non-repudiation
- Can public key crypto achieve both?
- Alice sends message to Bob
  - **Sign and encrypt** $\{[M]_{Alice}\}_{Bob}$
  - **Encrypt and sign** $[\{M\}_{Bob}]_{Alice}$
- Can the order possibly matter?

Sign and Encrypt

- M = "I love you"



- **Q:** What is the problem?
- **A:** Bill misunderstands crypto!

Encrypt and Sign

- M = "My theory, which is mine…."

Alice

$[\{M\}_{Bob}]_{Alice}$

Bill

$[\{M\}_{Bob}]_{Bill}$

Bob

- **Note** that Bill cannot decrypt M
- **Q:** What is the problem?
- **A:** Bob misunderstands crypto!

Public Key Infrastructure

Public Key Certificate

- Contains name of user and user's public key (and possibly other info)
- Certificate is **signed** by the issuer (such as VeriSign) who vouches for it
- Signature on certificate is verified using signer's public key

Certificate Authority

- Certificate authority (CA) is a trusted 3rd party (TTP) that issues and signs cert's
  - Verifying signature verifies the identity of the owner of corresponding private key
  - Verifying signature does **not** verify the identity of the source of certificate!
  - Certificates are public!
  - Big problem if CA makes a mistake (a CA once issued Microsoft certificate to someone else!)
  - Common format for certificates is X.509

PKI

- Public Key Infrastructure (PKI) consists of all pieces needed to securely use public key cryptography
  - Key generation and management
  - Certificate authorities
  - Certificate revocation (CRLs), etc.
- No general standard for PKI
- We consider a few "trust models"

PKI Trust Models

- Monopoly model
  - One universally trusted organization is the CA for the known universe
  - Favored by VeriSign (for obvious reasons)
  - Big problems if CA is ever compromised
  - Big problem if you don't trust the CA!

PKI Trust Models

- Oligarchy
  - Multiple trusted CAs
  - This approach used in browsers today
  - Browser may have 80 or more certificates, just to verify signatures!
  - User can decide which CAs to trust

PKI Trust Models

- Anarchy model
  - Everyone is a CA!
  - Users must decide which "CAs" to trust
  - This approach used in PGP (Web of trust)
  - Why do they call it "anarchy"? Suppose cert. is signed by Frank and I don't know Frank, but I do trust Bob and Bob says Alice is trustworthy and Alice vouches for Frank. Should I trust Frank?
- Many other PKI trust models

Confidentiality in the Real World

Symmetric Key vs Public Key

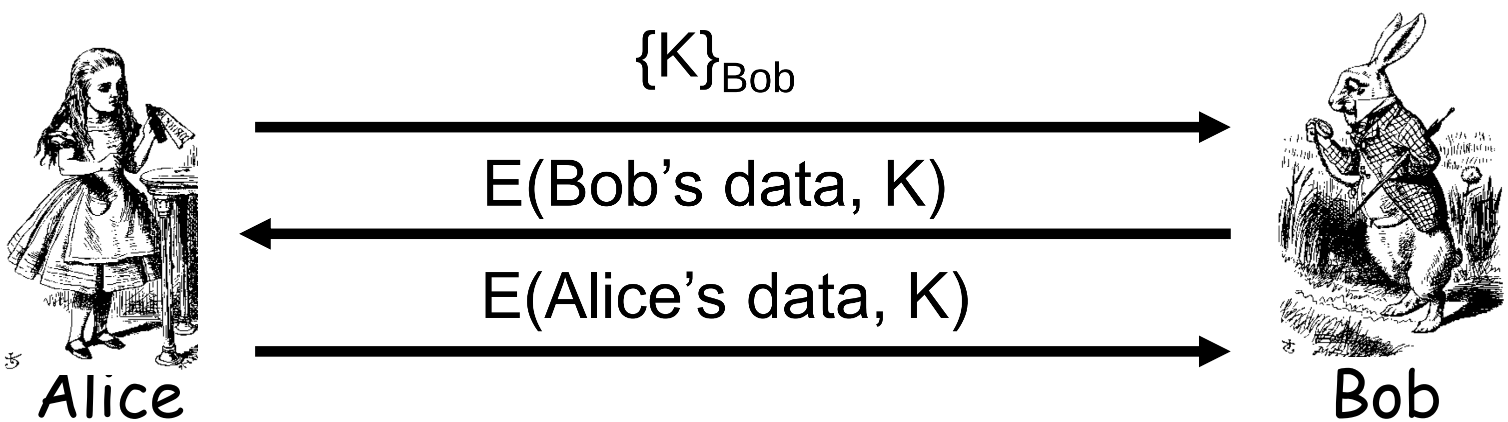
- Symmetric key +'s
  - **Speed**
  - No public key infrastructure (PKI) needed
- Public Key +'s
  - **Signatures** (non-repudiation)
  - No shared secret

Notation Reminder

- Public key notation
  - Sign message M with Alice's **private key**
    - $[M]_{Alice}$
  - Encrypt message M with Alice's **public key**
    - $\{M\}_{Alice}$
- Symmetric key notation
  - Encrypt plaintext P with symmetric key K
    - $C = E(P,K)$
  - Decrypt ciphertext C with symmetric key K
    - $P = D(C,K)$

Real World Confidentiality

- Hybrid cryptosystem
  - Public key crypto to establish a key
  - Symmetric key crypto to encrypt data
  - Consider the following



- Can Bob be sure he's talking to Alice?