Asymmetric Cryptography

Part 2

Pr. Noureddine Chikouche
noureddine.chikouche@univ-msila.dz
https://sites.google.com/view/chikouchenoureddine

Course outline

- Key exchange
- RSA Key exchange protocol
- Diffie-Hellman Protocol
- Comparison

Key exchange

- Internet protocols make extensive use of session keys.
- A session key is a symmetric **temporary** key used to protect the transmitted data by the symmetric encryption during a communication session.

Key exchange

- To exchange session keys, two methods can be used:
  - Secure channel: conversation, VPN, ...
  - Unsecured (public) channel: Internet, Wi-Fi, mobile phones, …
- Public key algorithms (like RSA) are not used for data encryption, but rather for the exchange of keys session for symmetric cryptography.

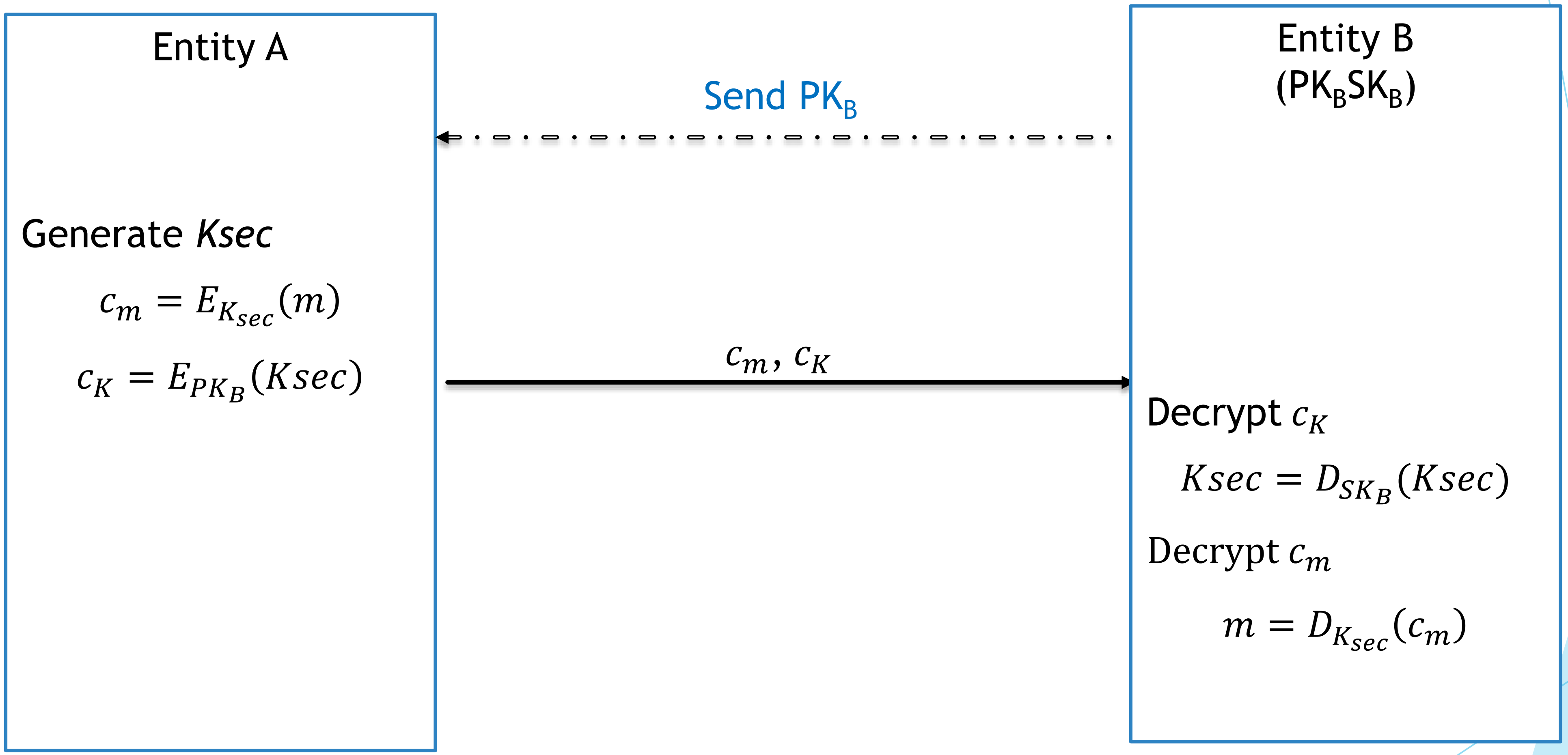
Key exchange

- For exchange keys,we use the key exchange protocols.
- There are two categories of key exchange protocols:
    - Use knowledge previously shared between communicating entities. For example, RSA key exchange protocol.
    - Do not use knowledge previously shared between the communicating entities.For example, Diffie-Hellman key exchange protocol.

RSA key exchange Protocol

- The information previously exchanged between the entities is the recipient's public key.
- This protocol is used in HTTPS and PGP protocols.
- We use the principle of hybrid encryption:
  - Encrypt the session key using the recipient's public key.
  - Encrypt the message to be sent using the session key (symmetric encryption).

RSA Key Exchange Protocol



Entity A

Generate *Ksec*

$$c_m = E_{K_{sec}}(m)$$

$$c_K = E_{PK_B}(Ksec)$$

Send PK$_B$

c_m, c_K

Entity B
(PK$_B$SK$_B$)

