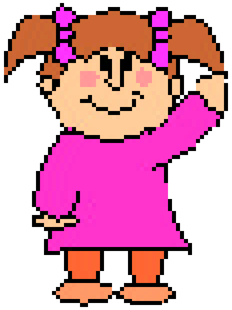
Introduction to Cryptography CS 355

Lecture 34

Key Establishment Protocols

Need for Key Establishment

$Encrypt_K(M)$



$C = Encrypt_K(M)$

$M = Decrypt_K(C)$

- Alice and Bob share a secret key K
- **How to establish the shared key?**
- **How to refresh it (not a good idea to encrypt a lot of data with the same key)**

Key Transport vs. Key Agreement

- **Key establishment**: process to establish a shared secret key available to two or more parties;
  - key transport: one party creates, and securely transfers it to the other(s).
  - key agreement: key establishment technique in which a shared secret is derived by two (or more) parties

Key Pre-distribution vs. Dynamic Key Establishment

- **Key establishment**
  - **Key pre-distribution**: established keys are completely determined a priori by initial keying material
    - generally in the form of key agreement
  - **Dynamic shared key establishment**: protocols that keys established between a fixed group of users varies in different sessions
    - also known as session key establishment
    - could be key transport or key agreement

Long-Term Key vs. Session Key

- **Session key**: temporary key, used for a short time period.
- **Long-term key**: used for a long term period, sometimes public and secret key pairs used to sign messages.
- Using session keys to:
  - limit available cipher-text encrypted with the same key
  - limit exposure in the event of key compromise
  - avoid long-term storage of a large number of distinct secret keys
  - create independence across communications sessions or applications

Basic Key Transport Protocol

- Assumes a long term symmetric key K shared between A and B
- Basic: new key is r_A

$$A \rightarrow B: E_K(r_{A,})$$

- Prevents replay: new key is r_A

$$A \rightarrow B: E_K(r_A, t_A, B)$$

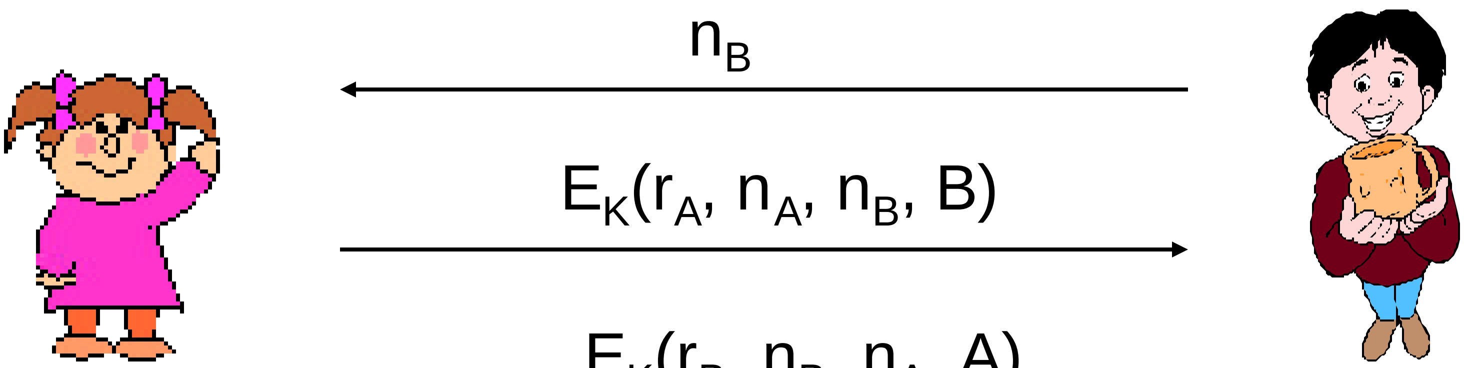
- Key transport with challenge/response:

