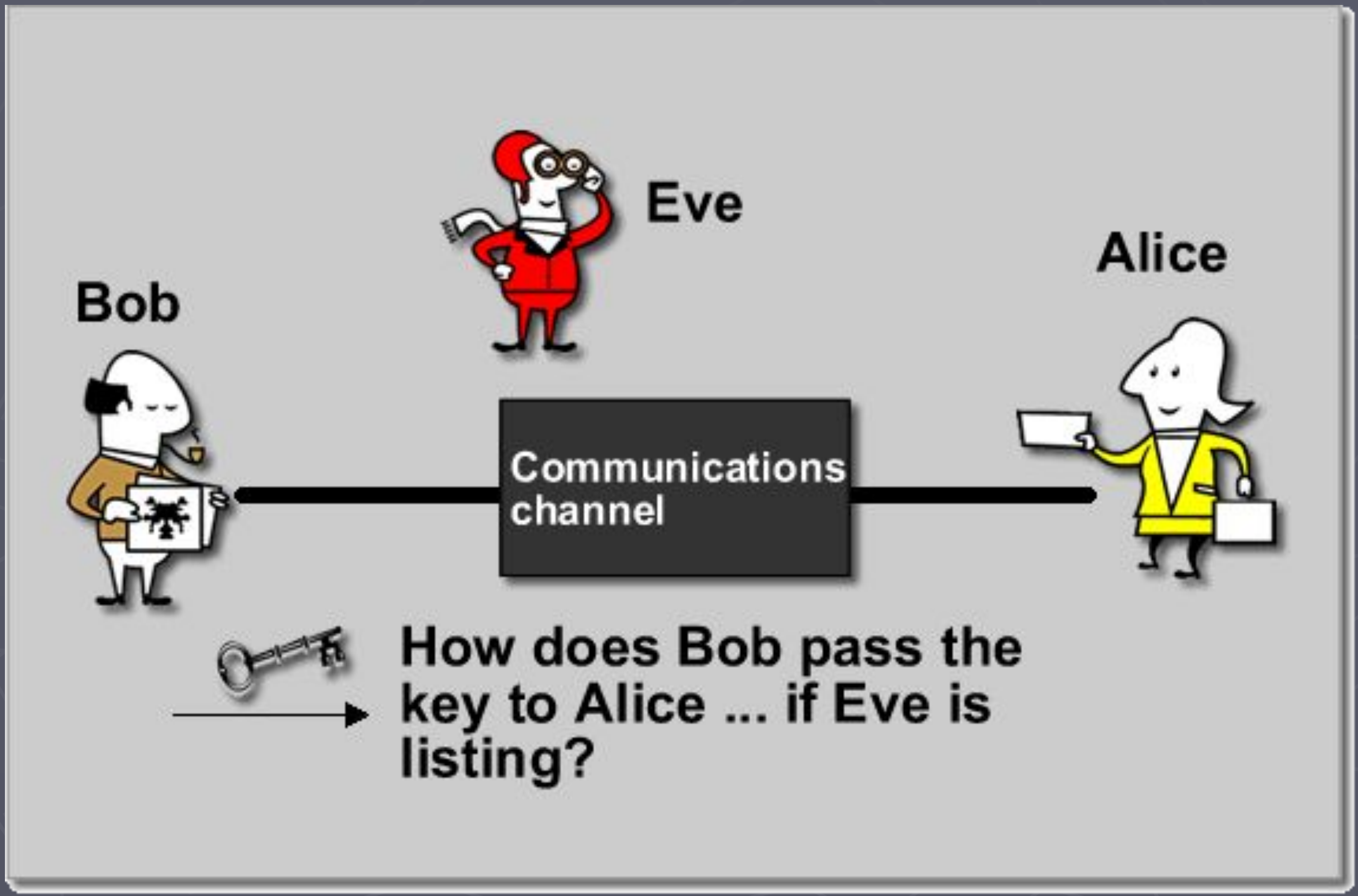
Diffie- Hellman Key Agreemaent

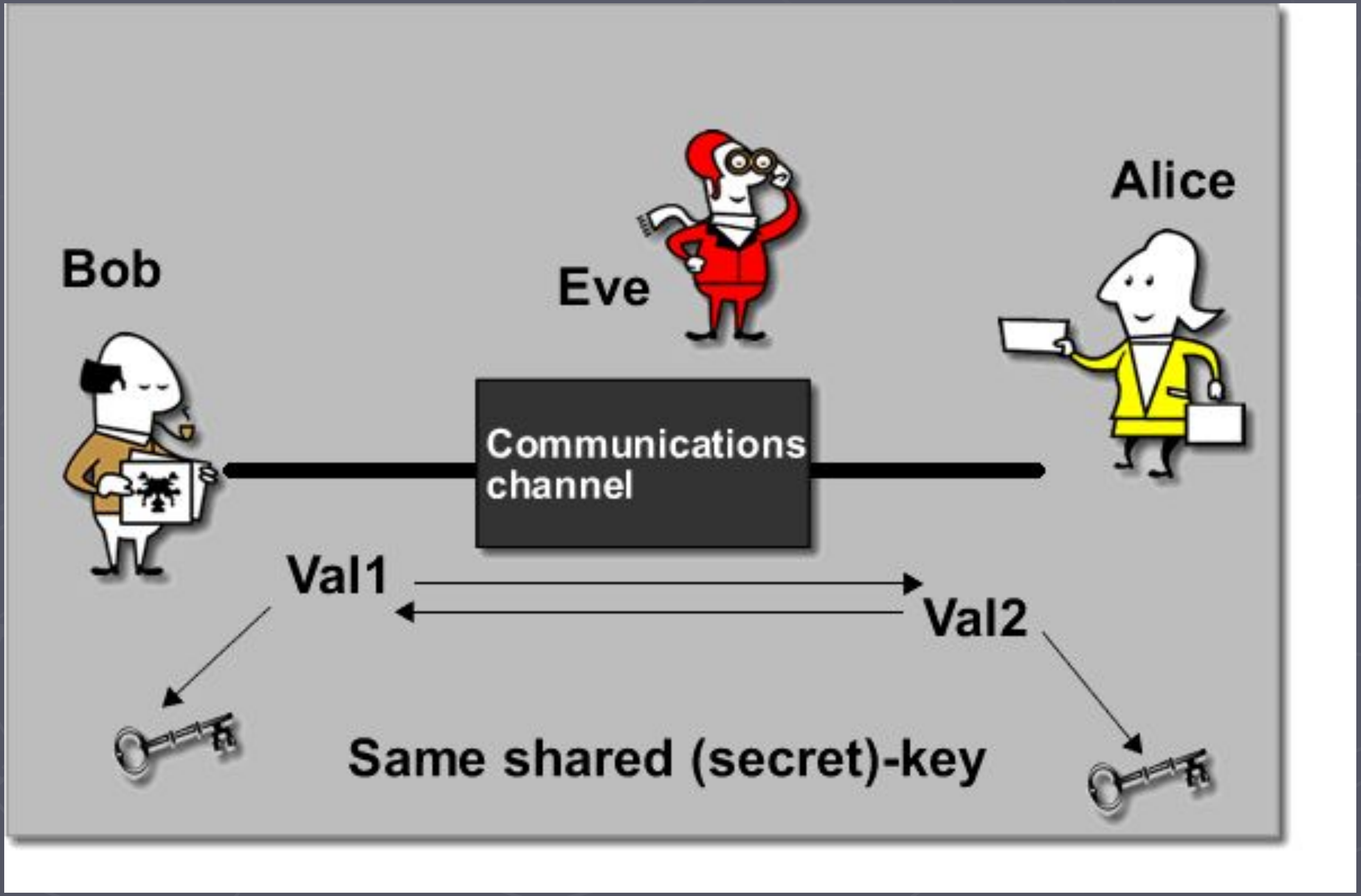
Diffie-Hellman

- Diffie-Hellman is a key exchange protocol developed by Diffie and Hellman in 1976.
- The purpose of Diffie-Hellman is to allow two entities to exchange a secret over a public (insecure) medium without having any prior secrets.