Decrypt c_K

$$Ksec = D_{SK_B}(Ksec)$$

Decrypt c_m

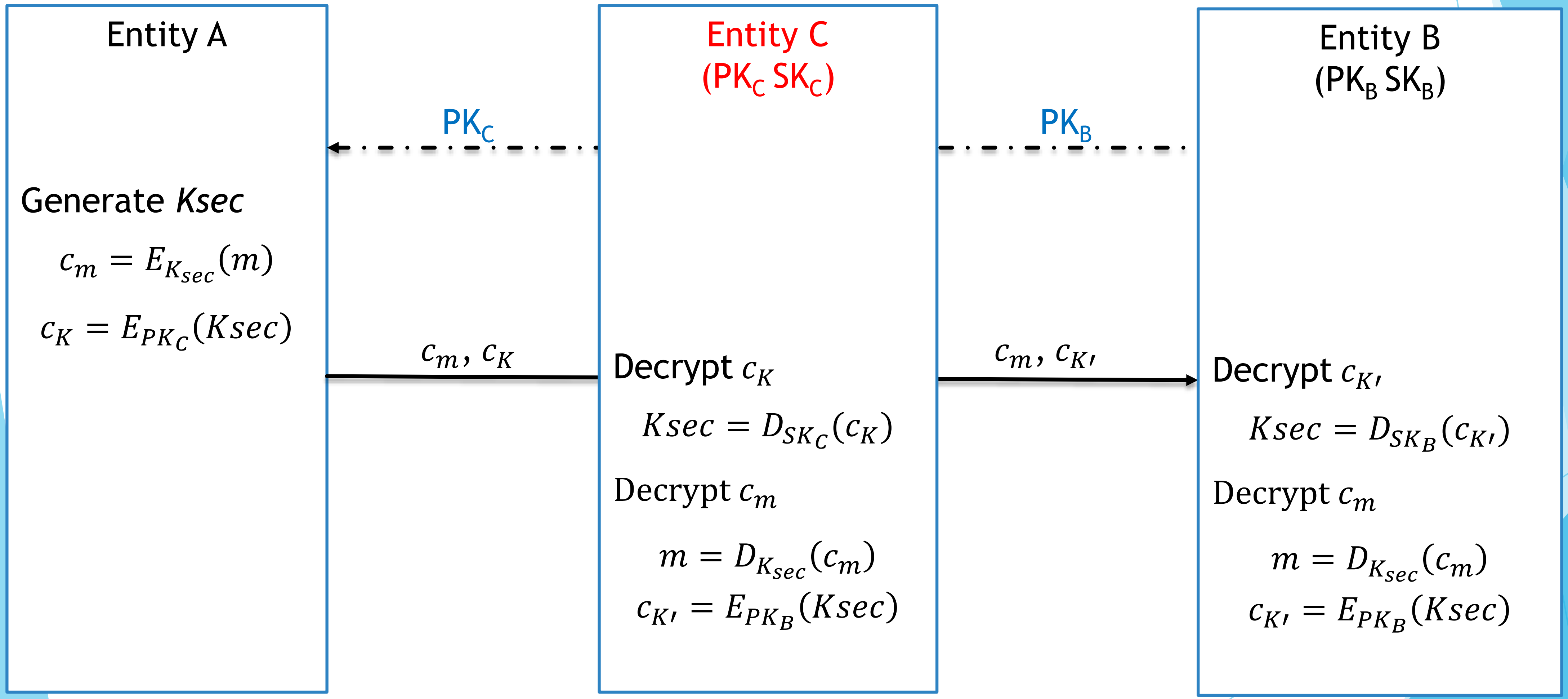
$$m = D_{K_{sec}}(c_m)$$

RSA key exchange Protocol

- Entity A randomly chooses a session key *Ksec*(*ex*.of size 128 bits for AES-128).
- A encrypts the message to be sent using the key $Ksec$ using the symmetric encryption function, $c_m = E_{K_{sec}}(m)$
- A encrypts the session key using the recipient's public key (PK_B), $c_K = E_{PK_B}(Ksec)$
- A sends c_m and c_K to entity B.
- B decrypts c_K using their private key to obtain the session key $Ksec = D_{SK_B}(Ksec)$
- B decrypts the encrypted message using the received session key $m = D_{K_{sec}}(c_m)$.

RSA Key Exchange Protocol

Attack



Entity A

Generate *Ksec*

$$c_m = E_{K_{sec}}(m)$$

$$c_K = E_{PK_C}(Ksec)$$

Entity C
(PK_C SK_C)

Decrypt c_K

$$Ksec = D_{SK_C}(c_K)$$

Decrypt c_m

$$m = D_{K_{sec}}(c_m)$$

$$c_{K'} = E_{PK_B}(Ksec)$$

Entity B
(PK_B SK_B)

Decrypt $c_{K'}$

$$Ksec = D_{SK_B}(c_{K'})$$

Decrypt c_m

$$m = D_{K_{sec}}(c_m)$$

$$c_{K'} = E_{PK_B}(Ksec)$$

RSA key exchange Protocol
Attack

- The RSA key exchange protocol is vulnerable to active attacks (*man-in-the-middle*).
- Principle of the attack:
  - During the exchange of the public key of *B* the attacker *C* replaces PK_B by its public key (PK_C) and sends it to entity *A*.
  - The entity *A* uses the public key of *C* to encrypt the session key. Then, the attacker can simply obtain the session key during the communication between A and B.
- To avoid this vulnerability, it is necessary to certify the recipient's public key.

Diffie-Hellman Protocol

Diffie & Hellman

- First public key scheme proposed in1976.

Diffie-Hellman Protocol

- This protocol is more commonly used for exchanging a session key without needing the previously shared information.
- It is often used in commercial products (SSL,...).
- Not authenticated in the basic version.
- The security of the DH protocol relies on the problem of discrete logarithm and on the problem of DH.

