Key Management

Key Management Problem: Getting Help from Trusted Intermediaries

Symmetric (Secret) key problem:

- How do two entities establish shared secret key over network?

Solution:

- trusted key distribution center (KDC) acting as intermediary between entities

Public key problem:

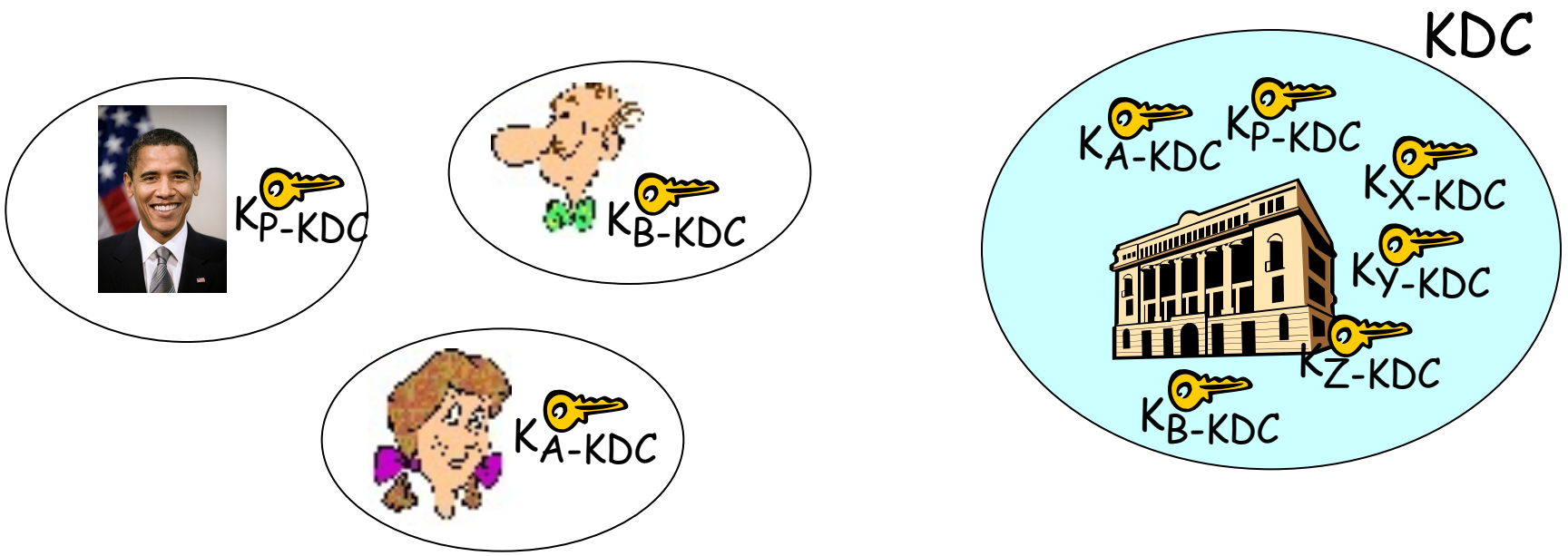
- When Alice obtains Bob's public key (from web site, e-mail, diskette), how does she know it is Bob's public key, not Trudy's?

Solution:

- Digital Certificate Issued by trusted certification authority (CA)

Key Distribution Center (KDC)

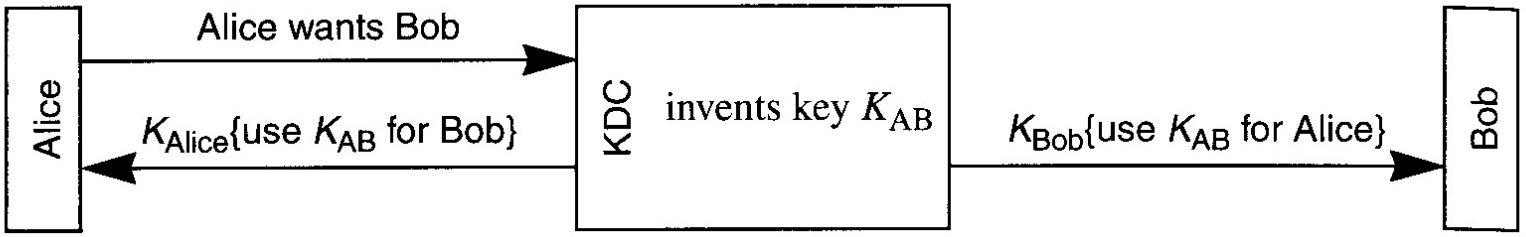
- Alice, Bob need a shared symmetric key.
- KDC: server shares different secret key with *each* registered user (many users)
- Alice, Bob know own symmetric keys, $K_{A\text{-}KDC}$ $K_{B\text{-}KDC}$, for communicating with KDC.