Key Establishment: The problem (cont.)



Diffie-Hellman Key Exchange



Diffie-Hellman Key Exchange

- Suppose we have two people wishing to communicate: Alice and Bob.
- They do not want Eve (eavesdropper) to know their message.

Algorithm

- Requires two large numbers, one prime p, and generator g is a primitive root of mod p, (p and g are both publicly available numbers).

Note: Anyone has access to these numbers.

- Users pick random private values x (x < p) and y (y < p)
- Compute public values
  - $R1 = g^x \bmod p$
  - $R2 = g^y \bmod p$
- Public values R1 and R2 are exchanged
- Compute shared, private key
  - $k_{alice} = (R2)^x \bmod p$
  - $k_{bob} = (R1)^y \bmod p$
- Algebraically it can be shown that $k_{alice} = k_{bob}$
  - Users now have a symmetric secret key to encrypt

Proof

- We know

  $R1 = g^x \bmod p$

  $R2 = g^y \bmod p$

- $k_{alice} = (R2)^x \bmod p$

  $= (g^y \bmod p)^x \bmod p$

  $= (g^y)^x \bmod p$

  $= (g)^{yx} \bmod p$

  $= (g^x)^y \bmod p$

  $= (g^x \bmod p)^y \bmod p$

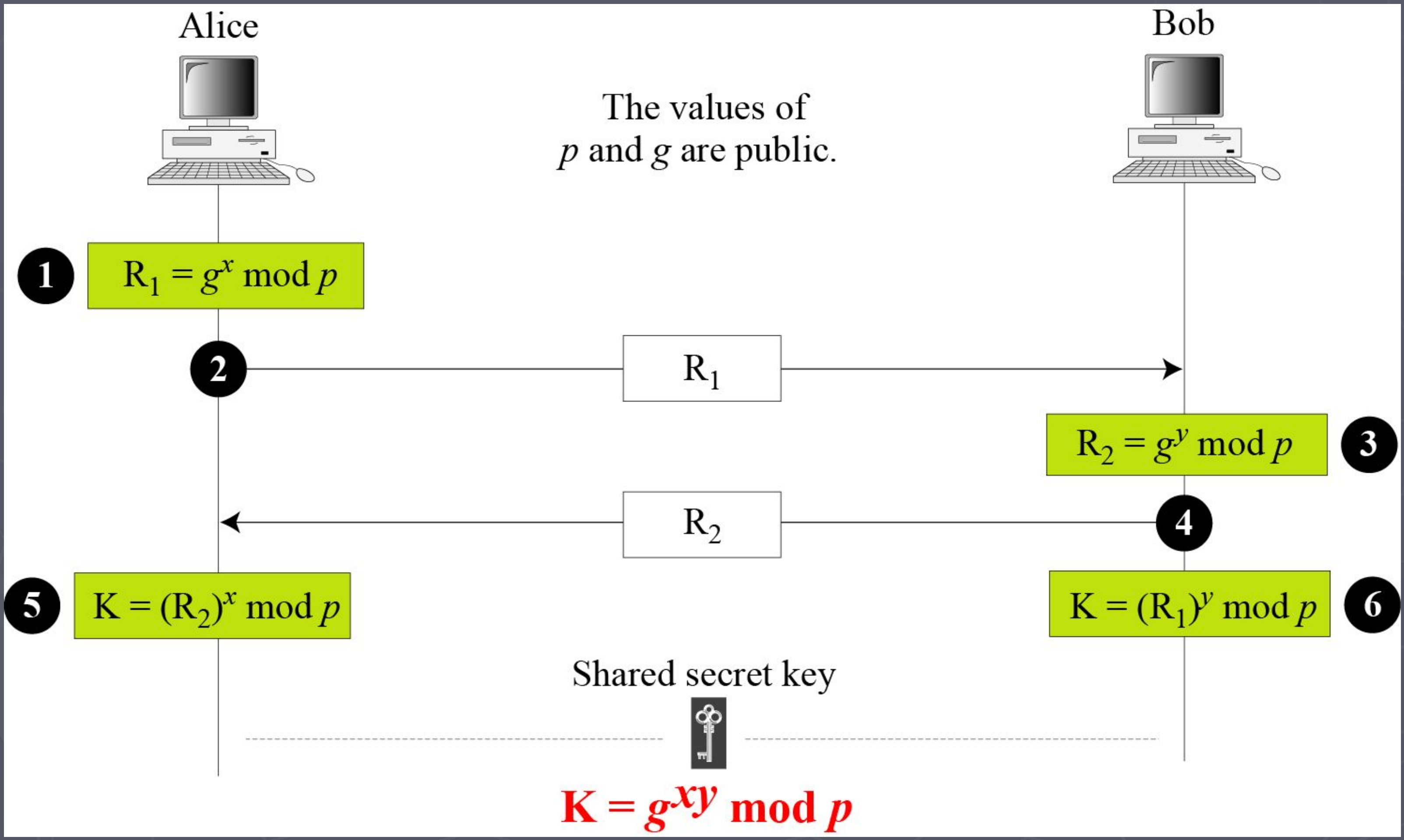
  $= (R1)^y \bmod p$

  $= k_{bob}$

Diffie-Hellman Key Exchange

