CS 4173/5173
COMPUTER SECURITY

Key Distribution Center (KDC)

OUTLINE LAST TIME

- Design a perfect authentication protocol requires non-trivial efforts
  - Can be based on symmetric or public key systems
- Some guidelines to check a protocol:
  - The initiators should authenticate themselves first
  - Need asymmetric challenge-response, be aware of reflection attacks
    - Make two parties do different things
  - Provide mutual authentication
  - Avoid message decryption
- Design based on public key:
  - RSA key negotiation
  - Diffie-Hellman with authentication

TRUSTED KEY SERVERS

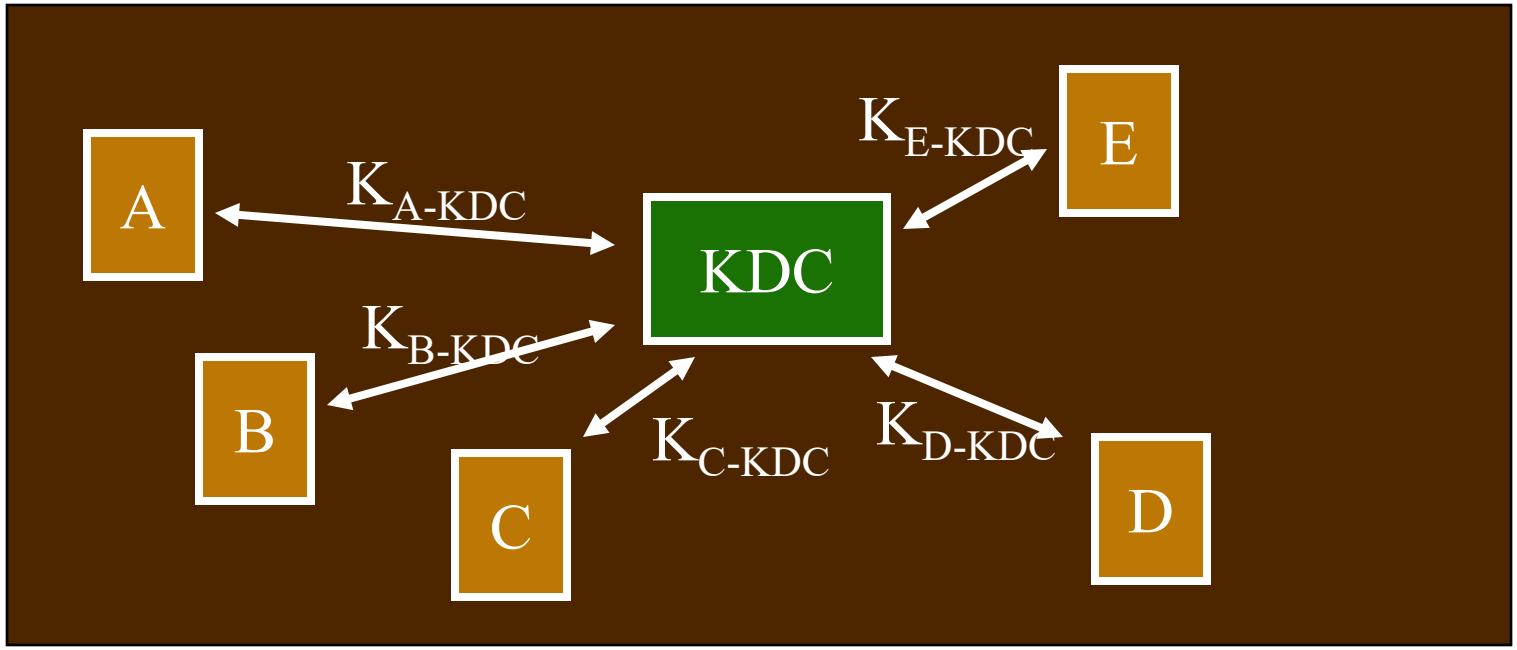
- How do a large number of users authenticate each other?
  - inefficient / impractical for every pair of users to negotiate a secret key or share passwords
- Alternative: everybody shares a key with (and authenticates to) a single trusted third-party

TRUSTED INTERMEDIARIES

- Problem: authentication for large networks
- Solution #1
  - Key Distribution Center (KDC)
    - Representative solution: Kerberos
  - Based on secret key cryptography
- Solution #2
  - Public Key Infrastructure (PKI)
  - Based on public key cryptography

KEY DISTRIBUTION CENTER

- Shared keys between the *Key Distribution Center (KDC)* and users

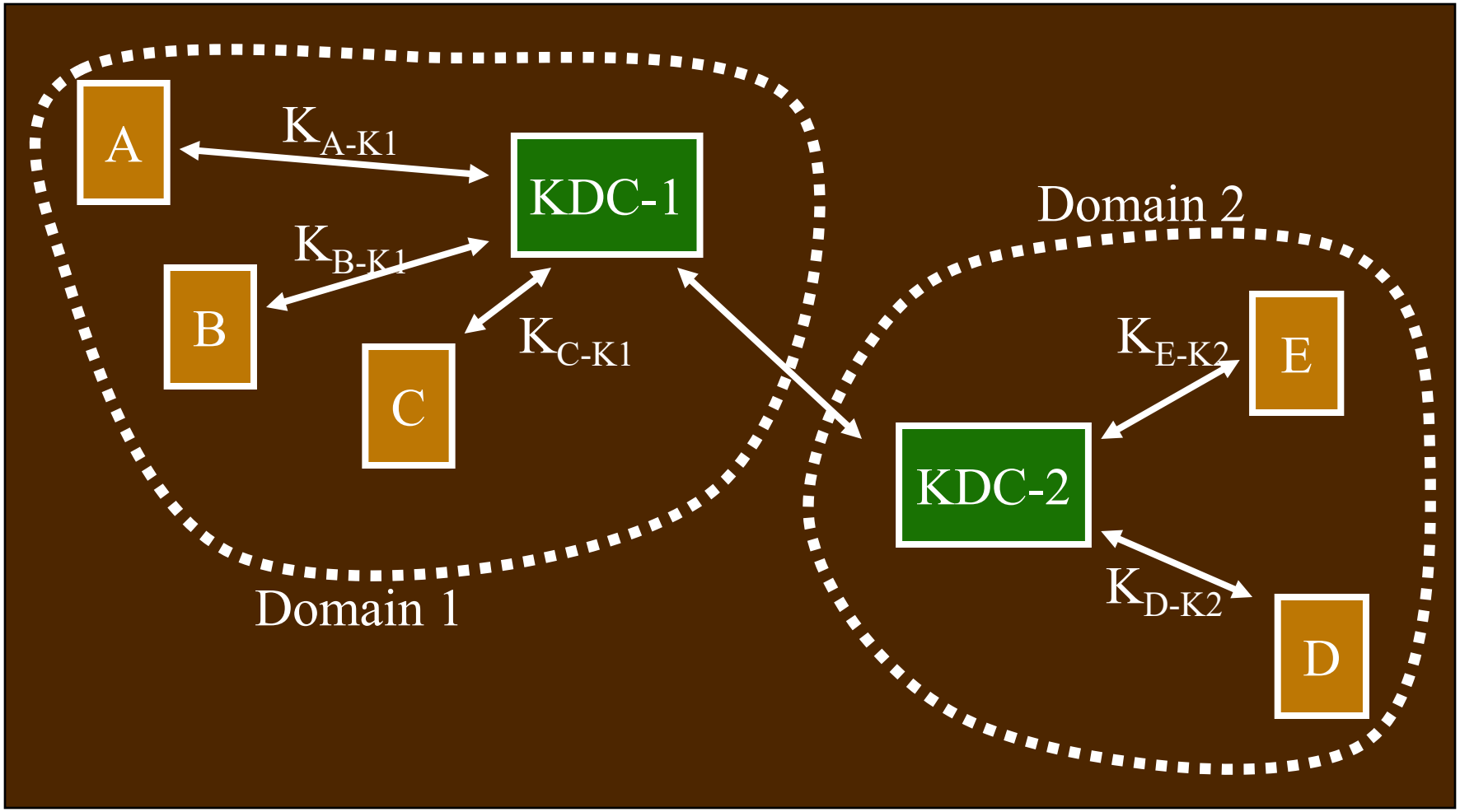


Q: Can users establish the same key with KDC?