Diffie-Hellman Protocol

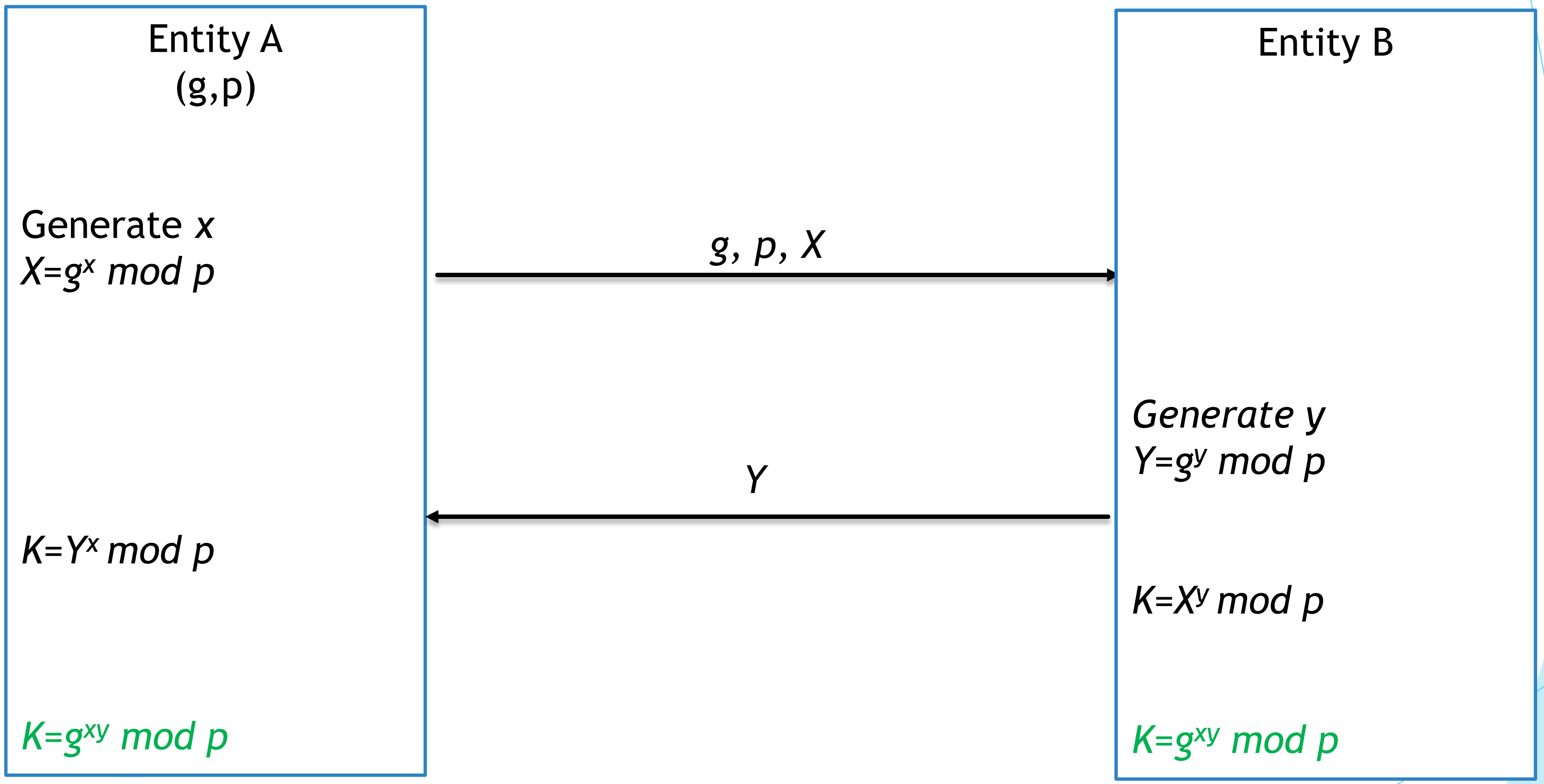
discrete logarithm

- Either *G* a cyclic group. Let *p* a prime number and *g*a generator of $\mathbb{Z}_p^*$.
- Given *x* calculate $y=g^x$ mod *p*. This is easily done by rapid exponentiation.
- Problem: Find $x=\log_g(y)$ mod *p*-1. Knowing *y*, *g and p*.
- **Applications:** El-Gamal cryptosystem and Diffie-Hellman protocol.

Diffie-Hellman Protocol
Basic version

- The problem ofDH means the difficulty of calculating $k = g^{xy} \bmod p$ from $X = g^x \bmod p$ and $Y = g^y \bmod p$, when p is big prime number.
- x and y are private keys.
- X and Y are public keys.

Diffie-Hellman Protocol
Basic version



Entity A
(g,p)

Generate x
$X=g^x \bmod p$

g, p, X

Entity B

Generate y
$Y=g^y \bmod p$

Y

$K=Y^x \bmod p$

$K=X^y \bmod p$

$K=g^{xy} \bmod p$

$K=g^{xy} \bmod p$

Diffie-Hellman Protocol
Basic version

- Either p a big prime number and g is a generator of Z_p^*.
- A selects an integer x and calculate $X=g^x$ mod p.
- A Sends X, g, and p to B.
- B selects an integer y and calculate $Y=g^y$ mod p.
- B sends Y to A.