$$A \leftarrow B: n_B$$

$$A \rightarrow B: E_K(r_A, n_B, B)$$

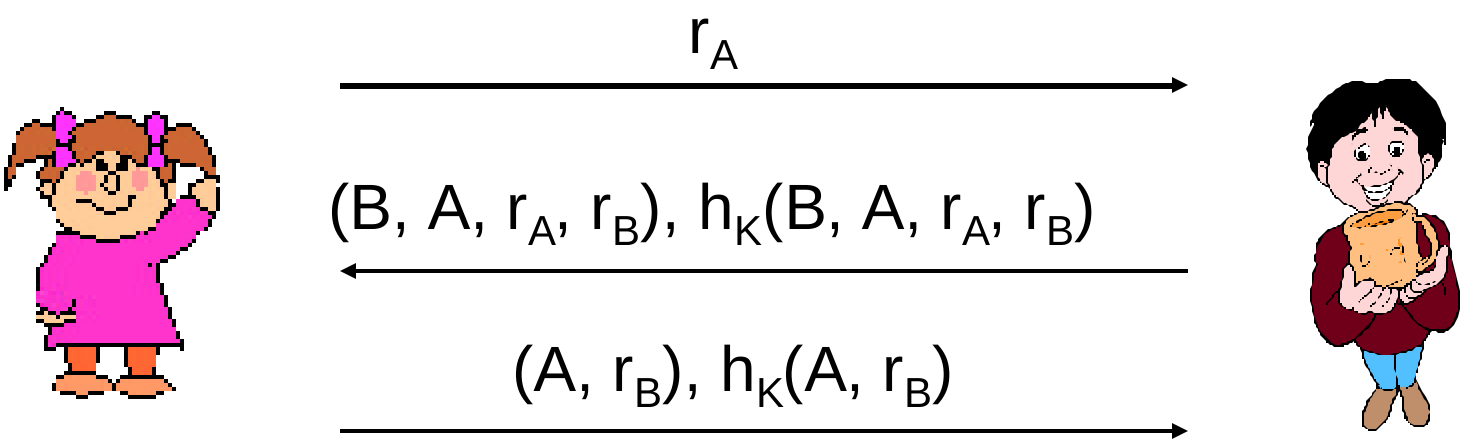
Basic Key Transport Protocol (cont.)

- Provides mutual authentication and key authentication
- Jointly control the key
- Does not provide perfect forward secrecy

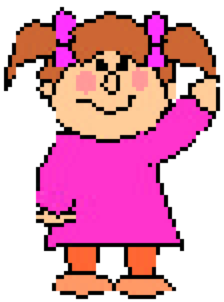


Authenticated Key Exchange Protocol 2 (AKEP2)

- Setup: A and B share long-term keys K and K'
- h_K is a MAC (keyed hash function)
- $h'_{K'}$ is a pseudo-random permutation (a block cipher)
- establish key $W = h'_{K'}(r_B)$



Shamir's No Key Algorithm



$K^a \bmod p$ →

$(K^a)^b \bmod p$ ←

$(K^{ab})^{a^{-1}} \bmod p$ →

- Setup: p is public, key K is transmitted over a public channel without authentication

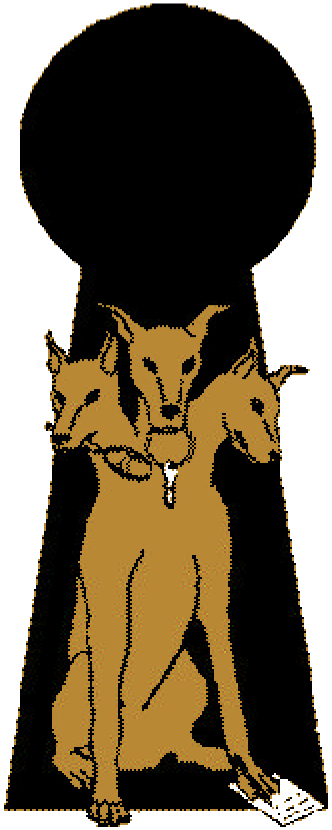
Needham-Schroeder Shared-Key Protocol

- Parties: A, B, and trusted server T
- Setup: A and T share K_{AT}, B and T share K_{BT}
- Goal: Mutual entity authentication between A and B; explicit key authentication
- Messages:

$$A \rightarrow T: \quad A, B, N_A \qquad (1)$$
$$A \leftarrow T: \quad E[K_{AT}]\,(N_A, B, k, E[K_{BT}](k,A)) \qquad (2)$$
$$A \rightarrow B: \quad E[K_{BT}]\,(k, A) \qquad (3)$$
$$A \leftarrow B: \quad E[k]\,(N_B) \qquad (4)$$
$$A \rightarrow B: \quad E[k]\,(N_B\text{-}1) \qquad (5)$$

What is Kerberos?

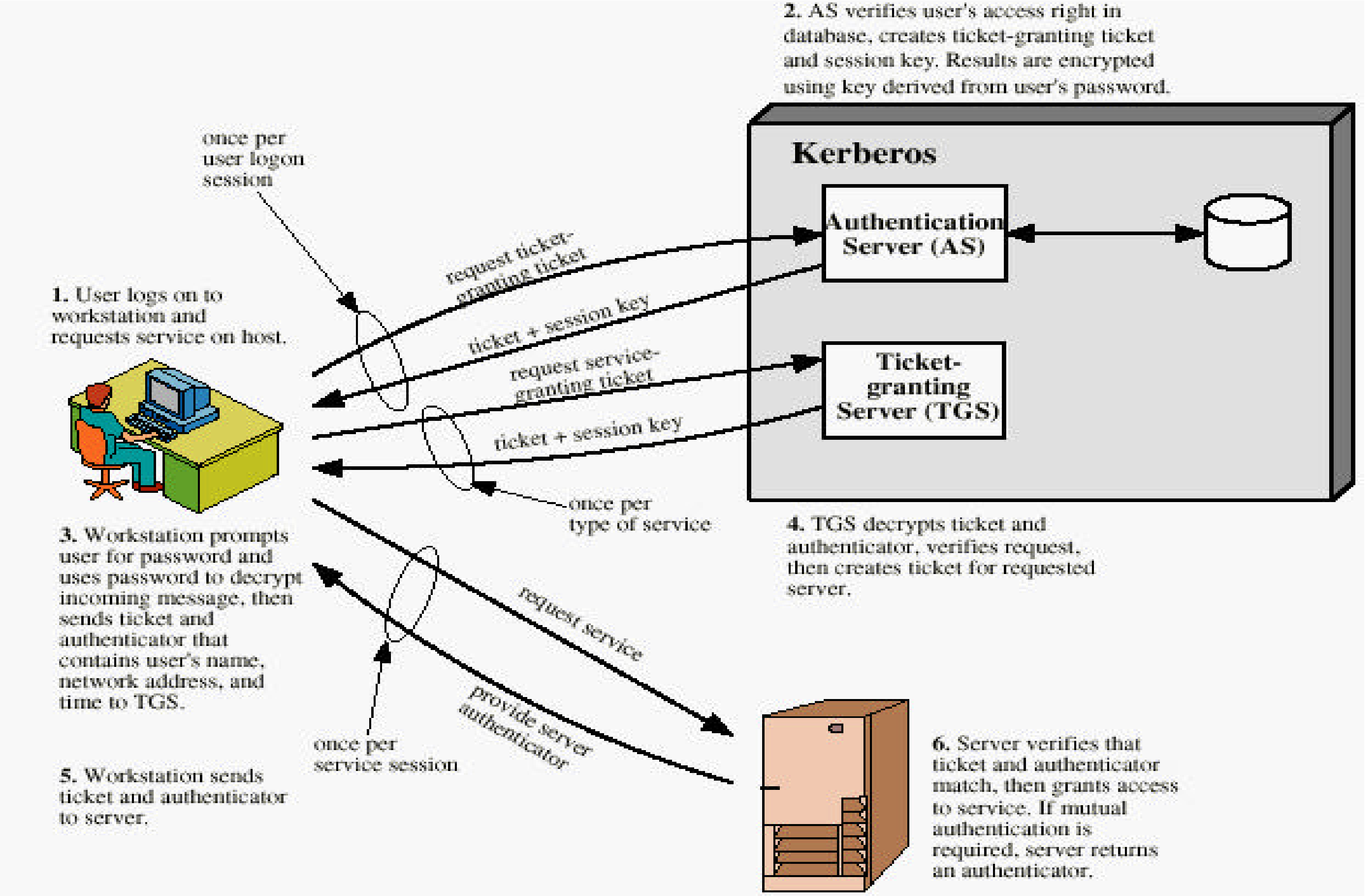
- Kerberos is a **network authentication protocol**
- Provides authentication for client-server applications, and data integrity and confidentiality
- Relies entirely on **symmetric cryptography**
- Developed at MIT: two versios, Version 4 and Version 5 (specified as RFC1510)
- http://web.mit.edu/kerberos/www