(SIMPLIFIED) EXAMPLE OF USE

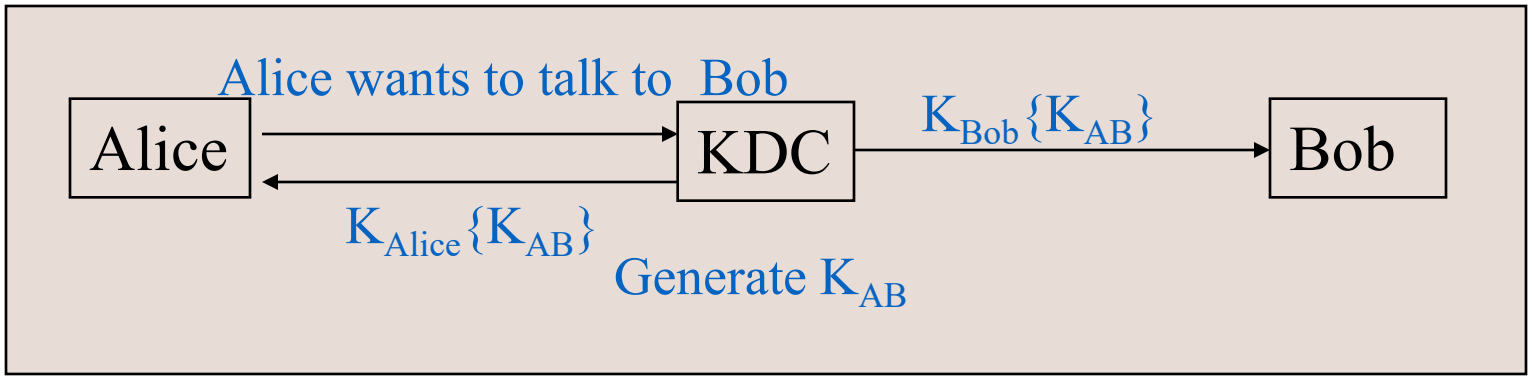
- Alice wishes to communicate securely with Bob; Alice has previously shared $K_{A\text{-}KDC}$ with the KDC, Bob has negotiated $K_{B\text{-}KDC}$
  1. Alice requests from the KDC a session key to use with Bob
  2. KDC generates session key K_S, sends to Alice, encrypted with $K_{A\text{-}KDC}$
  3. KDC also sends K_S to Bob, encrypted with $K_{B\text{-}KDC}$
     - Alice and Bob can then communicate using K_S

A HIERARCHY OF KDCS



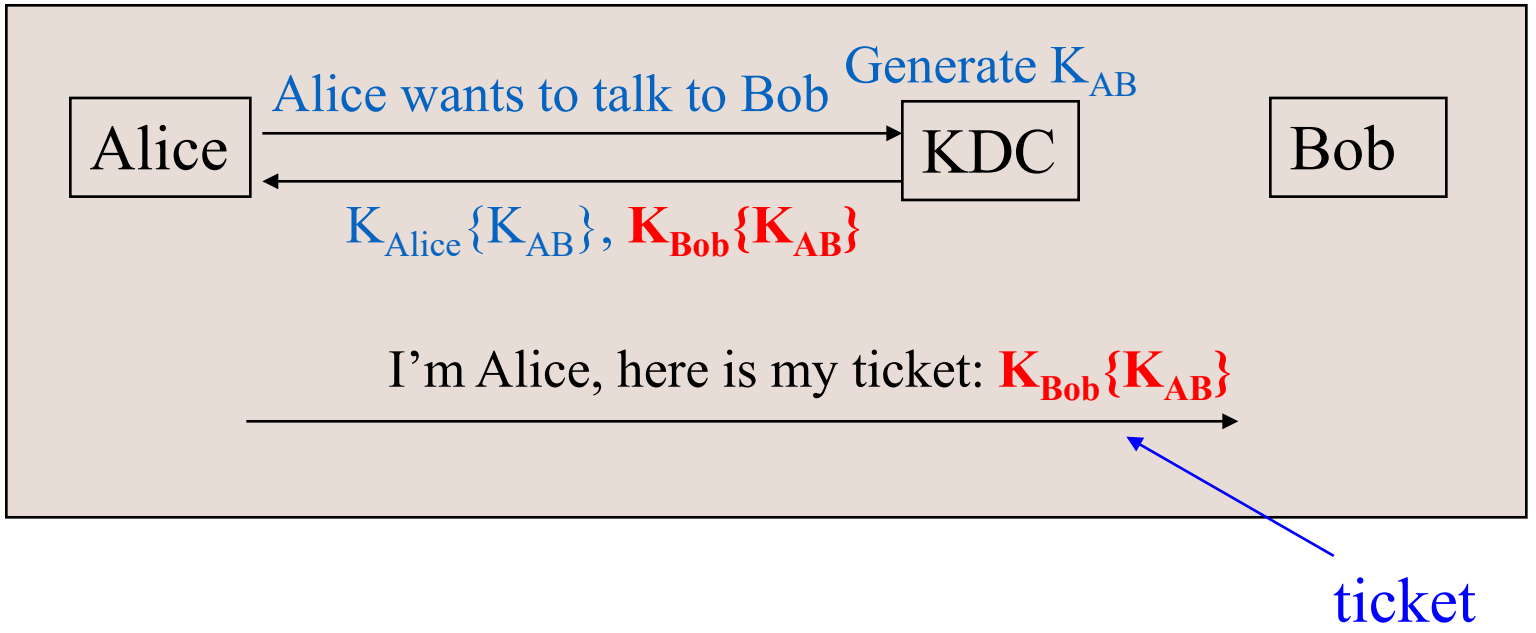
MEDIATED AUTHENTICATION (WITH KDC)

KDC operation (in principle)



- Some concerns
  - Trudy may claim to be Alice and talk to KDC
    - Trudy cannot get anything useful
  - Messages encrypted by Alice using K_{AB} may arrive at Bob before KDC's message $K_{Bob}\{K_{AB}\}$ arrive