Diffie-Hellman Protocol
Basic version

- Entity A calculates $(g^y \text{ mod } p)^x \text{ mod } p = g^{xy} \text{ mod } p$
- Entity B calculates $(g^x \text{ mod } p)^y \text{ modp} = g^{xy} \text{ mod } p$
- The shared secret key is:

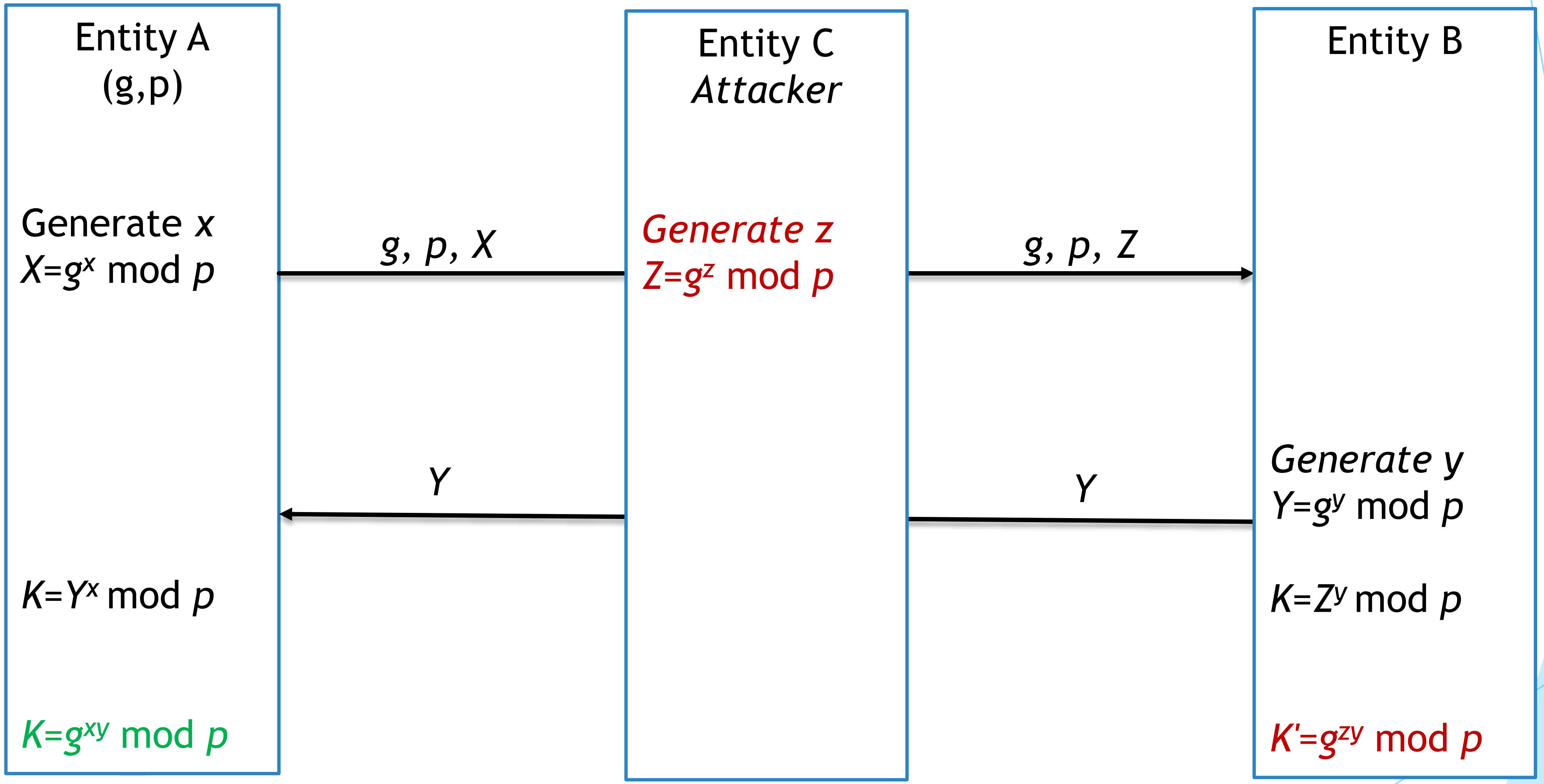
$$K = g^{xy} \text{ modp}$$

Diffie-Hellman Protocol
Basic version

- **Example:**
- We assume that entities A and B share $p = 233$ and $g = 45$:
- A selects = 11and Bob= 20, so:
  - $X = 45^{11} \bmod 233 = 147$,
  - $Y = 45^{20} \bmod 233 = 195$,
- $Y^{x} \bmod p = 195^{11} \bmod 233 = 169$
- $X^{y} \bmod p = 147^{20} \bmod 233 = 169$.
- Entities A and B share a secret key, $k = 169$.