Kerberos Overview

- Client wants service from a particular server
- An Authentication Server allows access
- How? Based on tickets
- **Ticket**: specifies that a particular client (authenticated by the Authentication Server) has the right to obtain service from a specified server S
- **Realm**: network under the control of an Authentication Server

Overview of Kerberos

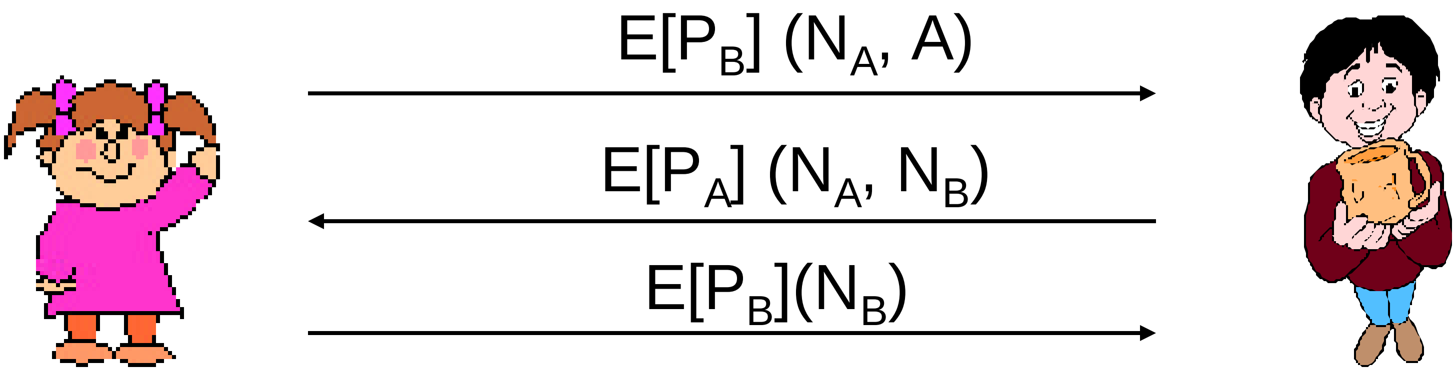


Key Establishment by Means of Public Key Encryption

- Often use public-key certificates
- Require off-line Trusted Third Party in the form of CA

Needham-Schroeder Public Key Protocol

- Setup: A and B both have each other's public key
- Goal: mutual entity authentication and authenticated key establishment
- [NS78]



Lowe's Attack on Needham-Schroeder Public-key Protocol [95]

The intruder can convince B that it is A.