- If Eve wants to compute *k, then she would* need either *a or b.*
- Otherwise, Eve would need to solve a Discrete Logarithm Problem.
  - There is no known algorithm to achieve this in a reasonable amount of time.

Diffie-Hellman Key Exchange



Example

- Alice and Bob get public numbers
  - $P = 23, \; G = 9$
- Alice and Bob pick private values x=4 & y=3 respectively
- Alice and Bob compute public values
  - $R1 = 9^4 \bmod 23 = 6561 \bmod 23 = 6$
  - $R2 = 9^3 \bmod 23 = 729 \bmod 23 = 16$
- Alice and Bob exchange public numbers
- Alice and Bob compute symmetric keys
  - $k_{alice} = (R2)^x \bmod p = 16^4 \bmod 23 = 9$
  - $k_{bob} = (R1)^y \bmod p = 6^3 \bmod 23 = 9$
- Alice and Bob now can talk securely!

Example

- Alice and Bob get public numbers
  - P = 17, G = 2
- Alice and Bob pick private values x=3 & y=7 respectively
- Alice and Bob compute public values
  - R1 = 2^3 mod 17 = 8 mod 17 = 8
  - R2 = 2^7 mod 17 = 128 mod 17 = 9
- Alice and Bob exchange public numbers
- Alice and Bob compute symmetric keys
  - k_{alice} = $(R2)^x$ mod p = 9^3 mod 17 = 15
  - k_{bob} = $(R1)^y$ mod p = 8^7 mod 17 = 15
- Alice and Bob now can talk securely!