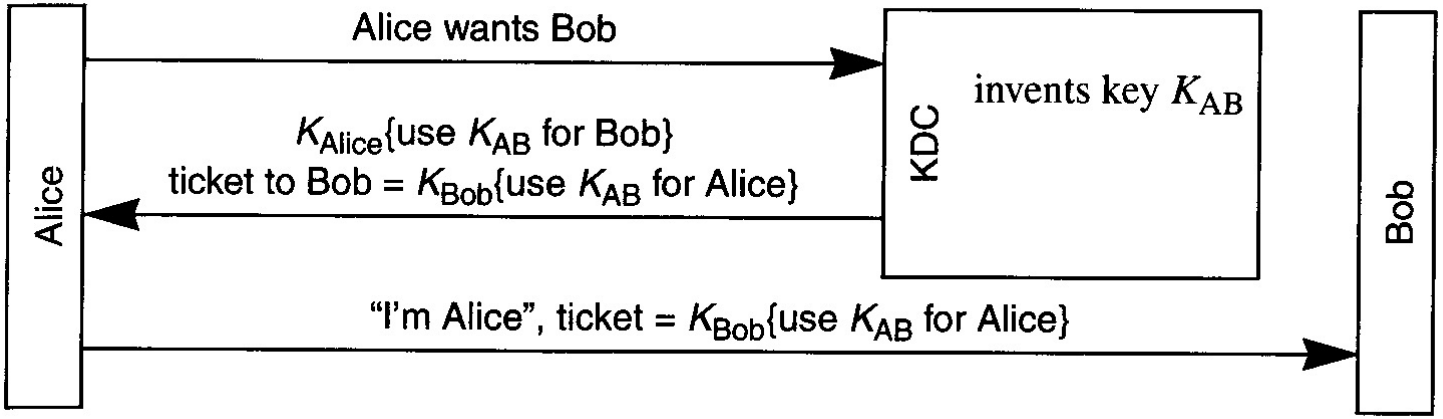
Key Distribution Center (KDC)

- Responsible for distributing keys to pairs of users (hosts, processes, applications)
- Each user must share a unique key, the *master* key, with the KDC
  - Use the master key to communicate with KDC to get a temporary *session* key for establishing a secure "session" with another user
  - Master keys are distributed in some *non-cryptographic* ways