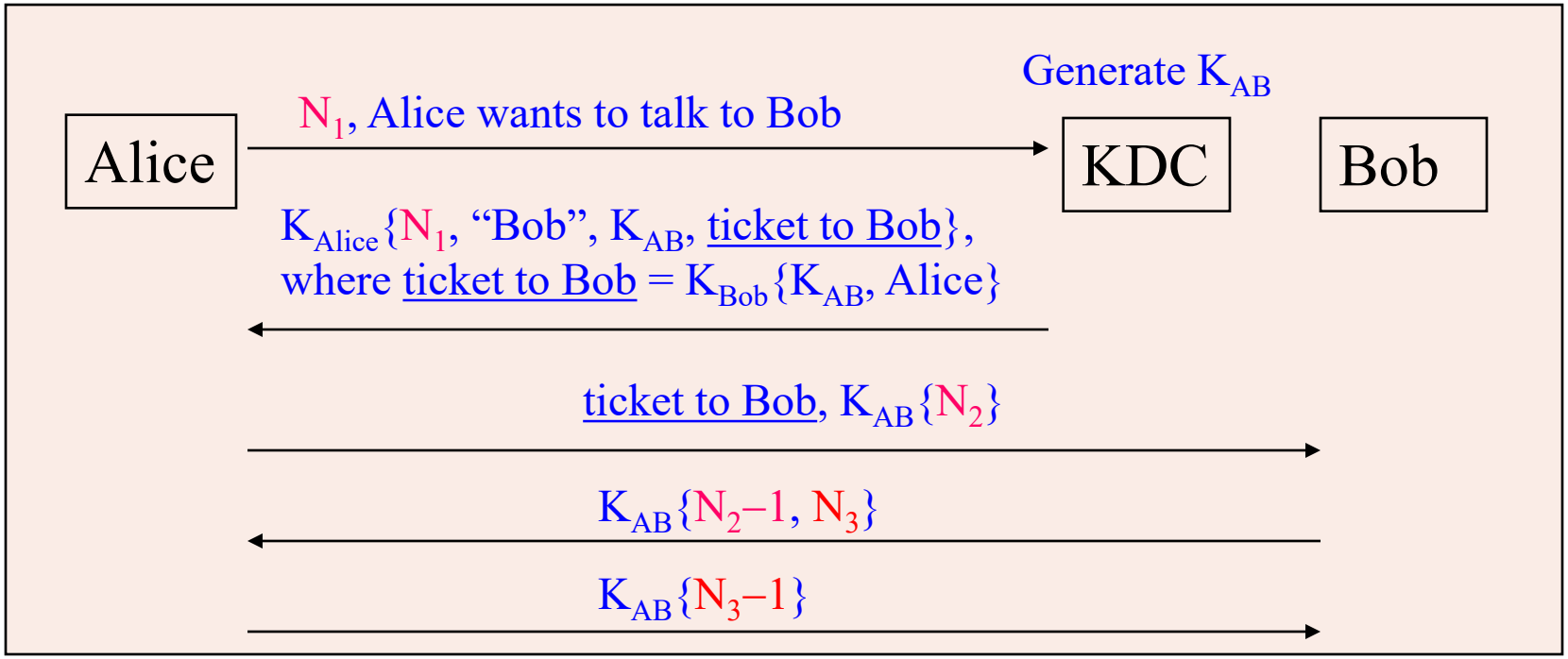
MEDIATED AUTHENTICATION (WITH KDC)



- Must be followed by a mutual authentication exchange
  - To confirm that Alice and Bob have the same key

NEEDHAM-SCHROEDER PROTOCOL

- Classic protocol for authentication with KDC
  - Many others have been modeled after it (e.g., Kerberos)

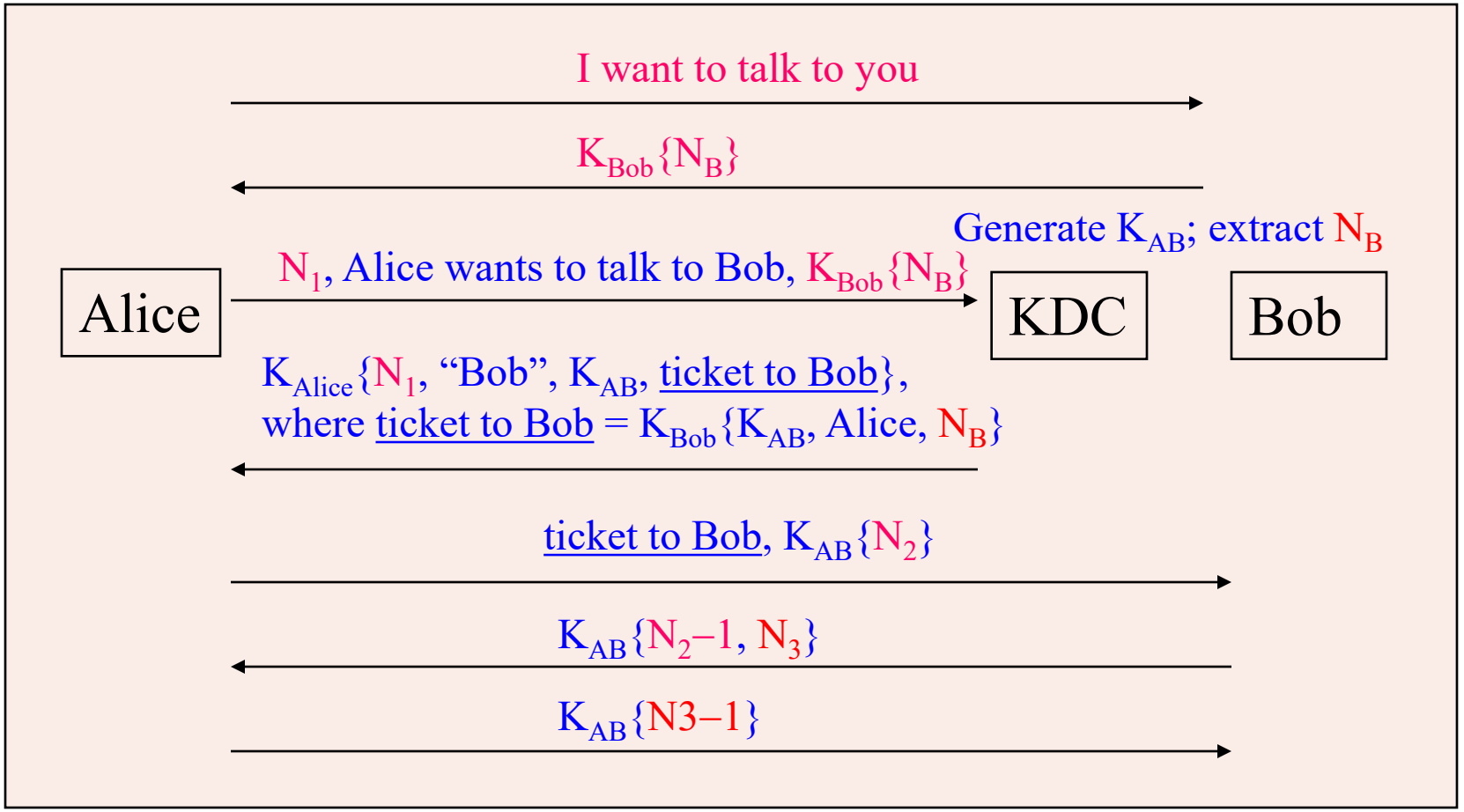


Q: Why N_1, N_2, N_3?

NEEDHAM-SCHROEDER PROTOCOL (CONT'D)

- A vulnerability
  - When Trudy gets a previous key used by Alice, Trudy may reuse a previous ticket issued to Bob for Alice
    - Example of relay attacks
  - Essential reason
    - The ticket to Bob stays valid even if Alice changes her key

EXPANDED NEEDHAM-SCHROEDER PROTOCOL



AUTHENTICATION IN LARGE NETWORKS

- Problem: authentication for large networks
- Solution #1
  - Key Distribution Center (KDC)
  - Based on secret key cryptography
  - Representative solution: Kerberos
- Solution #2
  - Public Key Infrastructure (PKI)
  - Based on public key cryptography
  - Representative solution: SSL/TLS

CS 4173/5173
COMPUTER SECURITY

Kerberos

GOALS OF KERBEROS

1. User ↔ server mutual authentication
2. Users should only need to authenticate once to obtain services from multiple servers
3. Should scale to large numbers of users and servers
   - makes use of a Key Distribution Center so servers don't need to store information about users

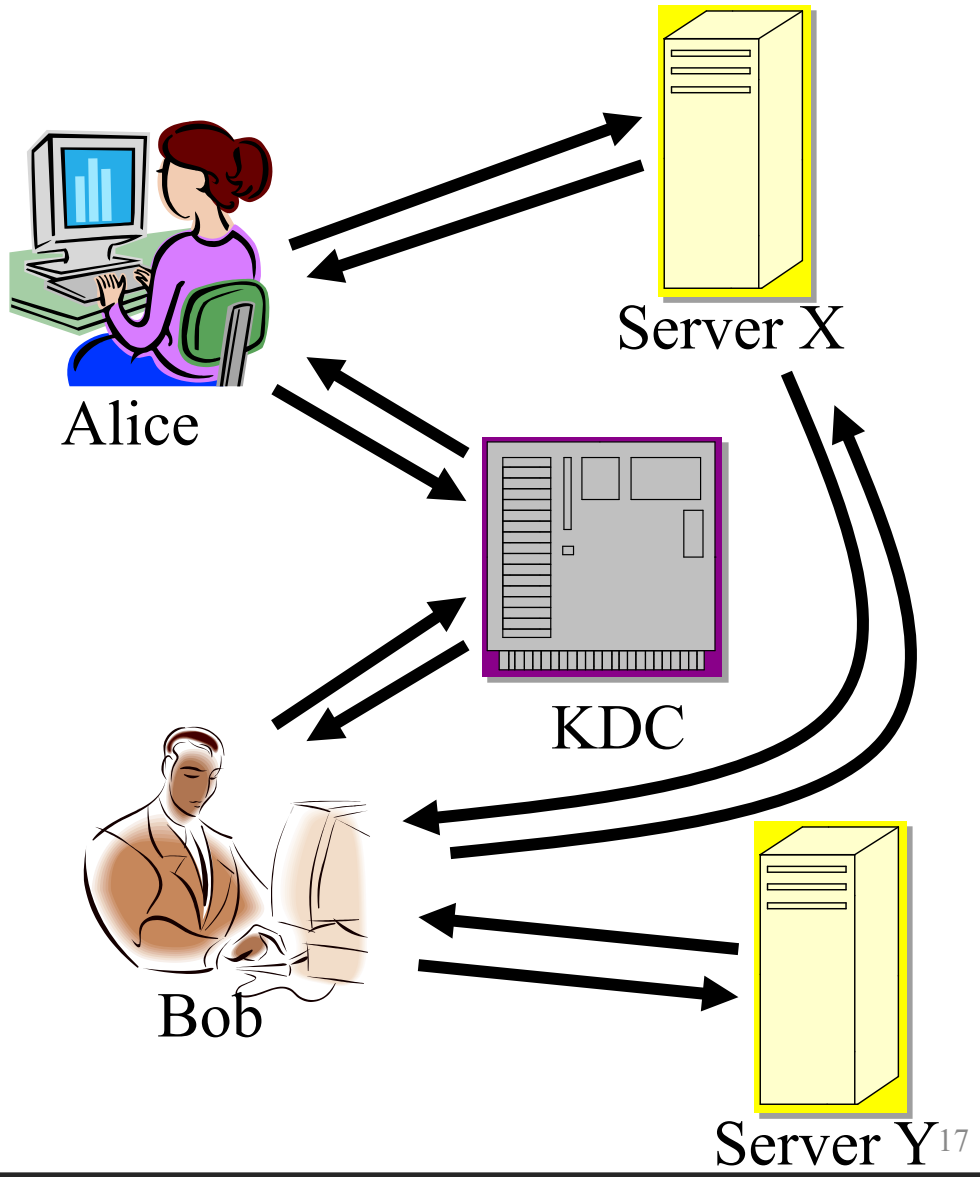
SOME PROPERTIES

- Kerberos uses only secret key (symmetric) encryption
  - originally, only DES, but now 3DES and AES as well
- A *stateless* protocol
  - KDCs do not need to remember what messages have previously been generated or exchanged
  - the state of the protocol negotiation is contained in the message contents

EXAMPLE SCENARIO

- Alice wants to make use of services from X, contacts the KDC to authenticate, gets ticket to present to X

- Bob wants to make use of services from X and Y, contacts the KDC, gets tickets to present to X and Y



THE KDC

- Infrastructure needed (KDC components)
  1. the database of user information (IDs, password hash, shared secret key, etc.)
  2. an authentication server (AS)
  3. a ticket-granting server (TGS)
- The KDC of course is critical and should be carefully guarded

SECRETS MANAGED BY THE KDC

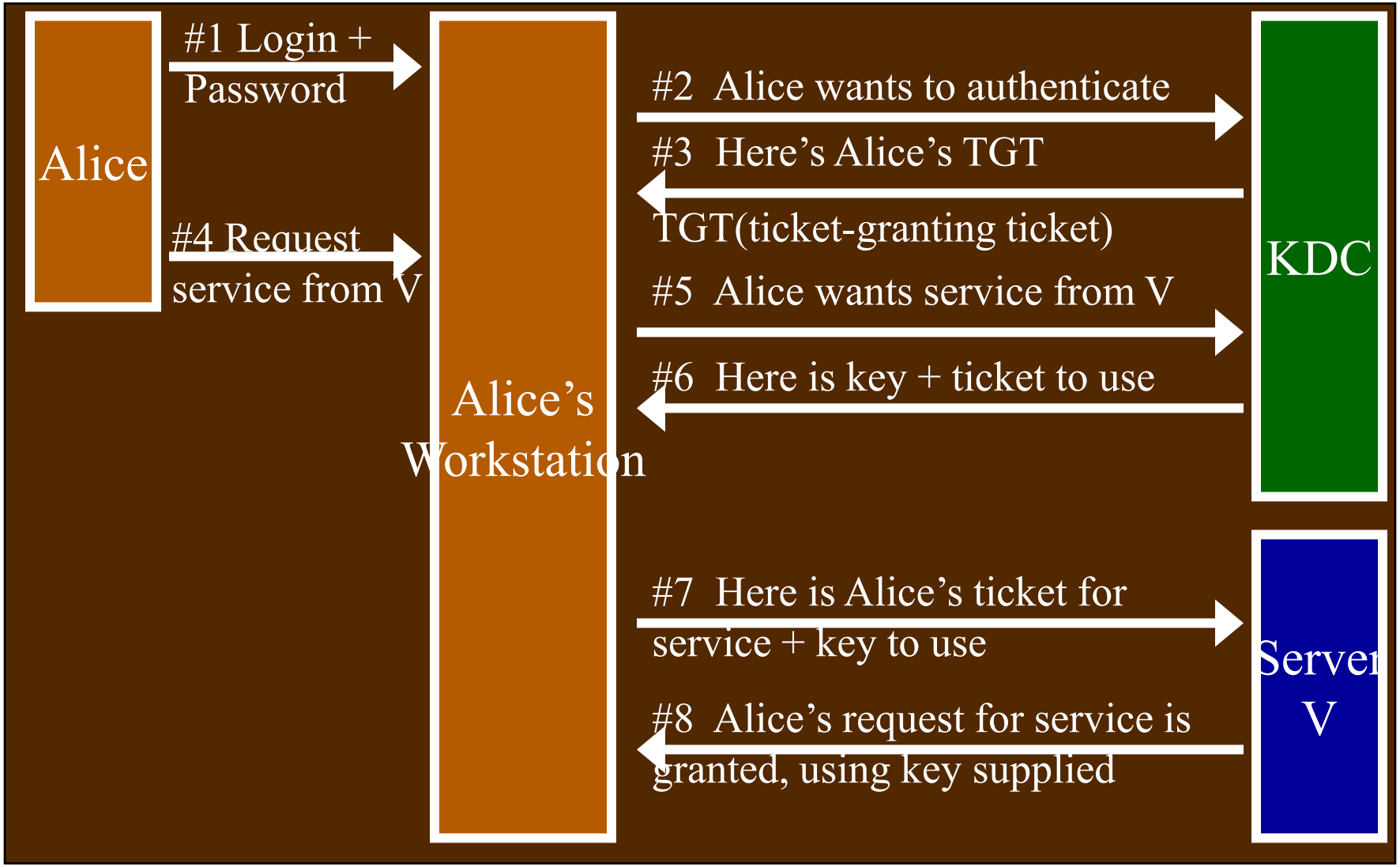
- A *personal* key used for encrypting/decrypting the database
- A master shared key for each server