Example in Two Steps

$p = 17, g = 2, x = 3, y = 7$

$(2^3)^7 \bmod 17 = (2^7)3 \bmod 17$

$2^{21} \bmod 17 = 2^{21} \bmod 17$

Alice				Bob		
Secret	Public	Calculates	Sends	Calculates	Public	Secret
a	p, g		p,g$\longrightarrow$			b
a	p, g, A	g^a mod p = A	A$\longrightarrow$		p, g	b
a	p, g, A		$\longleftarrow$B	g^b mod p = B	p, g, A, B	b
a, **s**	p, g, A, B	B^a mod p = s		A^b mod p = s	p, g, A, B	b, **s**

1. Alice and Bob agree to use a prime number p=23 and base g=5.
2. Alice chooses a secret integer ***a***=**6**, then sends Bob A = g^a mod p
   - A = 5^6 mod 23
   - A = **15,625** mod 23
   - A = 8
3. Bob chooses a secret integer ***b***=**15**, then sends Alice B = g^b mod p
   - B = 5^{15} mod 23
   - B = **30,517,578,125** mod 23
   - B = 19
4. Alice computes **s** = B^{a} mod p
   - **s** = 19^6 mod 23
   - **s** = **47,045,881** mod 23
   - **s** = **2**
5. Bob computes **s** = A^{b} mod p
   - **s** = 8^{15} mod 23
   - **s** = **35,184,372,088,832** mod 23
   - **s** = **2**
6. Alice and Bob now share a secret: **s** = **2**. This is because 6*15 is the same as 15*6. So somebody who had known both these private integers might also have calculated s as follows:
   - **s** = 5^{6*15} mod 23
   - **s** = 5^{15*6} mod 23
   - **s** = 5^{90} mod 23
   - **s** = **807,793,566,946,316,088,741,610,050,849,573,099,185,363,389,551,639,556,884,765,625** mod 23
   - **s** = **2**