KDC Operating Principles



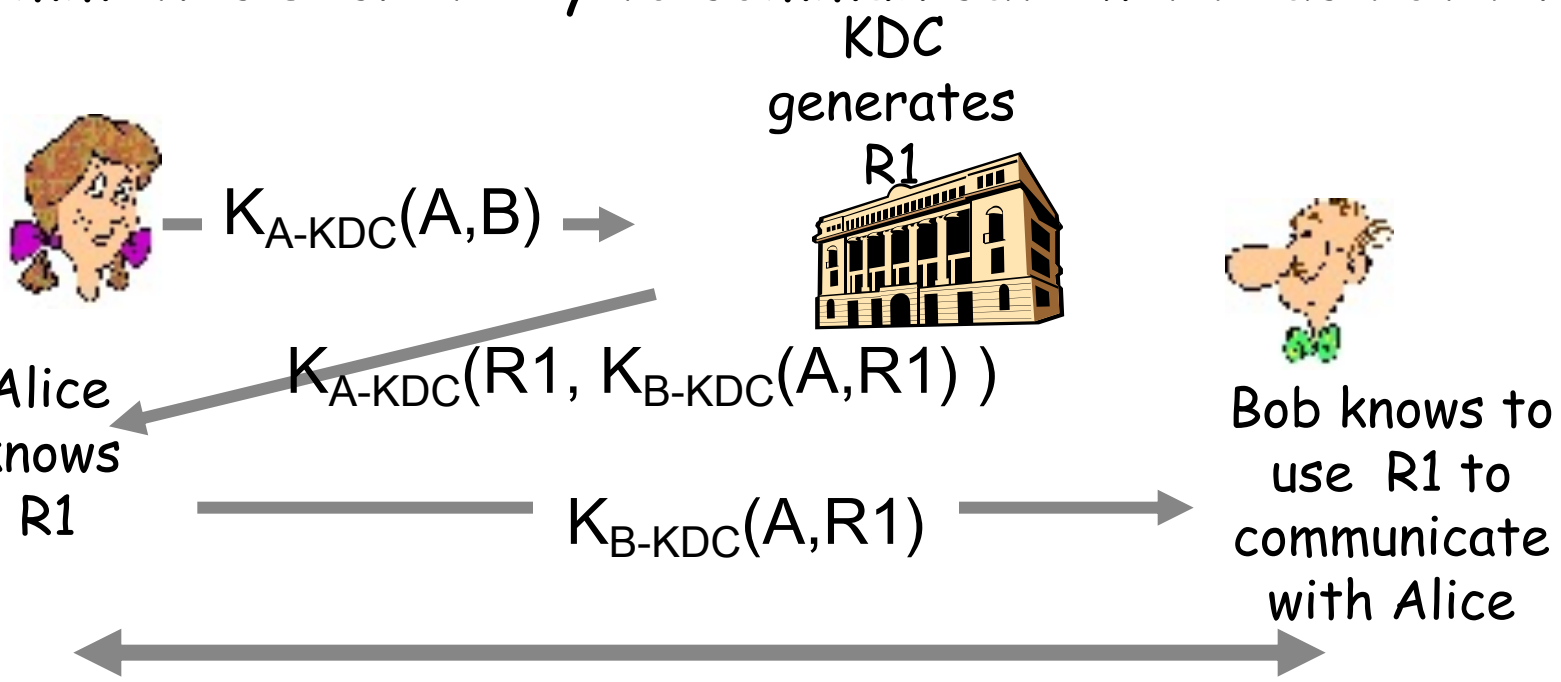
a) Overall concept



b) First Refinement

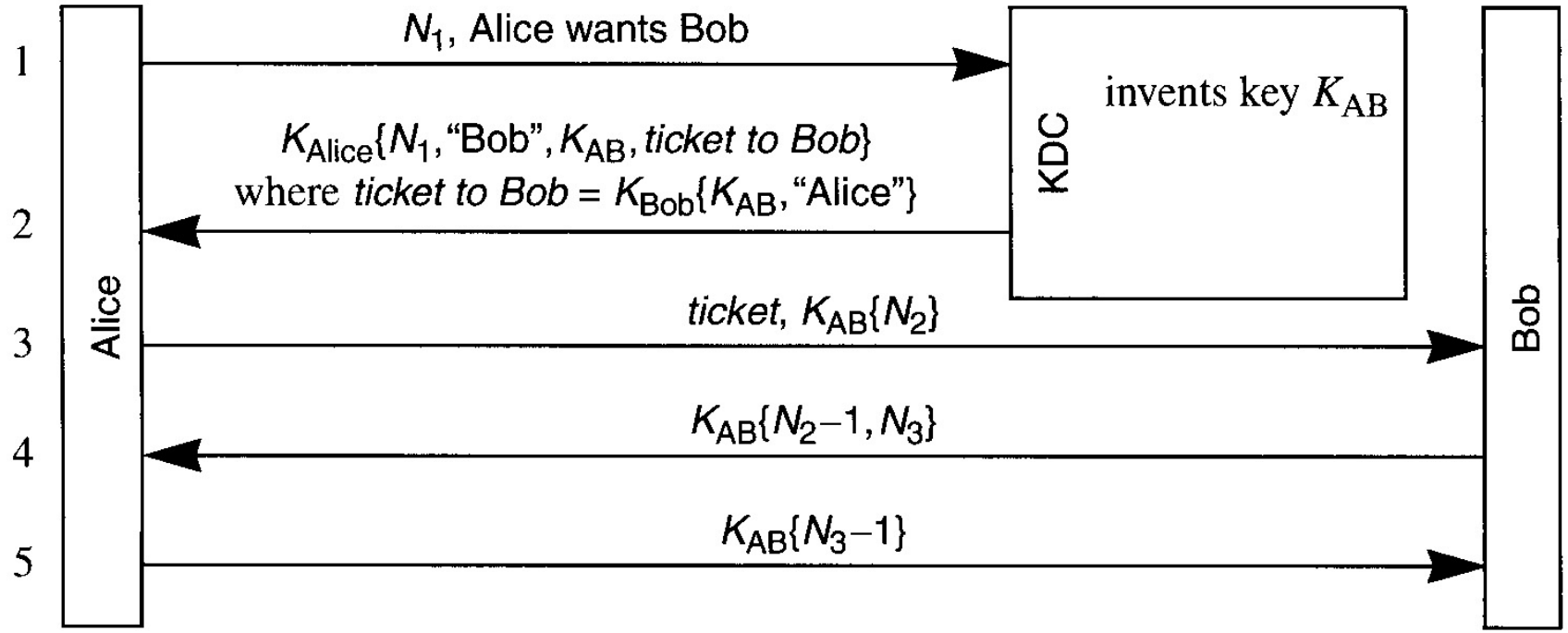
Key Distribution Center (KDC)

Q: How does KDC allow Bob, Alice to determine shared symmetric secret key to communicate with each other?

