Key Agreement

- We have assumed that Alice and Bob have agreed upon a key known only to themselves
- How did they do that?
- Secret key agreement: Alice and Bob agree upon shared key K, over a public channel, without any eavesdroppers learning K
- How can we achieve secret key agreement?

Diffie-Hellman (1/3)

- Two people can agree over an insecure channel on a secret key in such a way that both of them receive the same key without anyone else knowing it
- Original protocol published in 1976
- p: A prime number being 2000 to 4000 bits long

  - Prime: A number that has exactly two divisors, 1 and itself
- Taking a modulo: Just divide r by p, throw away the quotient and keep the remainder as the answer

  - Example: 25 modulo 7, you divide 25 by 7, which gives a quotient of 3 with a remainder of 4, so 25 mod 7 = 4
- We first choose a large prime p and a primitive element g which generates a finite field

Diffie-Hellman (2/3)

Alice | Bob

$$\text{Alice} \xrightarrow{g\hat{\ }x \bmod p} \text{Bob}$$

$$\text{Alice} \xleftarrow{g\hat{\ }y \bmod p} \text{Bob}$$

K = (g^y)^x | K = (g^x)^y

Diffie-Hellman (3/3)

- The attacker sees g^x and g^y but not x or y
- The problem of computing g^xy given g^x and g^y is known as the DH problem
- As long as p and g are chosen correctly there is no way to compute this efficiently
- In the finite field is called discrete logarithm and the problem of computing x from g^x in a finite group is known as the discrete logarithm problem

An Active Attack

Alice | Mallory | Bob

Alice $\xrightarrow{g^x \bmod p}$ Mallory

Mallory $\xrightarrow{g^u \bmod p}$ Bob

Mallory $\xleftarrow{g^y \bmod p}$ Bob

Alice $\xleftarrow{g^w \bmod p}$ Mallory

Alice: $K = (g^w)^x$

Mallory: $K = (g^x)^w$

Mallory: K\` $= (g^y)^u$

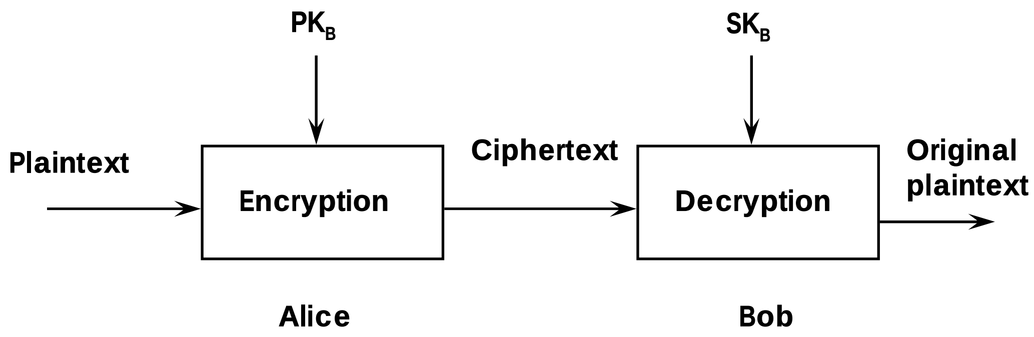
Bob: K\` $= (g^u)^y$

Asymmetric Key Cryptosystems

- In public key cryptography each person has a pair of keys
  - The *public key* and the *private key*
- Public key is published and widely distributed
  - While the private key is kept secret
- Need for exchanging secret keys is eliminated
  - All communications involve only public keys
- Examples of public key cryptographic algorithms are:
  - RSA, ElGamal, Rabin