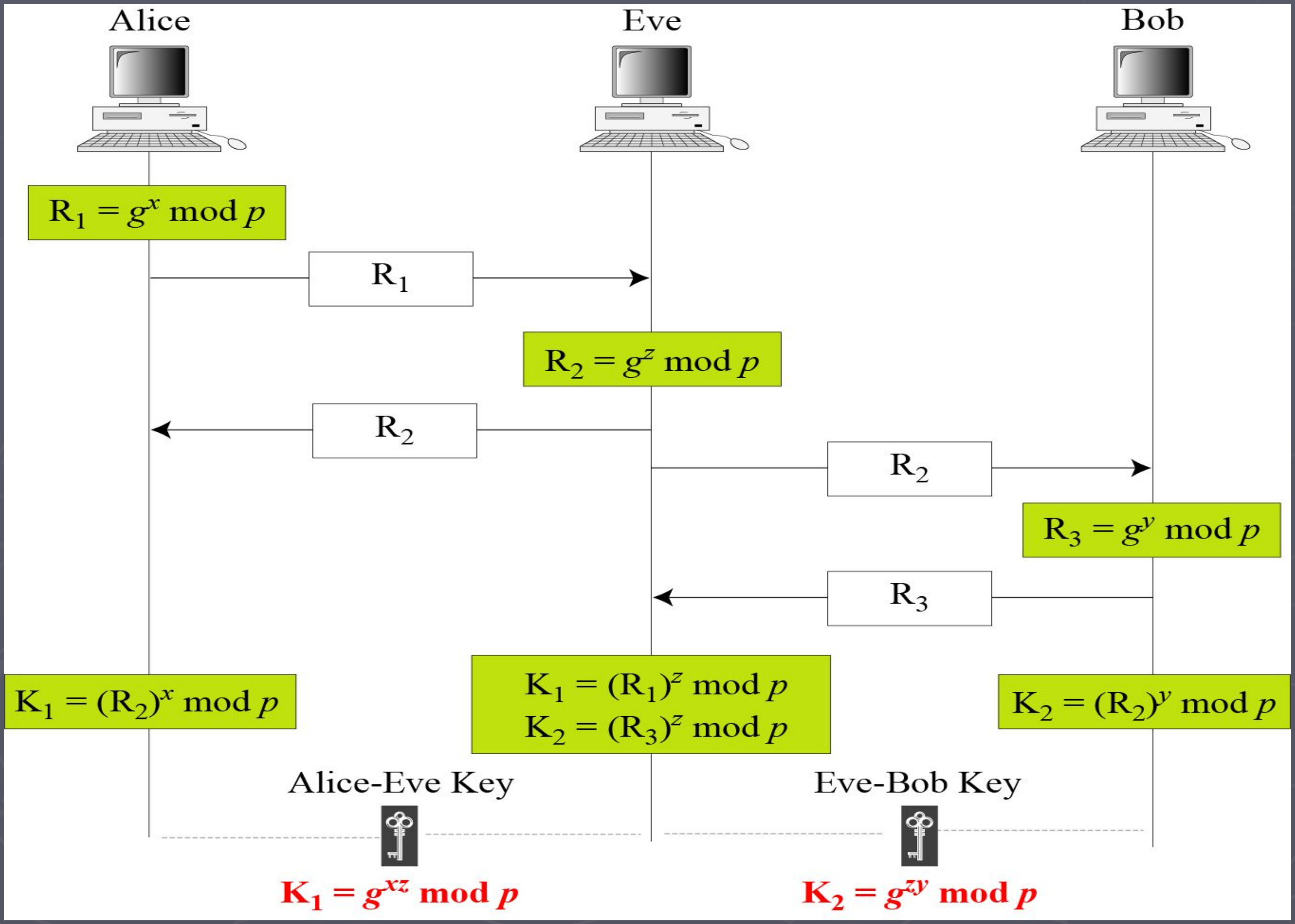
Security of Diffie-Hellamn

- This protocol vulnerable to two attacks:
  - The Man-in-the-middle attack
  - The Discrete logarithmic attack

Man-in-the-middle attack

- (p and g are publicly known)
- An adversary Eve intercepts Alice's public value and sends her own public value to Bob.
- When Bob transmits his public value, Eve substitutes it with her own and sends it to Alice.
- Eve and Alice thus agree on one shared key and Eve and Bob agree on another shared key.
- After this exchange, Eve simply decrypts any messages sent out by Alice or Bob, and then reads and possibly modifies them before re-encrypting with the appropriate key and transmitting them to the other party.
- This is present because Diffie-Hellman key exchange does not authenticate the participants.

Man-in-the-middle attack (cont.)



Solution to Man-in-the-middle attack

- The basic idea is as follows.
  - Prior to execution of the protocol, the two parties Alice and Bob each obtain a public/private key pair and a certificate for the public key.
  - During the protocol, Alice calculates a signature on certain messages, covering the public value g^a mod p. Bob proceeds in a similar way. Even though Eve is still able to intercept messages between Alice and Bob,
  - She cannot forge signatures without Alice's private key and Bob's private key. Hence, the enhanced protocol defeats the man-in-the-middle attack.

Discrete Logarithmic Attack

- The security of the key exchange is based on the difficulty of the discrete logarithm problem.
- Eve can intercept R1 and R2.
- If she can find x from R1 = g^x mod p and y from R2 = g^y mod p,
- Then she calculate the symmetric key K= g^{xy} mod p.
- The secret key is not secret anymore.

Discrete Logarithmic Attack (cont.)

- To make Diffie-Hellman safe from the discrete logarithm attack, the following are recommended.
  - The prime p must be very large. Then it is computationally infeasible to calculate the shared secret key $k = (g^{xy} \bmod p)$ given the two public values $(g^x \bmod p)$ and $(g^y \bmod p)$.
  - Bob and Alice must destroy x and y after they have calculated the symmetric key. The values of x and y must be used only once.

Summary

- Key agreement protocol- is a specific method of exchanging cryptographic keysKey agreement protocol- is a specific method of exchanging cryptographic keys. It is one of the earliest practical examples of key exchangeKey agreement protocol- is a specific method of exchanging cryptographic keys. It is one of the earliest practical examples of key exchange implemented within the field of cryptography.
- The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secretThe Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communicationsThe Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly