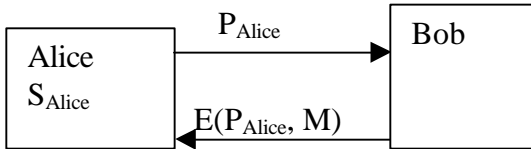


Date 02/28/06

Review

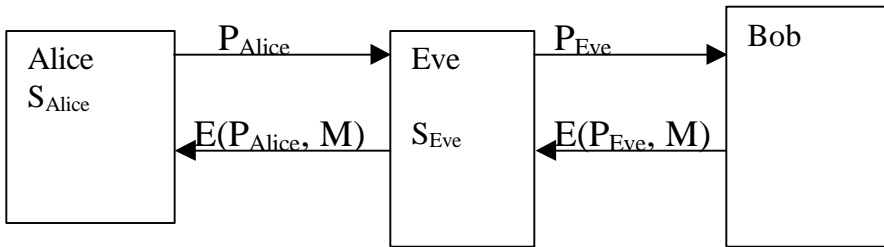
- RSA



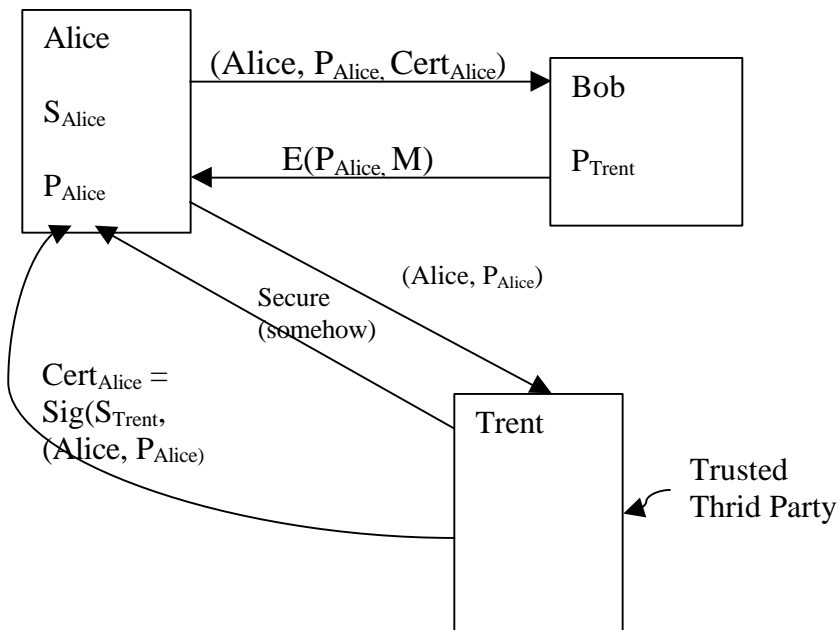
Is this secure?

Eve can impersonate Alice

Man-in-the middle attack



- In the above situation, Bob and Alice don't even know about the attack.
- Bob believes that he is communicating with Alice. Hence, we need to authenticate the sender. Hence, there is a need to bind the public keys to identities or in the other words to authenticate public keys.
- Public key infrastructure:



In the above situation, Trent somehow will verify the identity of Alice.

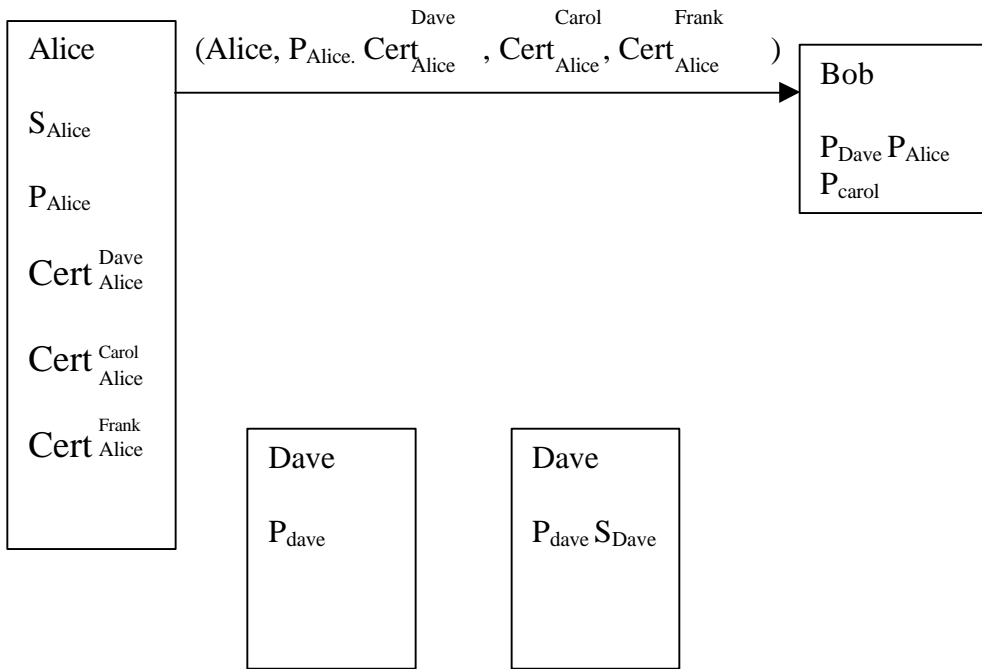
- Trent's signature means that he has verified the identity of Alice and its public key is P_{Alice} .