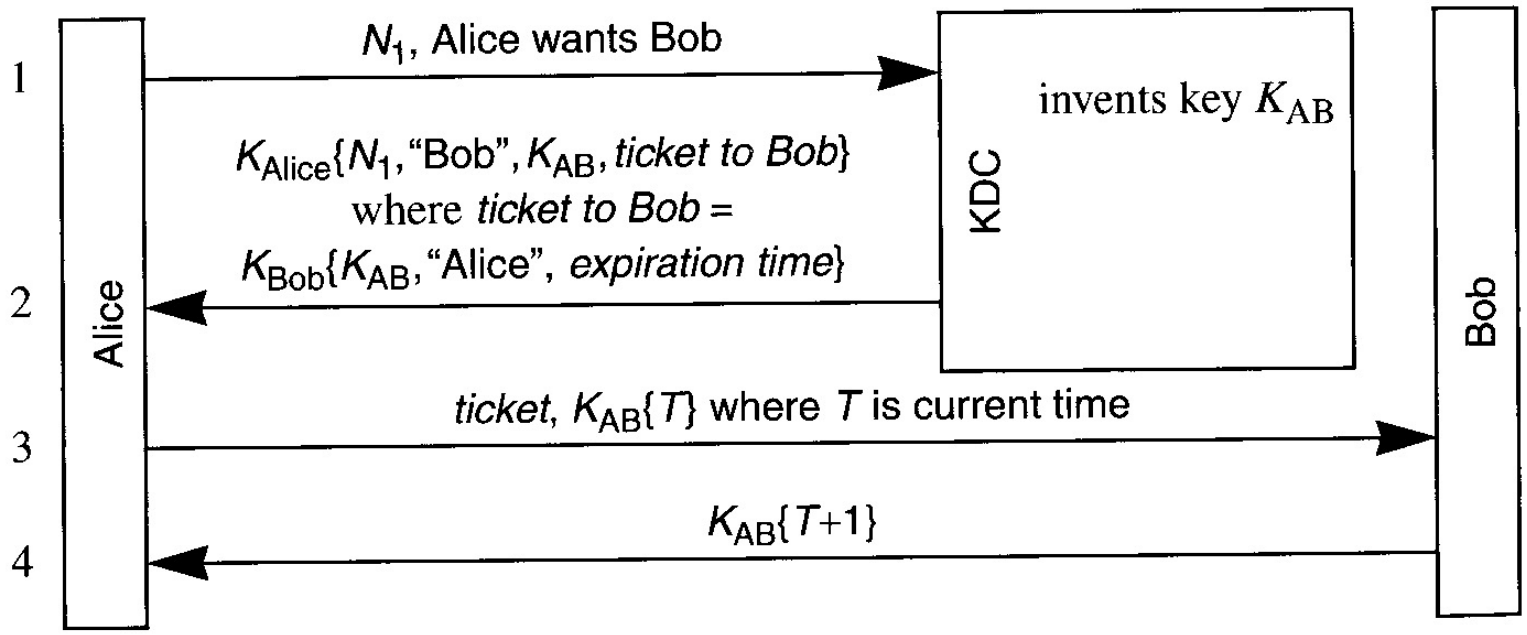


Alice and Bob communicate: using R1 as *session key* for shared symmetric encryption