$A \rightarrow I$: $E[P_I] (N_A, A)$

$I \rightarrow B$: $E[P_B] (N_A, A)$

$I \leftarrow B$: $E[P_A] (N_A, N_B)$

$A \leftarrow I$: $E[P_A] (N_A, N_B)$

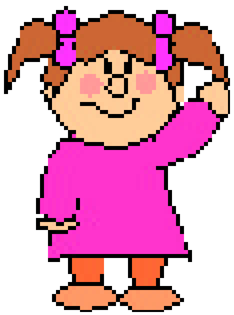
$A \rightarrow I$: $E[P_I] (N_B)$

$I \rightarrow B$: $E[P_B] (N_B)$

Fix: add B's name the second message

Key Agreement: Diffie-Hellman Protocol

- Key agreement protocol, both A and B contribute to the key
- Setup: p prime and g generator of Z_p*, p and g public.



g^a mod p →

← g^b mod p

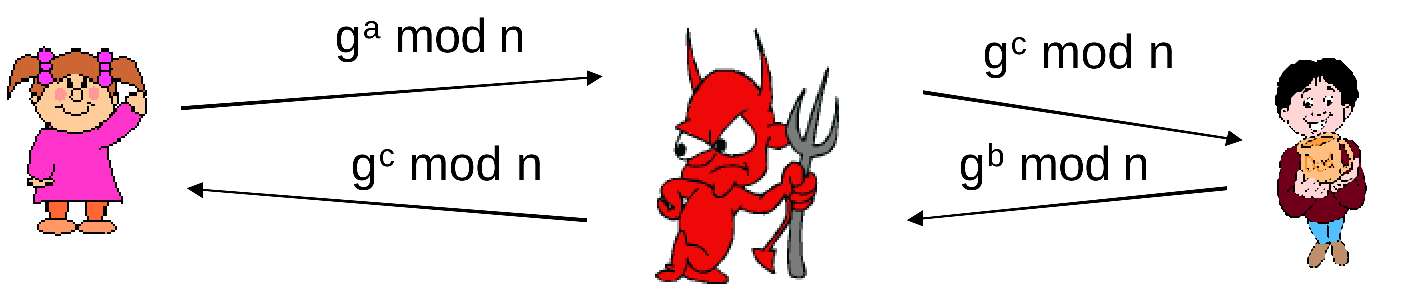
Pick random, secret a
Compute and send g^a mod p

$K = (g^b \bmod n)^a = g^{ab} \bmod p$

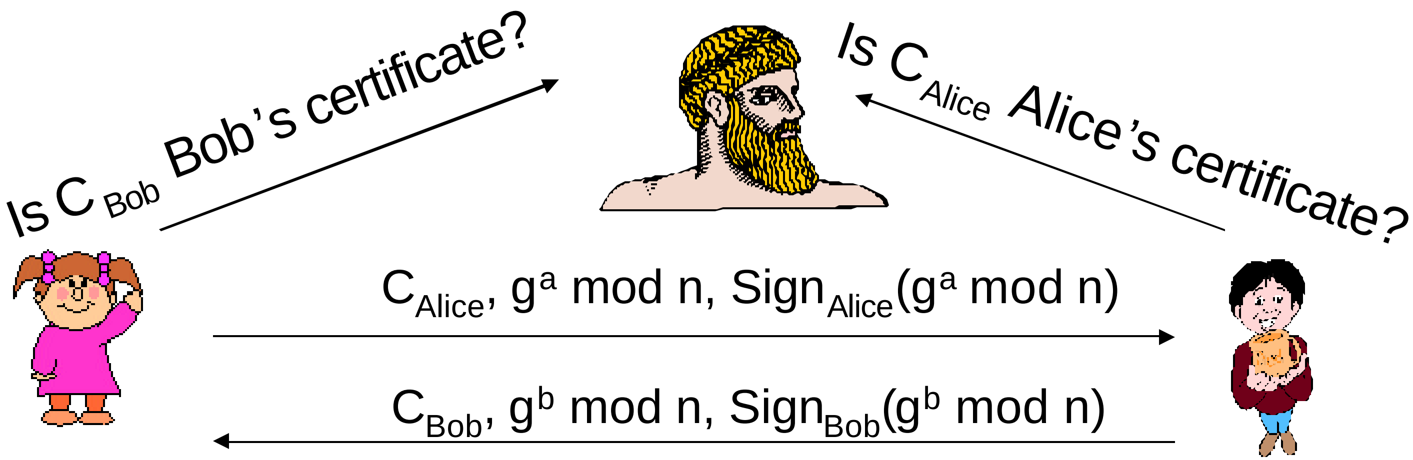
Pick random, secret b
Compute and send g^b mod p