- Biggest criticism of this public key system is that Trent is all powerful. If he is malicious then the whole system breaks down.

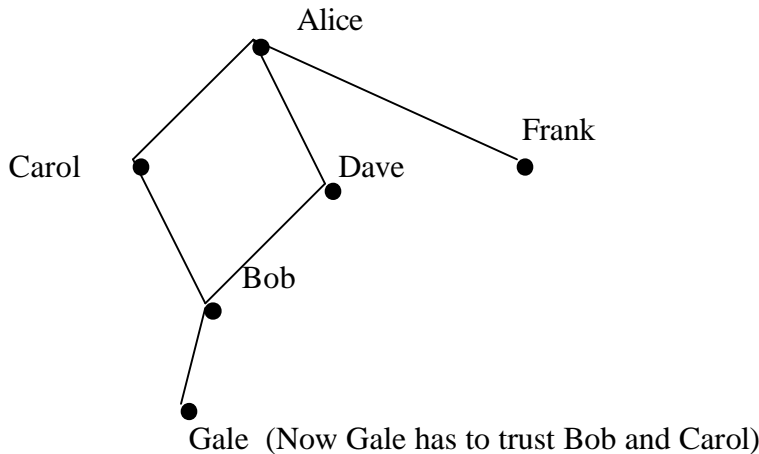
- ✓ How to solve the problem of mistaken certificate?
 - Certificate revocation list.

Normally there are multiple 3rd parties and if there is a certificate from one of the parties then that is good enough.

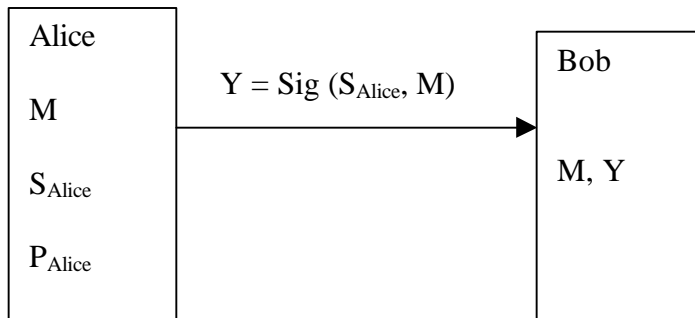
- ✓ PGP web of trust.



- ✓ Bob will check how much he believes P_{Alice} based on how much he trusts Dave and Carol to sign for public keys. If the belief is over a threshold then he gives a certificate to Alice.



Public key signature (P_{Alice} is public information)



$$\text{Ver}(P_{\text{Alice}}, M, Y) = \text{valid}$$

Security for signature

A signature scheme consists of sig & ver algorithms such that $\text{ver}(P_x, M, \text{Sig}(S_x, M)) = \text{valid}$ and without knowing S_x it is difficult to compute any pair M, Y s.t $\text{ver}(P_x, M, Y) = \text{valid}$.

RSA Signature

Pick p, q

$$N = pq$$

$$Ed = 1 \pmod{\phi(N)}$$

$$P_{\text{Alice}} = N, e$$

$$S_{\text{Alice}} = N, d$$

$$S = \text{sig}(S_{\text{Alice}}, M) = M^d \pmod{N}$$

i.e. $\text{ver}(P_{\text{Alice}}, M, S) = \text{valid}$ iff $S^e = M \pmod{N}$

This scheme will not work, consider the following case

Given, $M_1, S_1 = M_1^d \pmod{N}$
 $M_2, S_2 = M_2^d \pmod{N}$
 then $S_1 \times S_2 \pmod{N} = M_1^d M_2^d \pmod{N}$
 $= (M_1 M_2)^d \pmod{N}$ and this is the signature of message $M_1 M_2$

Hence to solve this issue, we hash the message before doing the exponentiation

To sign M , compute $h = H(M)$
 $S = \text{sig}(S_{\text{Alice}}, M) = h^d \pmod{N}$
 $\text{ver}(P_{\text{Alice}}, M, S) = \text{valid}$ iff $S^e = H(M) \pmod{N}$

As $H(M_1) \cdot H(M_2) \neq H(M_1 M_2)$ this scheme works.

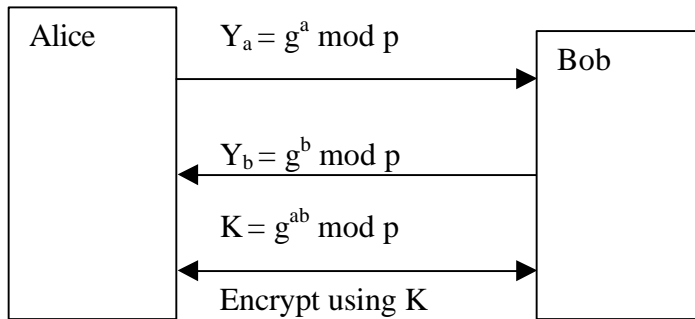
Diffie-Hellman Key Exchange

Alice – g, p (large prime number)

Bob – g, p

p is prime, $(p-1)/2$ is prime and $g^{(p-1)/2} = 1 \pmod p$

Alice picks a random number 'a' and Bob picks a random number 'b'



Problem: Given p, g, g^a computing 'a' is the computational Diffie-Hellman problem or discrete log problem and it roughly takes $O(p^{1/2})$ time.