Diffie-Hellman Protocol

Attack 1



Entity A
(g,p)

Generate x
$X=g^x \bmod p$

$K=Y^x \bmod p$

$K=g^{xy} \bmod p$

g, p, X

Y

Entity C
Attacker

Generate z
$Z=g^z \bmod p$

g, p, Z

Y

Entity B

Generate y
$Y=g^y \bmod p$

$K=Z^y \bmod p$

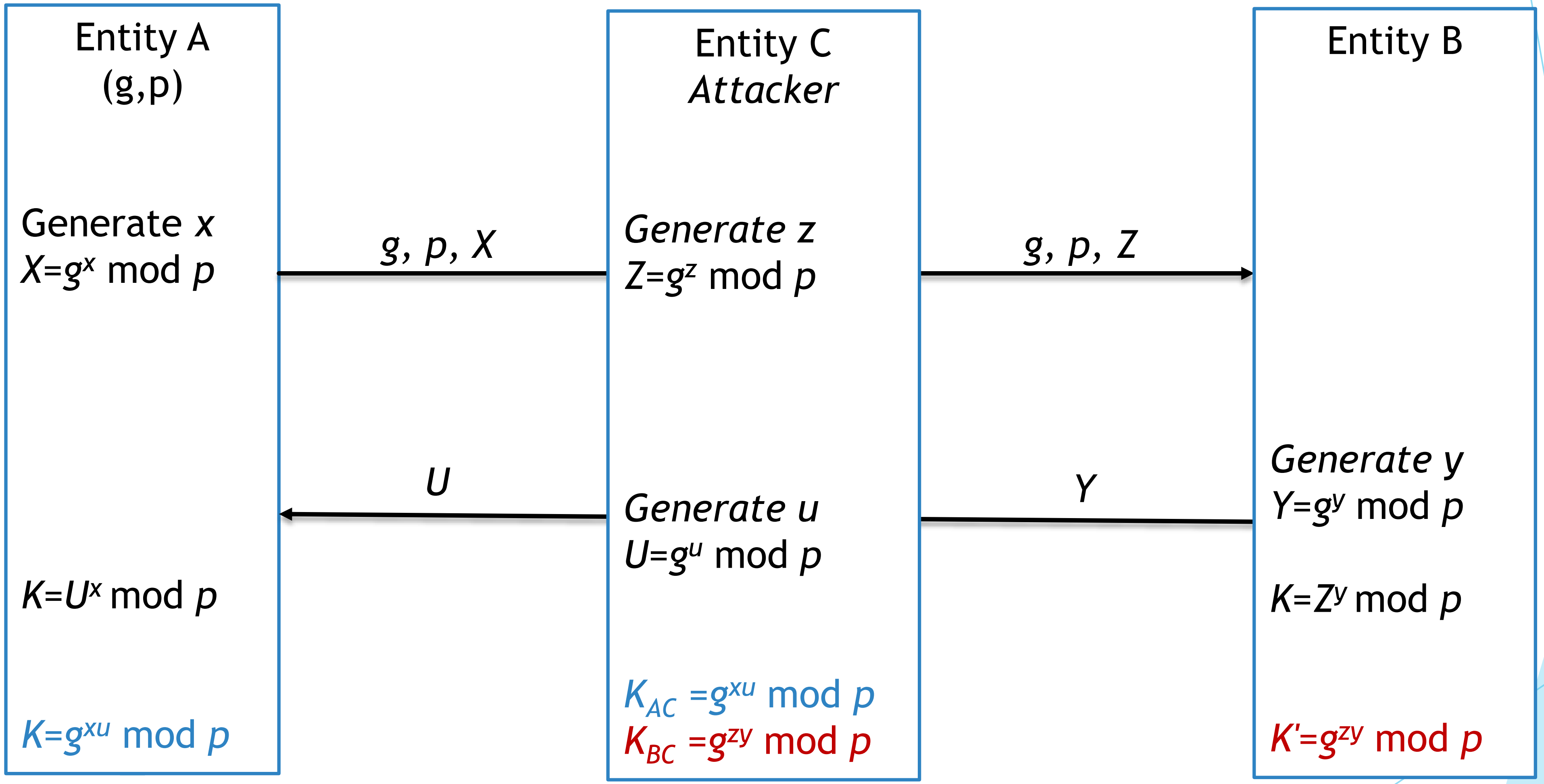
$K'=g^{zy} \bmod p$

Diffie-Hellman Protocol
Attack 1

- The DH protocol is vulnerable to activ eattacks (*man-in-the-middle*).
- During the exchange of keys,the attacker *C* blocks message X sent by A, selects a random number *z* and calculate $Z=g^z$ mod *p*. C sends message Z to entity B.
- Result: the two entities A and B calculate two different keys, $k_{HAS}=g^{xy}$ mod *p* for entity A and $k_B = g^{yz}$ mod *p* for entity B.
- When entity A sends an encrypted message *m* using its secret key, entity B cannot obtain the clear message from A.

Diffie-Hellman Protocol

Attack 2



Entity A
(g,p)

Generate *x*
$X=g^x \bmod p$

$K=U^x \bmod p$

$K=g^{xu} \bmod p$

$g,\ p,\ X$

U

Entity C
Attacker

Generate z
$Z=g^z \bmod p$

Generate u
$U=g^u \bmod p$

$K_{AC} = g^{xu} \bmod p$
$K_{BC} = g^{zy} \bmod p$

$g,\ p,\ Z$

Y

Entity B

Generate y
$Y=g^y \bmod p$

$K=Z^y \bmod p$

$K'=g^{zy} \bmod p$

Diffie-Hellman Protocol

Attack 2

- The DH protocolis vulnerable to attacks active (*man-in-the-middle*):
- During the exchange of keys,the attacker *C* modifies sent messages *X* and *Y and replaces them with new messages Z and U*, respectively.
- Entities A and B create different session keys, K and K'.
- The attacker creates keys K_{AC} andK_{BC} such as: K_{AC} =K and K_{BC}=K'.

Diffie-Hellman Protocol

Attack 2

- When *A* sends an encrypted message $E_K(m)$ to *B*, the attacker intercepts the message sent, then decrypts it using the same key K_{AC}.
- However, Entity B cannot obtain the clear message because of *K* and *K*‘ are different.
- The attacker *C* can apply the same scenario with B.