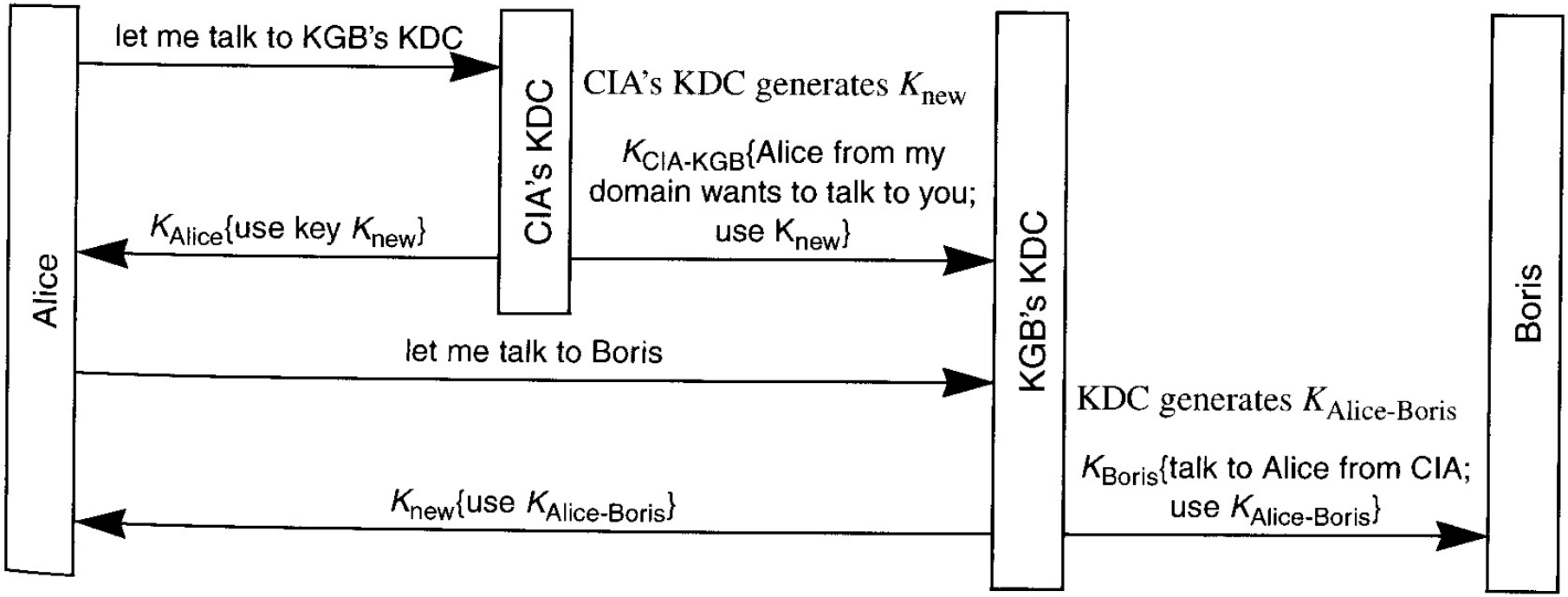
Needham-Schroeder Scheme



Simplified Secure Handshake for Kerberos (variant of the Needham-Schroeder scheme)



Inter-working between multiple KDC domains



KDC Vs. CA

KDC

- Participating entities need frequent contact with the KDC in order to initiate communications with each other
  - KDC needs to be online most of the time

  => KDC can be a single-point of failure, performance bottleneck, prime DDOS target
- KDC has enough information to impersonate anyone to anyone. If it is compromised, all network resources are vulnerably. Also, previously recorded conversations can be compromised
- Keys kept by the KDC are important secret ; have to be stored securely and protected from public view

CA

- Compromised CA cannot decrypted messages (without first impersonating one of the users) ; only active attacks can be mounted using CA's private key
- Certificates are not security sensitive, they can be stored in public database and transmitted over public network ;
- Do not need to be online => less vulnerable for network attack ; (except the support of CRL, which can be delegated to other public directory server) ; if a CA crashed, no new users can be added to the community by the existing members still can talk to each other with their existing public/private key-pairs

Kerberos

What Is Kerberos?

- Provides cryptographic authentication server(s) to authenticate users to servers and servers to users in a network environment.
  - Enable secure access control of networked resources
  - Relieve users/administrators the burden of managing potentially many accounts and passwords
- Relies on conventional encryption, making no use of public-key encryption
- Pioneered work by an MIT professor
- Two versions: version 4 and 5

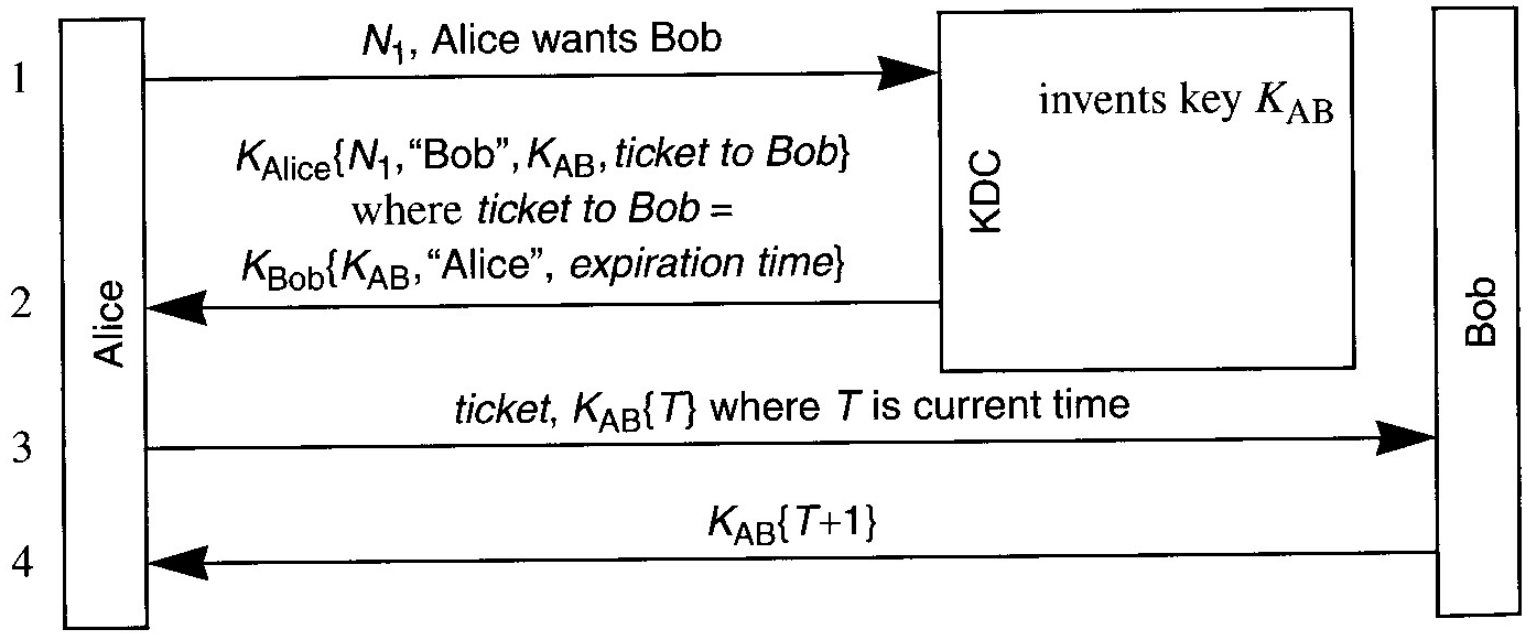
Kerberos

- Users wish to access services on servers.
- Three threats exist:
  - User pretend to be another user.
  - User alter the network address of a workstation.
  - User eavesdrop on exchanges and use a replay attack.