CS 4173/5173 COMPUTER SECURITY

Basics of the Kerberos v4 Standard

(details not required in homework/exams)

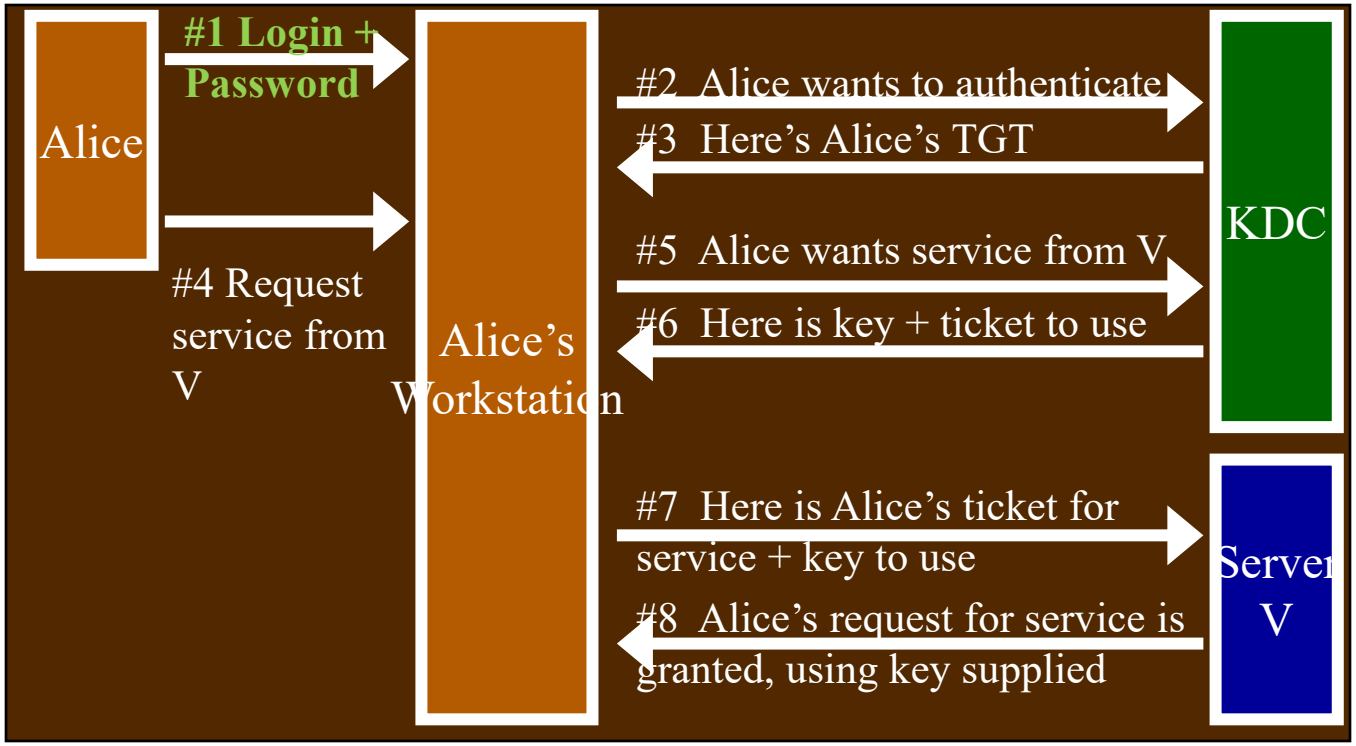
PROTOCOL SKETCH (COMMON CASE)



MSG#1: ENTER PASSWORD

#1 A→W: **"Alice" | password**

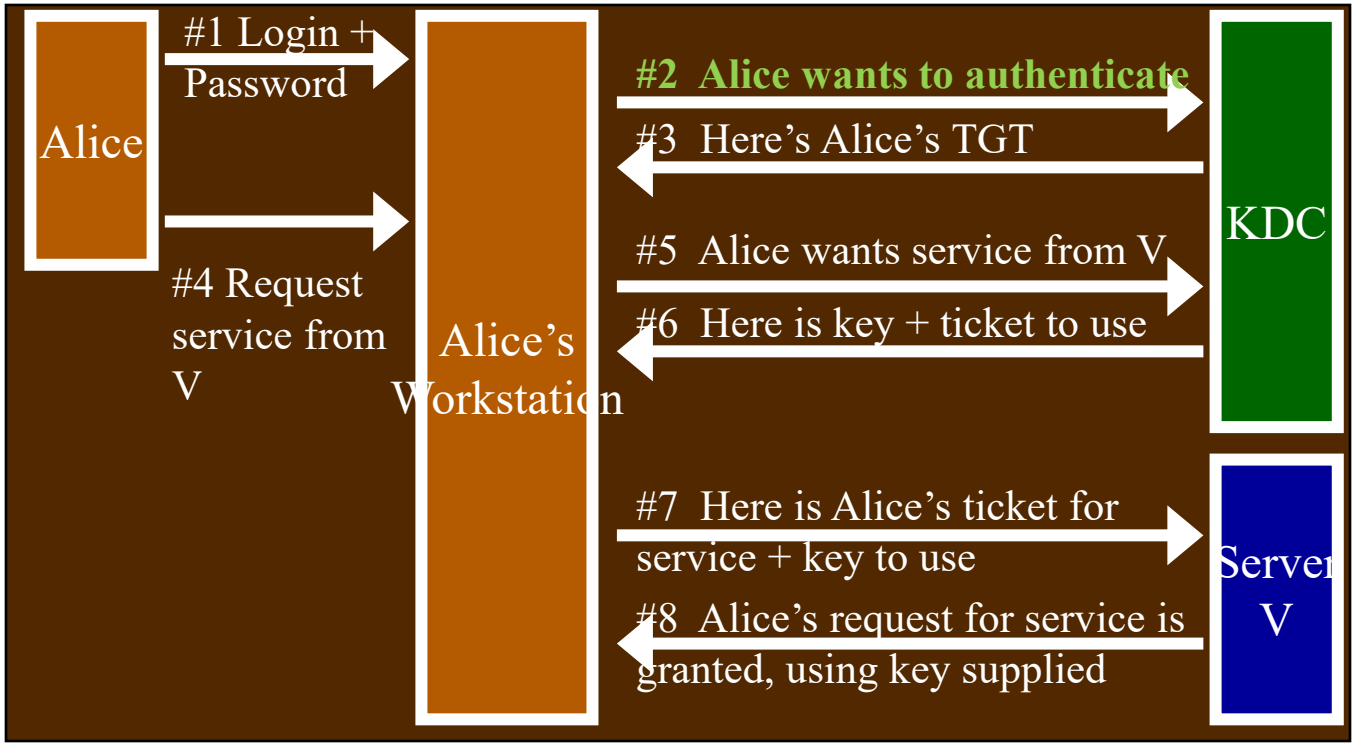
- Alice types in her user ID and password in unencrypted form into her workstation



