Lecture 12

KEY MANAGEMENT OF SYMMETRIC CRYPTOGRAPHY

R. Rhouma

UTAS
Sultanate of Oman
September 2022

CSSY2201 : Introduction to Cryptography

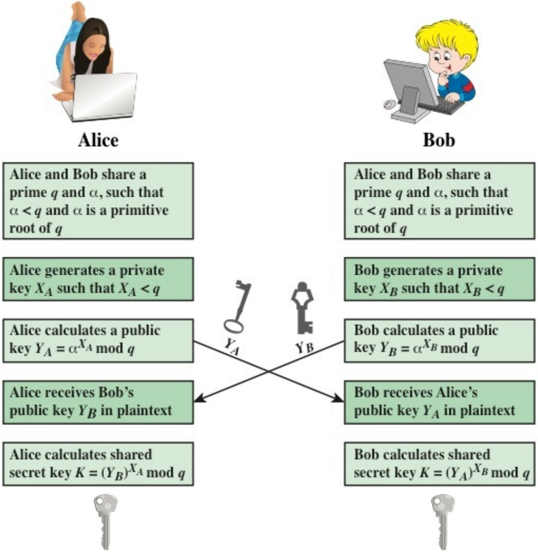
Plan

1. Diffie Hellman Key Sharing
2. Distribution of secret keys in symmetric algs

Diffie Hellman key swapping

- First public key alg
- a very large number of commercial products use this protocol
- Purpose : to allow two users to securely exchange a key which can then be used for symmetric message encryption
- Its efficiency is linked to the difficulty of calculating discrete logarithms

Diffie-Hellman Key Exchange



Diffie-Hellman Key Exchange

- The shared key between two entities A and B, is K_{AB}

$$K_{AB} = a^{x_A \times x_B} \, mod \, q$$

$$= y_A^{x_B} \, mod \, q \quad (\, calculation \, by \, B \,)$$

$$= y_B^{x_A} \, mod \, q \quad (\, calculation \, by \, A \,)$$

- K_{AB} is used as session key in a symmetric alg between A and B
- if Alice and Bob continue to communicate, they will have the same key as before, unless they choose new public keys
- an adversary must solve the discrete logarithm problem to compromise this algorithm (hard)

جامعة التقنية والعلوم التطبيقية
University of Technology and Applied Sciences
Salalah صلالة

Example

- Alice and Bob want to share a key
- They share a prime number q=353 and a number a=3
- choose random numbers secretly : $X_A = 97$, $X_B = 233$
- calculate the respective public keys :

$$y_A = 3^{97} \, mod \, 353 = 40 \;\; (Alice)$$

$$y_B = 3^{233} \, mod \, 353 = 248 \;\; (Bob)$$

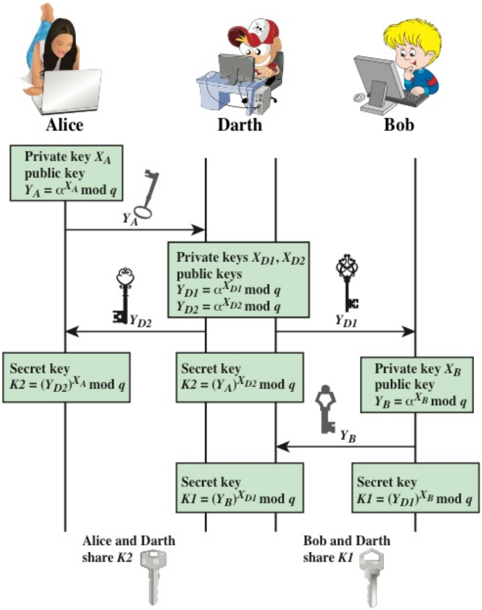
- calculate shared session key K_{AB} :