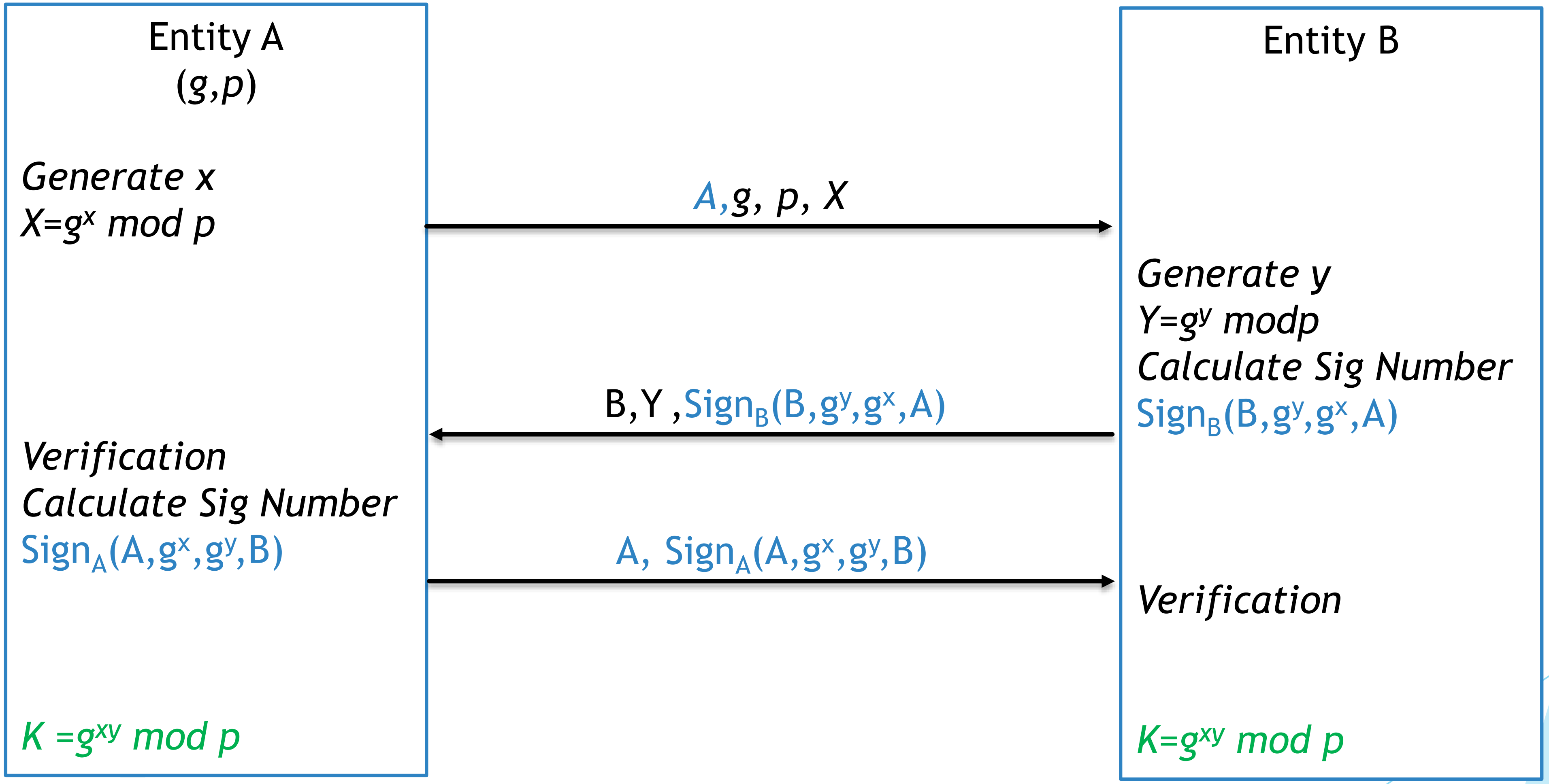
Diffie-Hellman Protocol Advantages & Disadvantages

- Benefits:
  - The discrete logarithm problem is difficult.
- Disadvantages:
  - An attacker can observe: p, g, $g^b \bmod p$ and $g^a \bmod p$
  - Like RSA, very slow.
  - Absence of authentication in the basic version

Diffie-Hellman Protocol
Signed version

- To avoid the attack through the middle, the solution is to use digital signature.
- It consists of the fact that:
  - Entity A sends (A, g^x),
  - B sends (B, g^y,$\text{Sign}_B(B,g^y,g^x,A)$)
  - A verifies the received signature and calculate the secrest key K.
  - Finally, A calculates his signature(A,$\text{Sign}_A(A,g^x,g^y,B)$) and sends it to B.
  - B checks the signature received and calculates the secret key K.

Diffie-Hellman Protocol
Signed version



Entity A
(g,p)

Generate x
$X=g^x \bmod p$

A, g, p, X

Entity B

Generate y
$Y=g^y \bmod p$
Calculate Sig Number
$\mathrm{Sign}_B(B,g^y,g^x,A)$

$B, Y, \mathrm{Sign}_B(B,g^y,g^x,A)$

Verification
Calculate Sig Number
$\mathrm{Sign}_A(A,g^x,g^y,B)$

$A, \mathrm{Sign}_A(A,g^x,g^y,B)$

Verification

$K=g^{xy} \bmod p$

$K=g^{xy} \bmod p$

Comparison

Symmetric encryption vs Asymmetric encryption

	Symmetric	Asymmetric
Benefits	• It is faster in execution and requires less computing power. • It uses small keys (64 – 256) bits. • It is easily implemented on the hardware.	• Allow messages to be signed. • Facilitate key distribution. • It requires 2*n* keys only for *n* entities communicators.
Disadvantages	• It does not provide certain services (e.g. signature). • Key distribution. • It requires *n(n-1)*/2 keys for *n* entities communicators.	• He is slow to execute (1000 times slower). • Key size greatly increased.