# Lecture 27

### Diffie-Hellman Key Exchange Section 3.5

## Diffie-Hellman Key Exchange

- No third party involved
- After a common shared key, is established, it can be used to encrypt message
- A common shared key is symmetric

### The Diffie-Hellman Key Exchange

From B's view



Alice and Bob share a prime q and  $\alpha$ , such that  $\alpha < q$  and  $\alpha$  is a primitive root of q

Alice generates a private key  $X_A$  such that  $X_A < q$ 

Alice calculates a public key  $Y_A = \alpha^{X_A} \mod q$ 

Alice receives Bob's public key *YB* in plaintext

Alice calculates shared secret key  $K = (Y_B)^{X_A} \mod q$ 

#### Bob

Alice and Bob share a prime q and  $\alpha$ , such that  $\alpha < q$  and  $\alpha$  is a primitive root of q

Bob generates a private key  $X_B$  such that  $X_B < q$ 

Bob calculates a public key  $Y_B = \alpha^{X_B} \mod q$ 

Bob receives Alice's public key  $Y_A$  in plaintext

Bob calculates shared secret key  $K = (Y_A)^{X_B} \mod q$ 



## Example

- A computes B computes
- Then communication key exchange ,
- A receives . B receives
- A computes
  - B computes

### Attack

- Adversary gets ,
- She needs to compute either or
- Secure?

### Discrete Log Problem

### Two cryptographic assumptions:

- Discrete logarithm problem (discrete log problem): Given for random , it is computationally hard to find
- **Diffie-Hellman assumption**: Given and for random, , no polynomial time attacker can distinguish between a random value R and.
  - Intuition: The best known algorithm is to first calculate and then compute, but this requires solving the discrete log problem, which is hard!
- Note: Multiplying the values doesn't work, since you get