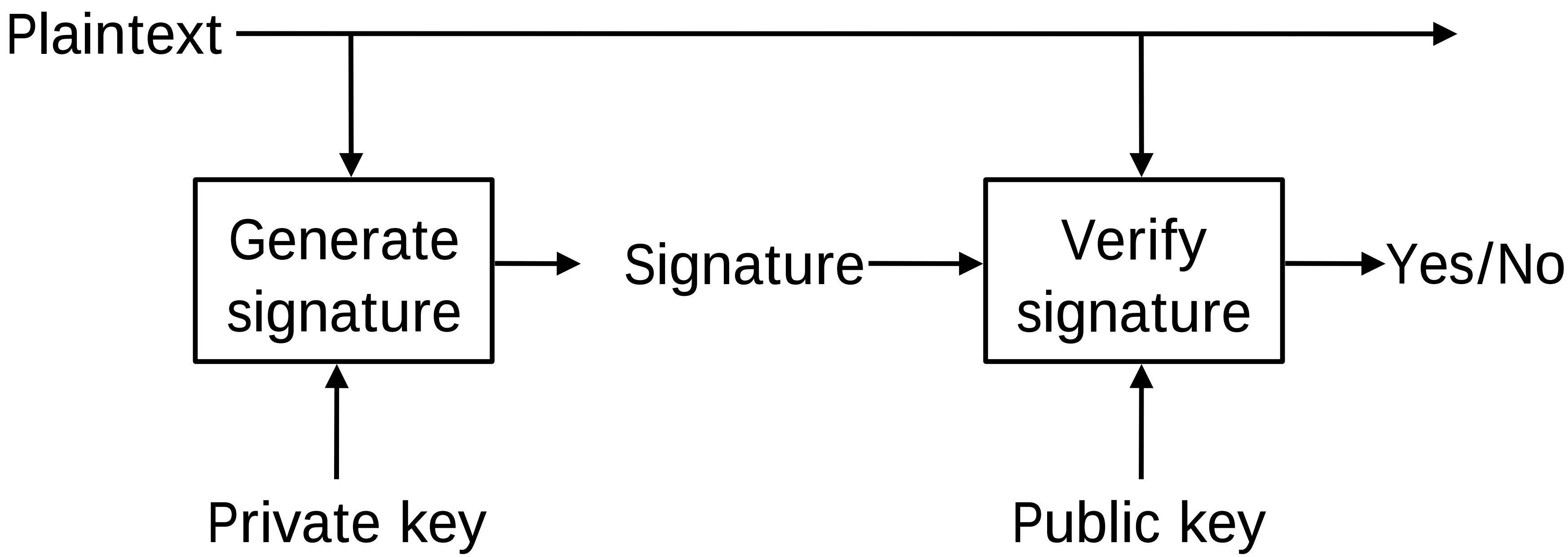
Public Key Cryptography

- Each user in a public key system creates his own private key (*SK*) and his own public key (*PK*)



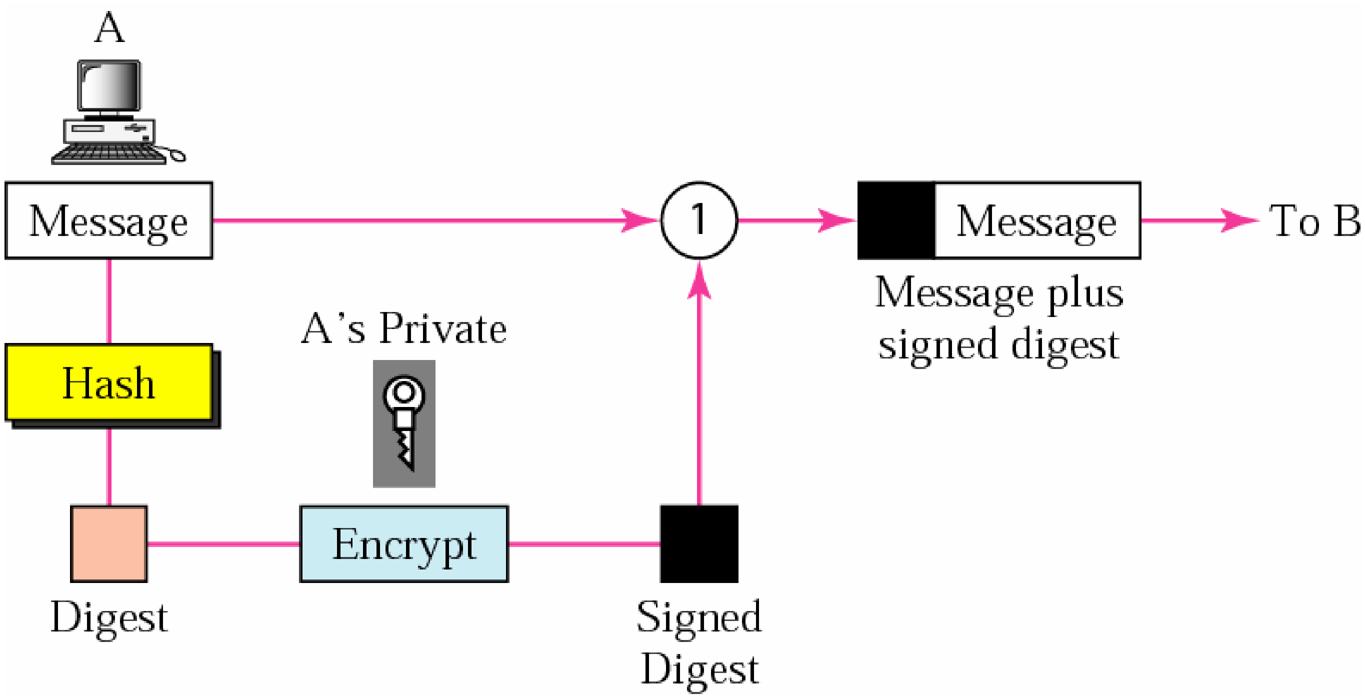
- When user Alice wants to send an encrypted message to Bob
  - She looks up his public key (PK_B) in a public directory
    - * Or obtains it by some other means

Public Key Integrity Protection



Signatures and Message Digests

- Instead of creating a digital signature on an arbitrarily large document
  - Compute a message digest on the document and then create a digital signature on the digest



Public Key Authentication

Alice (knows A's private key)

Bob (knows A's public key)

I'm Alice →

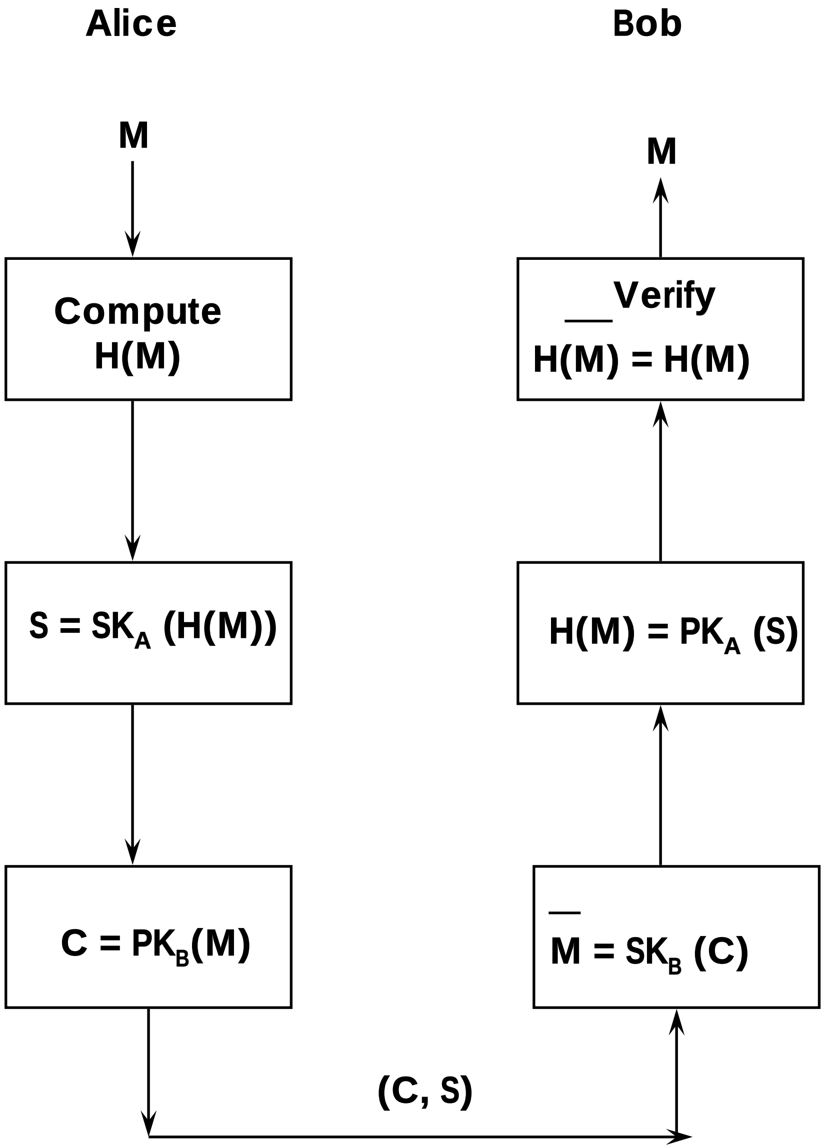
Picks random R
Encrypts R using A's public key (getting C)

← If you're Alice, decrypt C

Decrypts C

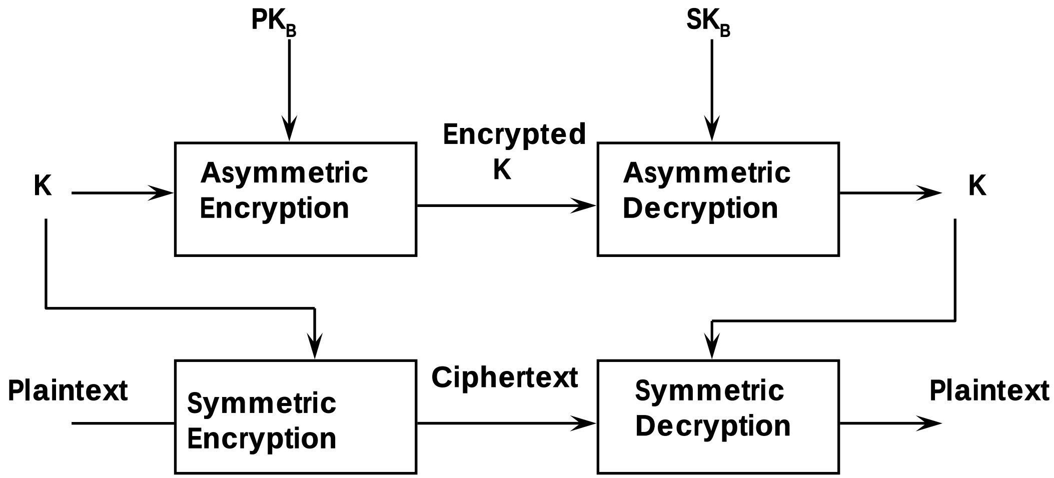
R →

Enveloped and Signed Data



Hybrid Schemes

- Public key algorithms are not a replacement for secret key algorithms such as AES
  - Rather they supplement AES or any other fast bulk encryption cipher



- The above example shows
  - How we can use a public key algorithm to securely transfer a *session key* (K) and
  - Use the session key for bulk encryption and decryption

RSA

- Named after its inventors Rivest, Shamir and Adleman who developed it in 1978 while working at MIT
- Algorithm:

- Randomly choose two different large primes p and q and compute
  $n = p * q$ (n is known as the *modulus*)
- Randomly choose an encryption key e such that e and $(p-1) * (q-1)$ are relatively prime
- To encrypt a message m the sender computes the ciphertext c as
  $c := m^e \pmod n$
- To decrypt a ciphertext c the receiver computes $c^d \pmod n$
- d is the decryption key: $e * d = 1 \pmod{(p - 1) * (q - 1)}$
- The pair (n, e) forms the public key
- The values (p, q, d) are the private key

- To sign a message m the owner of the private key computes s as
  $s := m^d \pmod n$ the pair (m, s) is now a signed message
- To verify the signature anyone who knows the public key can verify that
  $s^e = m \pmod n$

RSA Security

- The security of RSA is based on a trapdoor one-way function
- Given the public *n* and *e* it is easy to compute $m^e \pmod{n}$ from *m* but not the other way around
- However, if you know the factorization of *n* then it is easy to do the inverse computation
- The factorization of *n* is the trapdoor information
- As with encryption the security of the signature is based on the fact that the *e'th* root on *m* can only be computed by someone who knows the private key
- Just remember that computations of any roots modulo *n* require knowledge of the private key

Key Management

- Public key cryptography is based on the idea that
  - An individual will generate a key pair
  - Keep one component secret and publish the other component (public key)
- Other users on the network
  - Must be able to retrieve this public key and associate the user’s identity with it
- One way to form this association is
  - To enlist the services of a Trusted Third Party (*TTP*)

X.509 Certificates

- The TTP will construct a message referred to as a *certificate*

Subject (Identity of User)	Public Key	Validity Period	Issuer (Identity of TTP)	Other fields	Signature of TTP

- The certificate contains a number of fields

  - Identity of the user
  - Public key of the user
  - Validity period of the certificate
  - Identity of the TTP
  - Miscellaneous fields
  - A digital signature on the above fields with the secret key of the *TTP*

- It is assumed that every user in the system is equipped with the public key of the *TTP*

  - This allows one to verify the digital signature on the certificate
  - Thus guaranteeing that the public key is associated with the named user

Certification Hierarchy

- TTPs that issue certificates are referred to as Certification Authorities (*CA*s)
- The root *CA* issues certificates only to other *CA*s
- Each user of the system need only hold the public key of the root *CA*