$$K_{AB} = y_B^{xA} \, mod \, 353 = 248^{97} = 160 \;\; (Alice)$$

$$K_{AB} = y_A^{xB} \, mod \, 353 = 40^{233} = 160 \;\; (Bob)$$

Man-in-the-Middle Attack

1. Darth prepares by creating 2 private/public keys
2. Alice sends her public key to Bob
3. Darth intercepts this key and passes his first public key to Bob. Darth then calculates the shared key K_2 with Alice.
4. Bob receives the public key and calculates the shared key K_1 (with Darth instead of doing it with Alice !)
5. Bob sends his public key to Alice
6. Darth intercepts this message and transmits his second public key to Alice. Darth then calculates the shared key K_1 with Bob.
7. Alice receives the key and calculates the shared key K_2 with Darth (instead of Bob)
8. Darth can then intercept, decrypt, re-encrypt, transmit all messages between Bob and Alice.


Alice
Darth
Bob
Private key X_A public key $Y_A = \alpha^{X_A} \bmod q$
Y_A
Private keys X_{D1}, X_{D2} public keys $Y_{D1} = \alpha^{X_{D1}} \bmod q$ $Y_{D2} = \alpha^{X_{D2}} \bmod q$
Y_{D2}
Y_{D1}
Secret key $K2 = (Y_{D2})^{X_A} \bmod q$
Secret key $K2 = (Y_A)^{X_{D2}} \bmod q$
Private key X_B public key $Y_B = \alpha^{X_B} \bmod q$
Y_B
Secret key $K1 = (Y_B)^{X_{D1}} \bmod q$
Secret key $K1 = (Y_{D1})^{X_B} \bmod q$
Alice and Darth share $K2$
Bob and Darth share $K1$

Key distribution and management issue

- The problem of key generation and distribution is a major problem in secure communications.
- The security of encryption protocols and algs is based on this fundamental problem
- The management of the different keys of the different entities is also a major problem
- Symmetric algs require that both interlocutors share the same secret key
- asymmetric algs require interlocutors to have valid public keys of their correspondents

Distribution of keys

Alice and Bob have many alternatives for distributing a key

- Alice can select a key and physically deliver it to Bob (hand to hand)
- A third party can select and deliver the key to Alice and Bob
- If Alice and bob have communicated before, they can use the old key to encrypt the new one
- If Alice and Bob have secure lines with third party Charlie, then Charlie can relay the key between Alice and Bob.

Key hierarchy

Generally two types of keys

- session key
  - temporary key
  - used to encrypt data between two interlocutors
  - used for a single session then discarded
- main key :
  - used to encrypt session keys
  - shared by users and a key distribution center (KDC)

Authentication Protocols

- used to convince entities of their identities and to exchange session keys
- Can be one way or mutual (both ways)
- Authentication protocols ensure :
  - privacy : to protect session keys
  - Timeliness : to prevent replay attacks

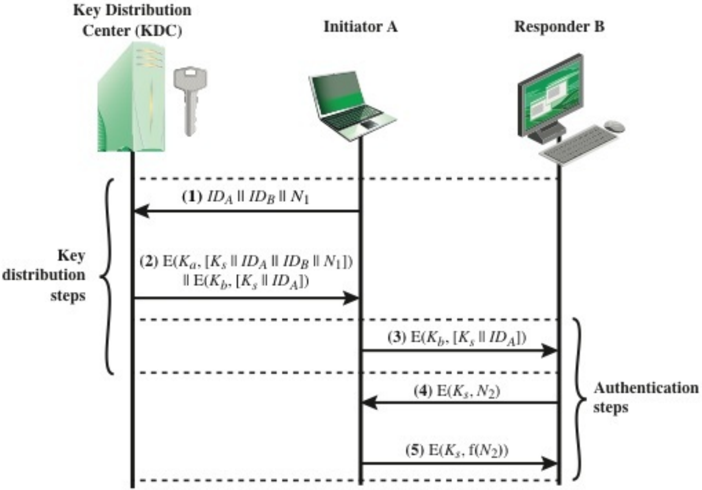
One-way authentication

- this type of authentication is required when sender and receiver are not in communication at the same time (ex : e-mail)
- the header of this type of protocol must be clear (unencrypted) to be delivered without problem by an email system
- email body content can be encrypted
- issuer must be authenticated