MSG#2: REQUEST FOR AUTHENTICATION

#2. W→KDC: $ID_A \mid TS_2 \mid ID_{KDC}$

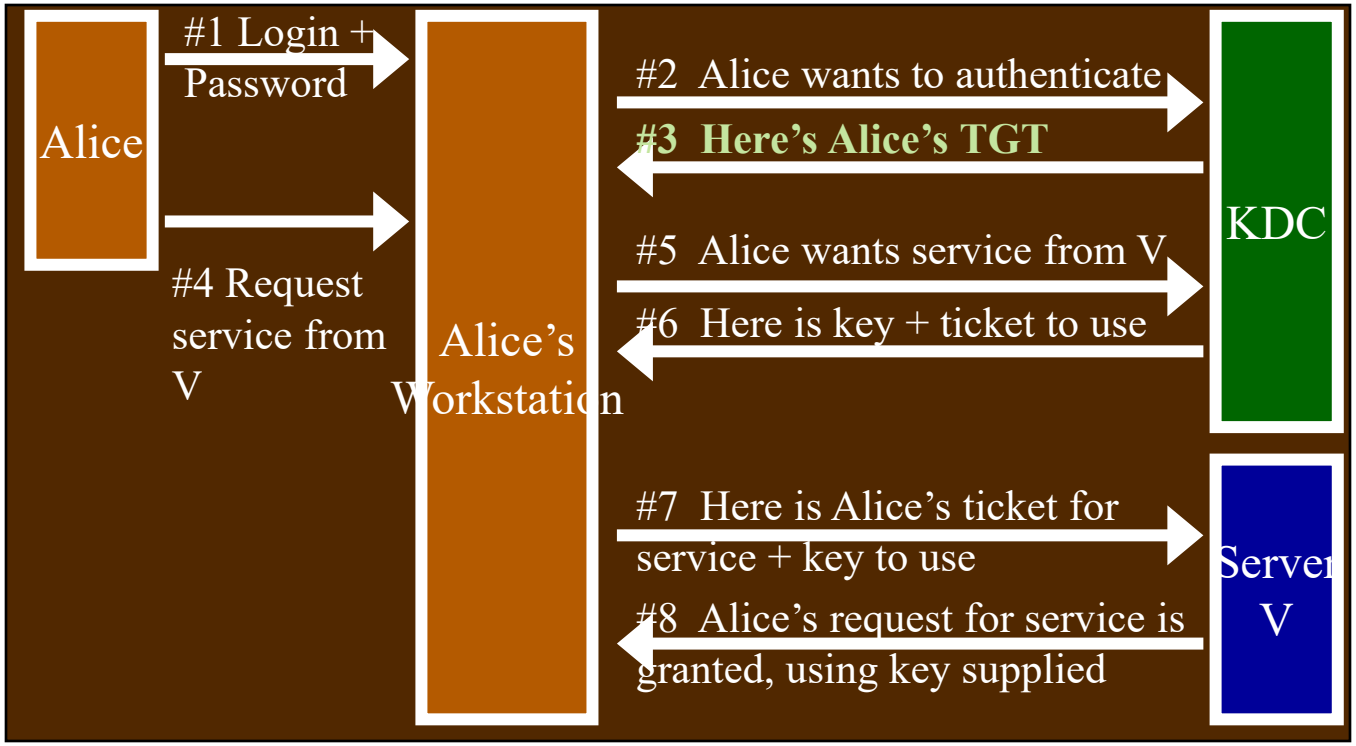
- Workstation sends a message to KDC with Alice's ID (in unencrypted form)
- Many of these messages contain timestamps, for a) liveness, and b) anti-replay (relay attacker: intercept some messages and reply them later)



MSG#3: AUTHENTICATION SUCCESS

#3. KDC→W: $K_{A\text{-}KDC}(ID_A \mid TS_3 \mid Lifetime_3 \mid \mathcal{K}_{A\text{-}S} \mid ID_{KDC} \mid TGT)$

- KDC sends Alice's workstation a session key and a TGT (ticket-granting ticket)
  - encrypted with the master key shared between Alice and the KDC
- $K_{A\text{-}KDC}$ is derived from Alice's password, used to decrypt session key $\mathcal{K}_{A\text{-}S}$



MSG#3: … (CONT'D)

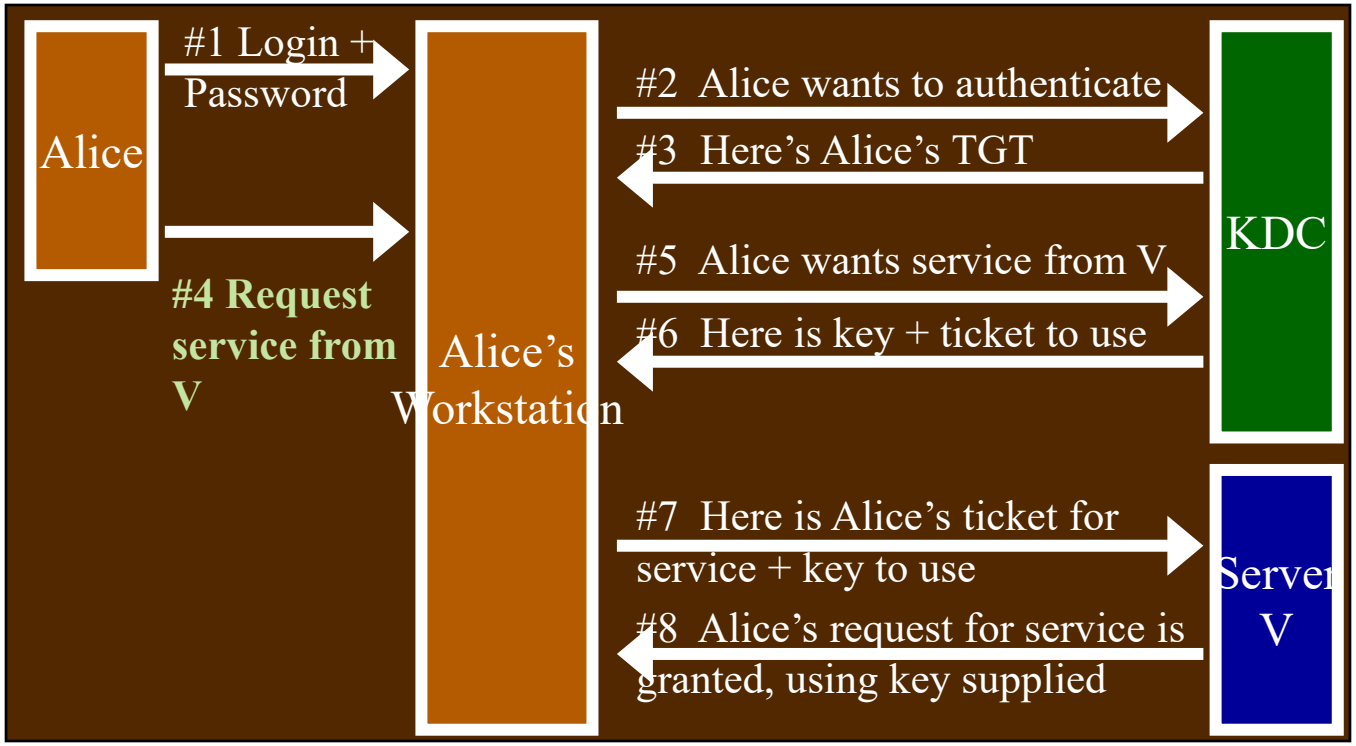
TGT: $K_{A\text{-}KDC}(ID_A \mid Addr_A \mid \mathcal{K}_{A\text{-}S} \mid Lifetime_{TGT} \mid TS_{TGT} \mid ID_{KDC})$

- The TGT is what allows the KDC to be stateless
  - means simpler, more robust KDC design
  - allows replicated KDCs (see later)
- The TGT contains
  - the session key to be used henceforth
  - the user ID (Alice)
  - the valid lifetime for the TGT