$K = (g^a \bmod n)^b = g^{ab} \bmod p$

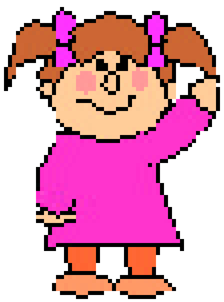
Authenticated Diffie-Hellman



Alice computes g^{ac} mod n and Bob computes g^{bc} mod n !!!



MTI



$g^x \bmod p$ →

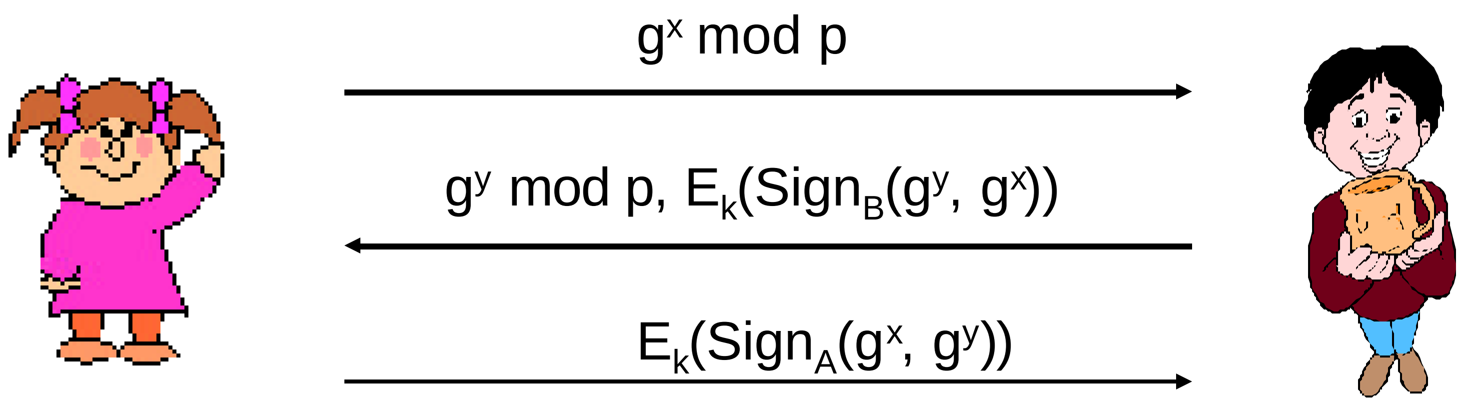
← $g^y \bmod p$

$k = (g^y)^a PK_b^{\,x} = g^{ya}\ g^{bx} = g^{ya+bx}$

$k = (g^x)^b PK_a^{\,y}$

- a and b are the private keys of A and B
- g^a and g^b are public keys of A and B
- Secure against passive attacks only
- Provides mutual (implicit) key authentication but neither key confirmation nor entity authentication

Station-to-Station (STS)



- where $k=(g^x)^y \bmod p$
- Provides mutual entity authentication

Coming Attractions …

- Information Theory