Where To Start...

- Every principal has a master (secret) key
  - Human user's master key is derived from passwd
  - Other resources must have their keys configured
- Every principal is registered with the Kerberos server, i.e. KDC
- All principals' master keys are stored in the KDC database, encrypted using the KDC master key

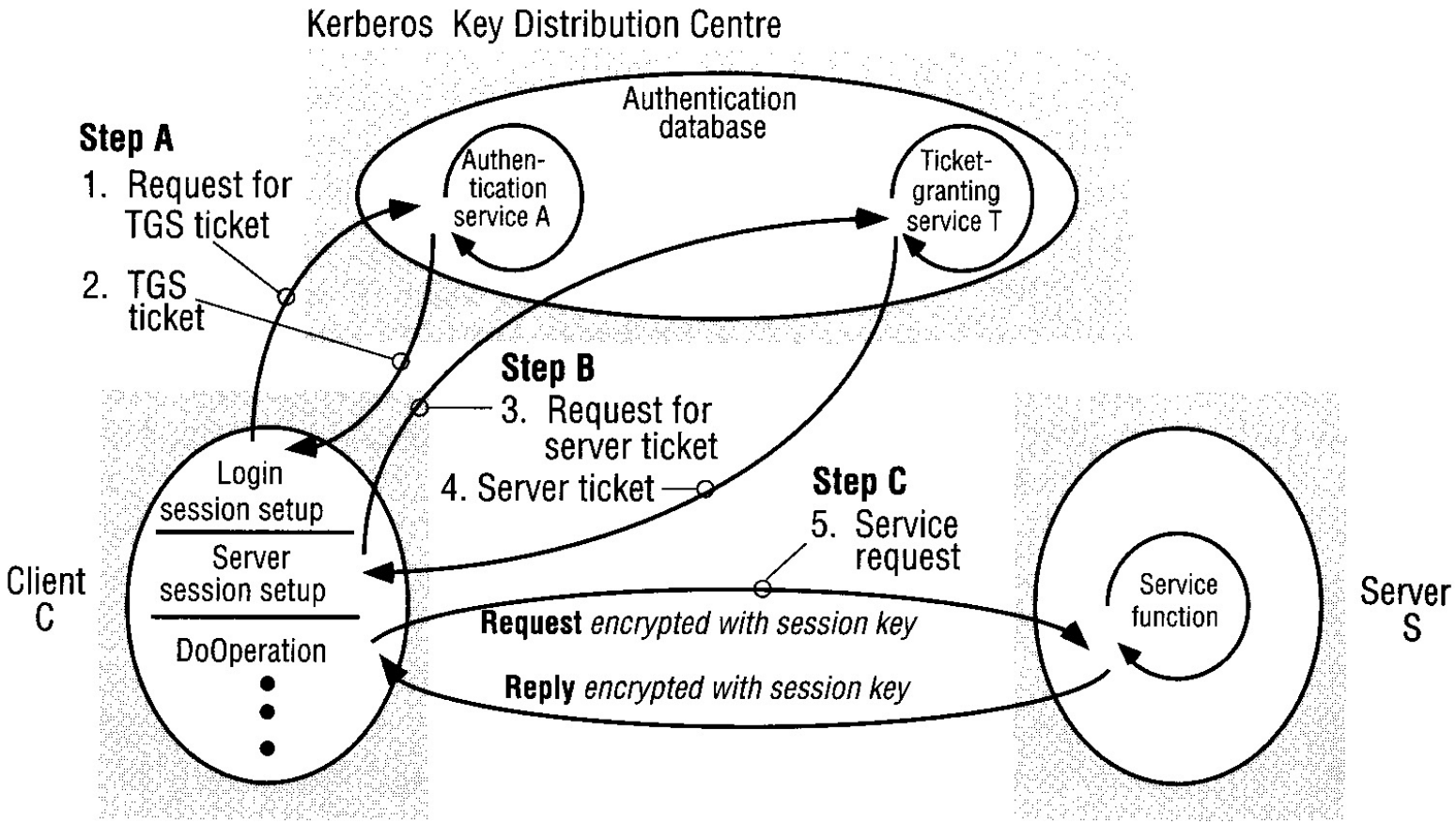
Simplified Secure Handshake for Kerberos (variant of the Needham-Schroeder scheme)



Session Key and Ticket-granting Ticket (TGT)

- Messages between a host and the KDC can be protected using the principal's master key
- For every request to KDC from the principal:
  - Insists on principal retyping in the password
  - Remember the principal's password
  - Remember the principal's master key derived from the password
- All options are equally inadequate!