MSG#4: ALICE REQUESTS SERVICE V

#4 A→W: **ReqServ(V)**

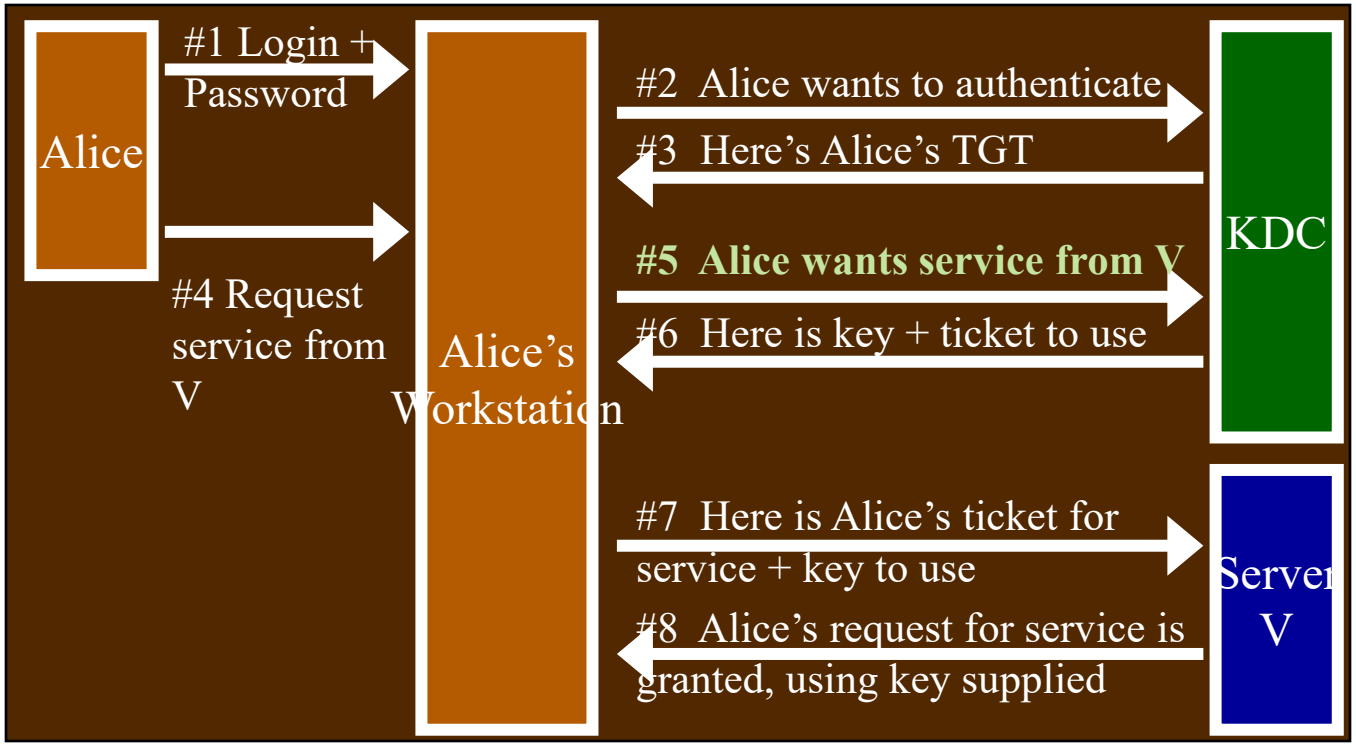
- Alice enters a request in the workstation to access the service provided by V



MSG#5: WORKSTATION REQUESTS SERVICE V

#5 W→KDC: $TGT \mid authenticator_5 \mid TS_5 \mid Lifetime_5 \mid ID_V$

- Workstation sends to the KDC…
  - the TGT previously granted, the server she wishes to request service from
  - an authenticator for this message



MSG#5… (CONT'D)

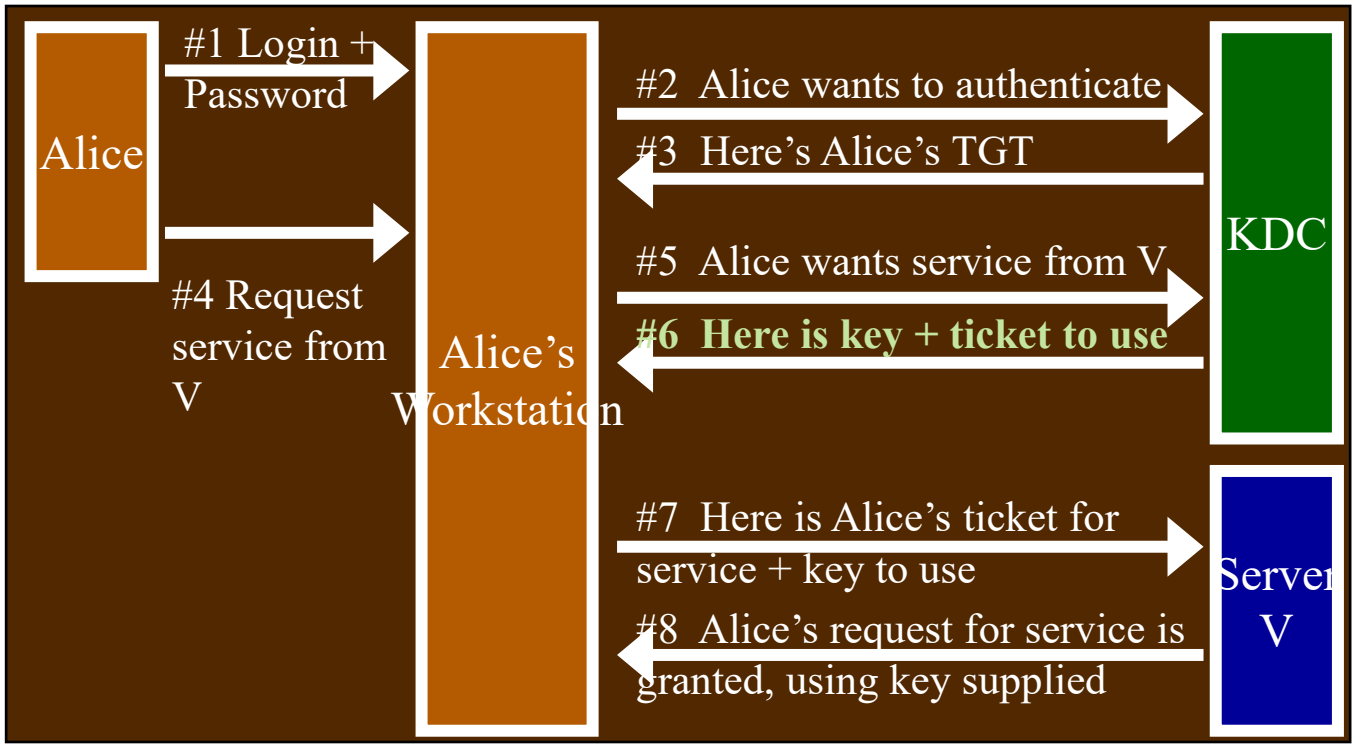
$$\mathcal{K}_{A\text{-}S}(ID_A \mid TS_{auth5})$$

- The authenticator is an encrypted timestamp
  - Why?
    - The KDC is able to verify
      - Alice's identify
      - the time that Alice required the service
      - Check the lifetime
  - (reminder: timestamps requires user and KDC clocks to be loosely synchronized)

MSG#6: KDC GENERATES TICKET

#6 KDC→W: $\mathcal{K}_{A\text{-}S}(ID_A \mid TS_6 \mid Lifetime_6 \mid \mathcal{K}_{A\text{-}V} \mid ID_V \mid TGT_V)$

- KDC decrypts the TGT and…
  - checks that lifetime has not expired; gets the shared key $\mathcal{K}_{A\text{-}S}$
- KDC sends back to workstation
  - identity of the server; a shared key ($\mathcal{K}_{A\text{-}V}$) for Alice and the server; a ticket (TGT) for Alice to present to V



MSG#6… (CONT'D)