Authentication by symmetric cryptography

- can be done through a Key Distribution Center (KDC)
- each entity shares its master key with the KDC
- the KDC generates the session keys used for the connections between the different entities
- master keys are used to distribute session keys

Key Distribution Scenario : Needham-Schroeder



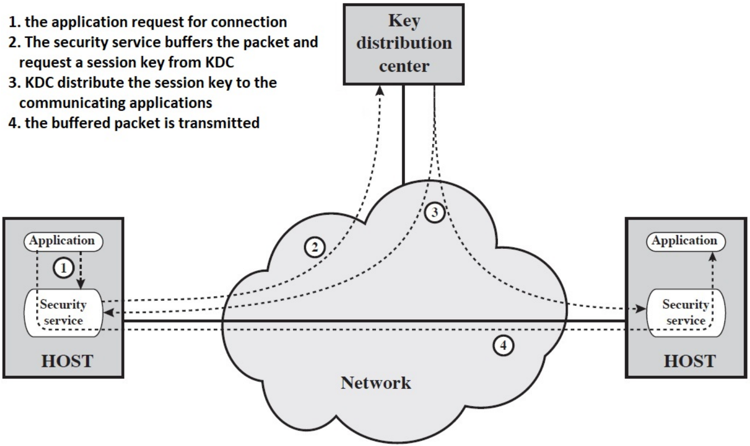
Needham-Schroeder Protocol

- third party key distribution
- for a session between two entities A and B orchestrated by a KDC
- the protocol is as follows :
  1. A -> KDC : $ID_A || ID_B || N_1$
  2. KDC -> A : $E(K_a, [K_s || ID_B || N_1 || E(K_b, [K_s || ID_A])])$
  3. A -> B : $E(K_b, [K_s || ID_A])$
  4. B -> A : $E(K_s, [N_2])$
  5. A -> B : $E(K_s, [f(N_2)])$

replay attack on Needham-Schroeder

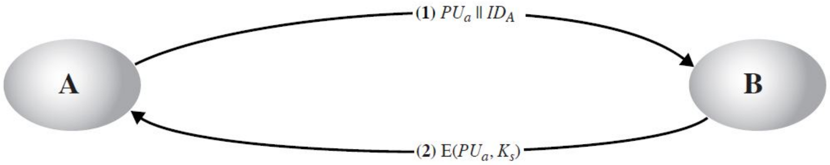
- The protocol is vulnerable to a replay-attack : the message from step 3 can be retransmitted convincing B that it is in communication with A
- solution to solve this problem :
- add timestamps in step 2 and 3
- add an external single-use random number for each key exchange K_s

Automatic key distribution in a connection-oriented protocol

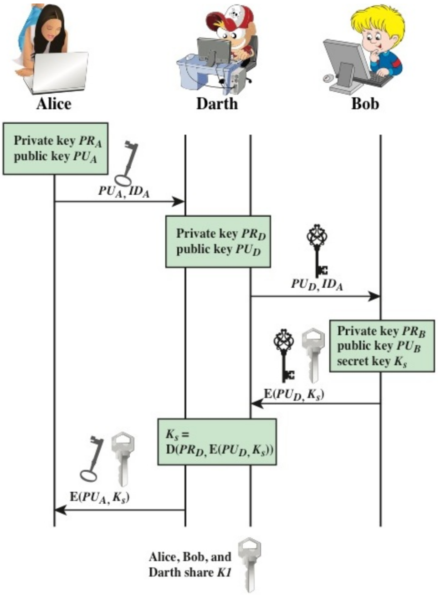


Distributing a key using an asymmetric alg : a simple key distribution

Merkle proposed the following protocol for distributing a key :



Man-in-the-middle-attack on Merkle's protocol



Distributing a Key Using an Asymmetric Alg : Privacy and Authentication