Kerberos (A clearer/ more abstract view)



Kerberos in more details
(Backup Slides)

Kerberos Deployment...

- KDCs are “physically” secured
- Kerberos libraries are distributed on all nodes with users, applications, and other Kerberos-controlled resources
- All Kerberos exchanges are protected against confidentiality and integrity attacks
- Kerberos-rized applications
  - telnet
  - rtools (rlogin, rcp, rsh)
  - Network file systems (NFS/AFS)

Where To Start...

- Every principal has a master (secret) key
  - Human user's master key is derived from passwd
  - Other resources must have their keys configured
- Every principal is registered with the Kerberos server, i.e. KDC
- All principals' master keys are stored in the KDC database, encrypted using the KDC master key

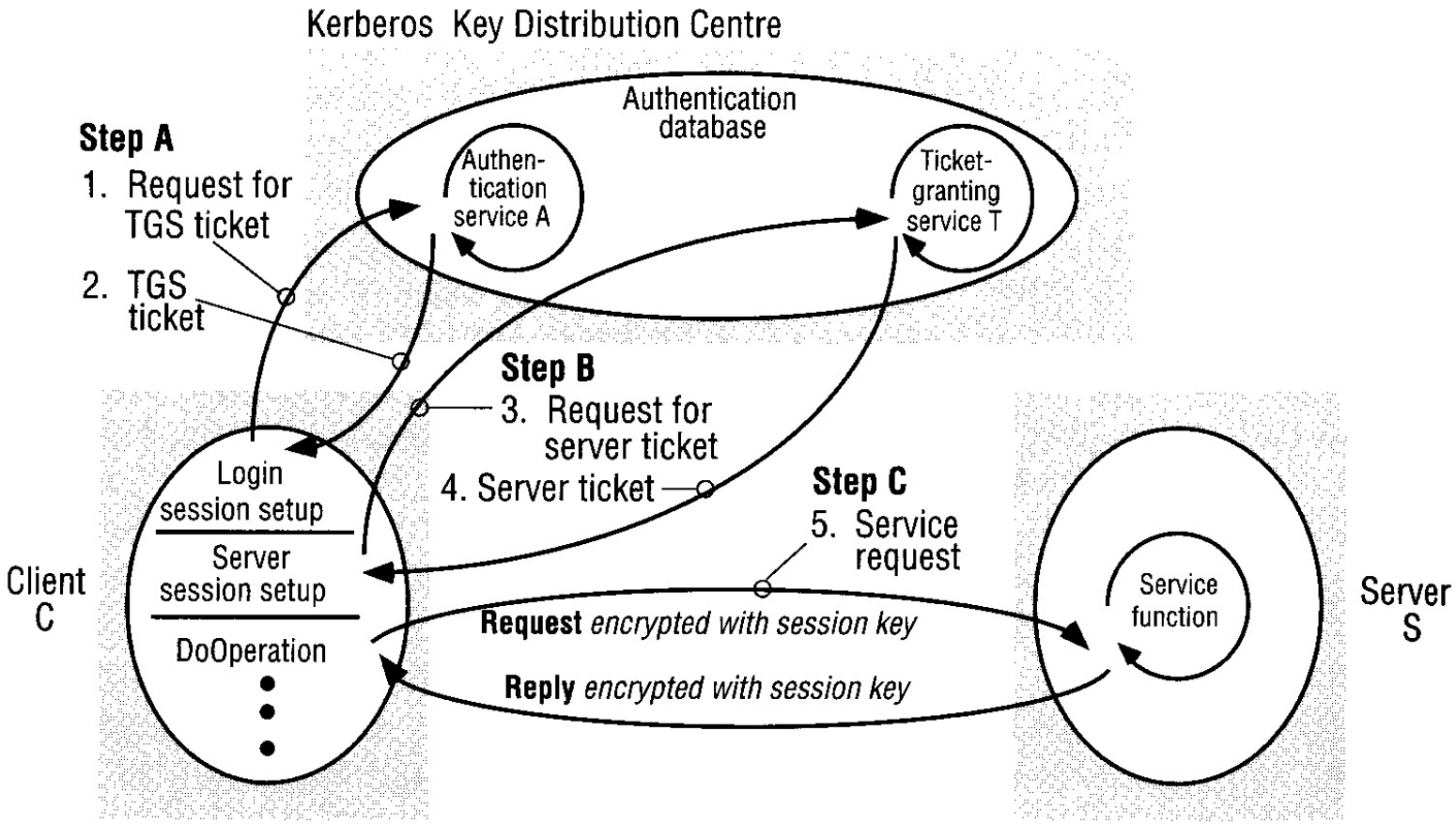
Tickets ...