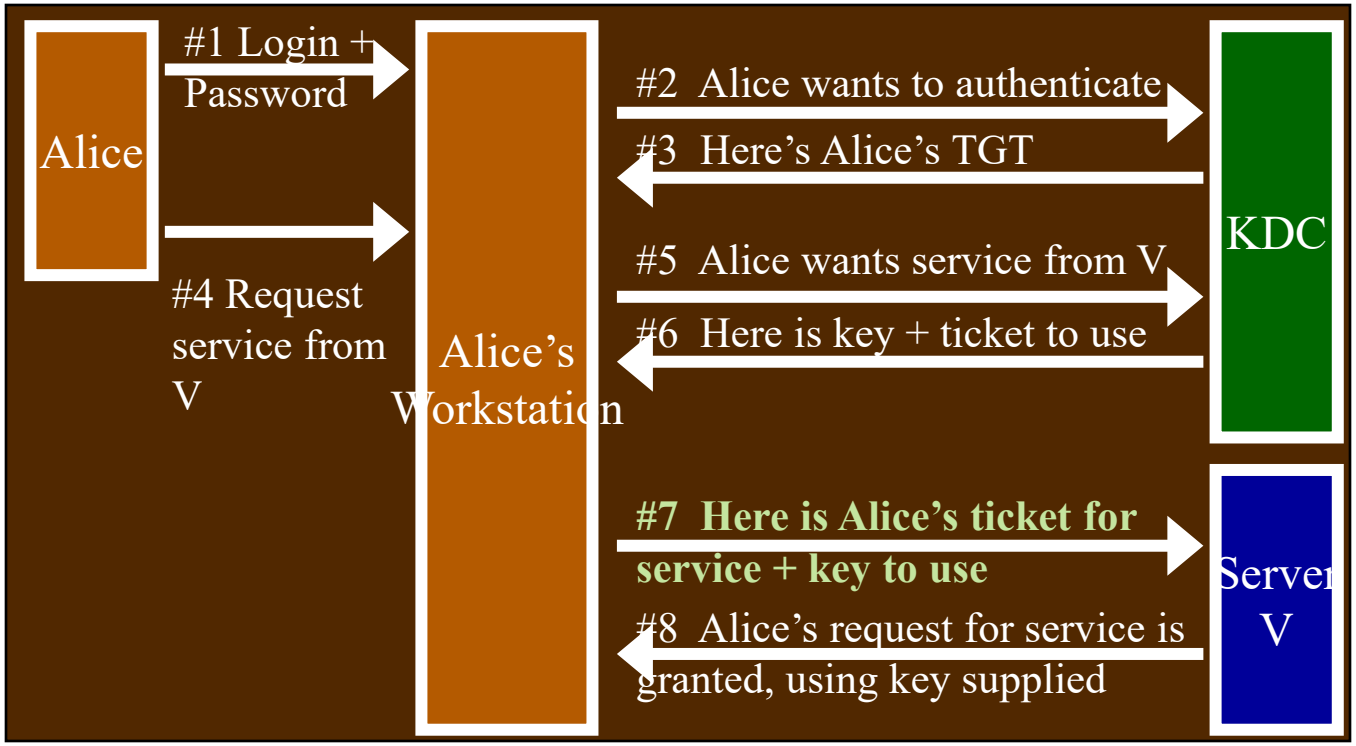
TGT_V: $K_{V\text{-}KDC}(ID_A \mid Addr_A \mid \mathcal{K}_{\mathcal{A}\text{-}\mathcal{V}} \mid Lifetime_{TKT} \mid TS_{TKT} \mid ID_V)$

- The ticket is encrypted by the master key shared between Sever V and the KDC $\mathbf{K_{V\text{-}KDC}}$
- The ticket contains
  - ID of the initiating user (i.e. Alice)
  - shared key $\mathcal{K}_{\mathcal{A}\text{-}\mathcal{V}}$
  - lifetime of the ticket

MSG#7: WORKSTATION CONTACTS SERVER

#7 W→V: $ID_V \mid TGT_V \mid authenticator_7$

- Message contains
  - ticket (from the KDC); authenticator



MSG#7… (CONT'D)

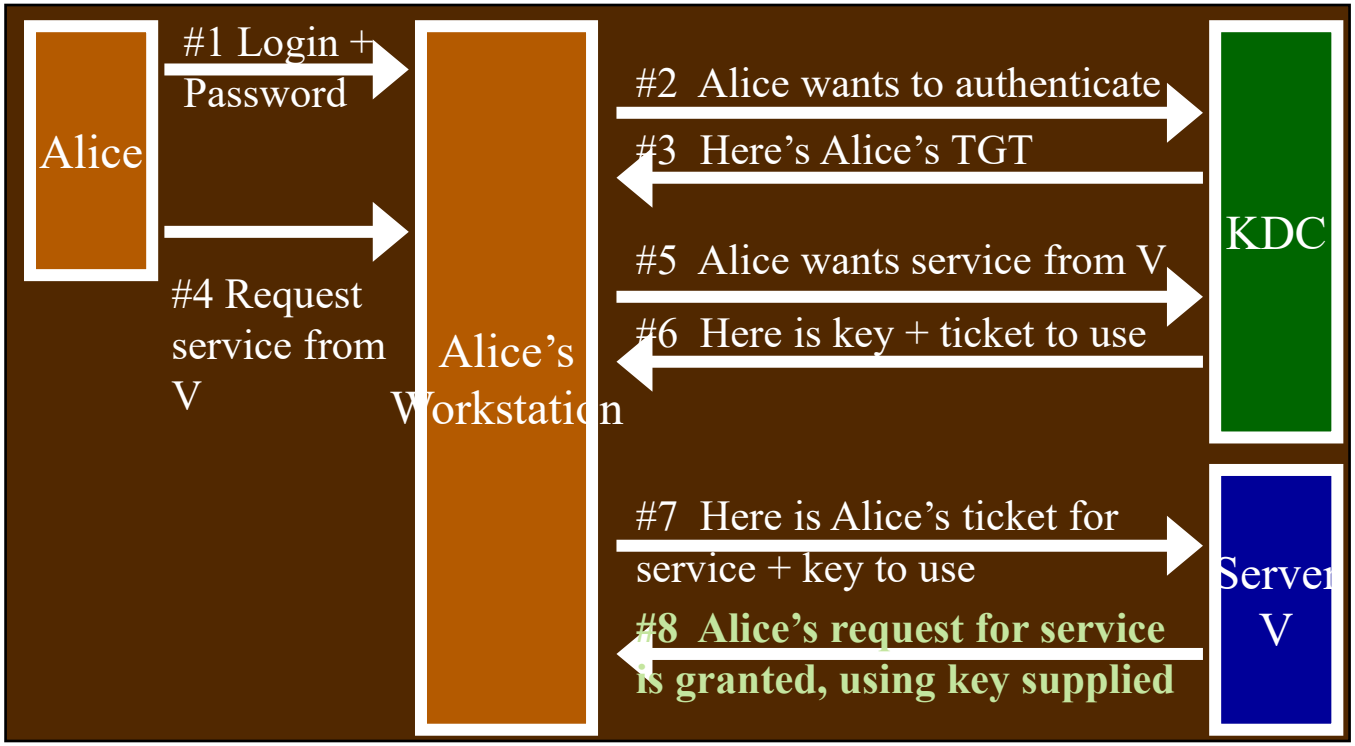
$$\mathcal{K}_{\mathcal{A}\text{-}\mathcal{V}}(ID_A \mid Chksum_{auth7} \mid TS_{auth7})$$

- Authenticator is valid for 5 minutes
  - loose synchronization required

MSG#8: SERVER AUTHENTICATES TO ALICE

#8 V→W : $\mathcal{K}_{A\text{-}V}(\text{Chksum}_{auth7} + 1)$

- Reply to Alice's workstation contains
  - checksum sent by Alice, incremented by 1



DONE!

1. Alice has authenticated to KDC (which is trusted by server)
2. Server has authenticated to Alice
3. A session key has been negotiated, for encryption, message authentication, or both.