- Every principal has a main shared secret with the KDC - principal's master key
- Any secure communication/access among principals must be "mediated" by KDC through *tickets*
- How would Alice talk to Bob?

Session Key and Ticket-granting Ticket (TGT)

- Messages between a host and the KDC can be protected using the principal's master key
- For every request to KDC from the principal:
  - Insists on principal retyping in the password
  - Remember the principal's password
  - Remember the principal's master key derived from the password
- All options are equally inadequate!

Kerberos (A clearer/ more abstract view)



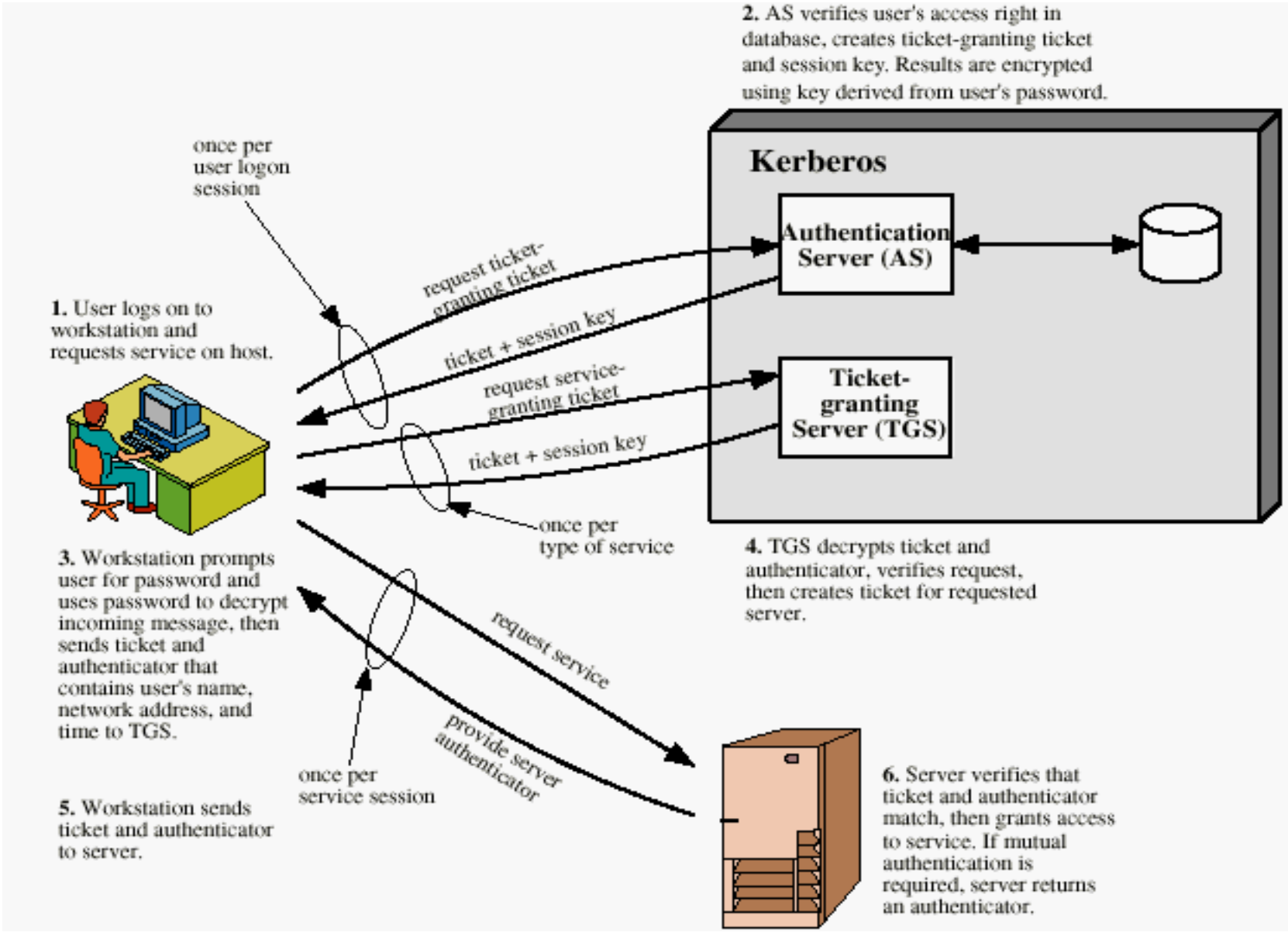
Session Key and TGT...

- To avoid potentially too much exposure to password/master key
  - At initial login, a per principal session key S_B (for Bob) is requested from KDC
  - S_B has a limited valid time period
  - A TGT (Ticket-Granting-Ticket) for Bob is also issued by the KDC, which includes the session key S_B and Bob's identification information, all encrypted using the KDC's master key

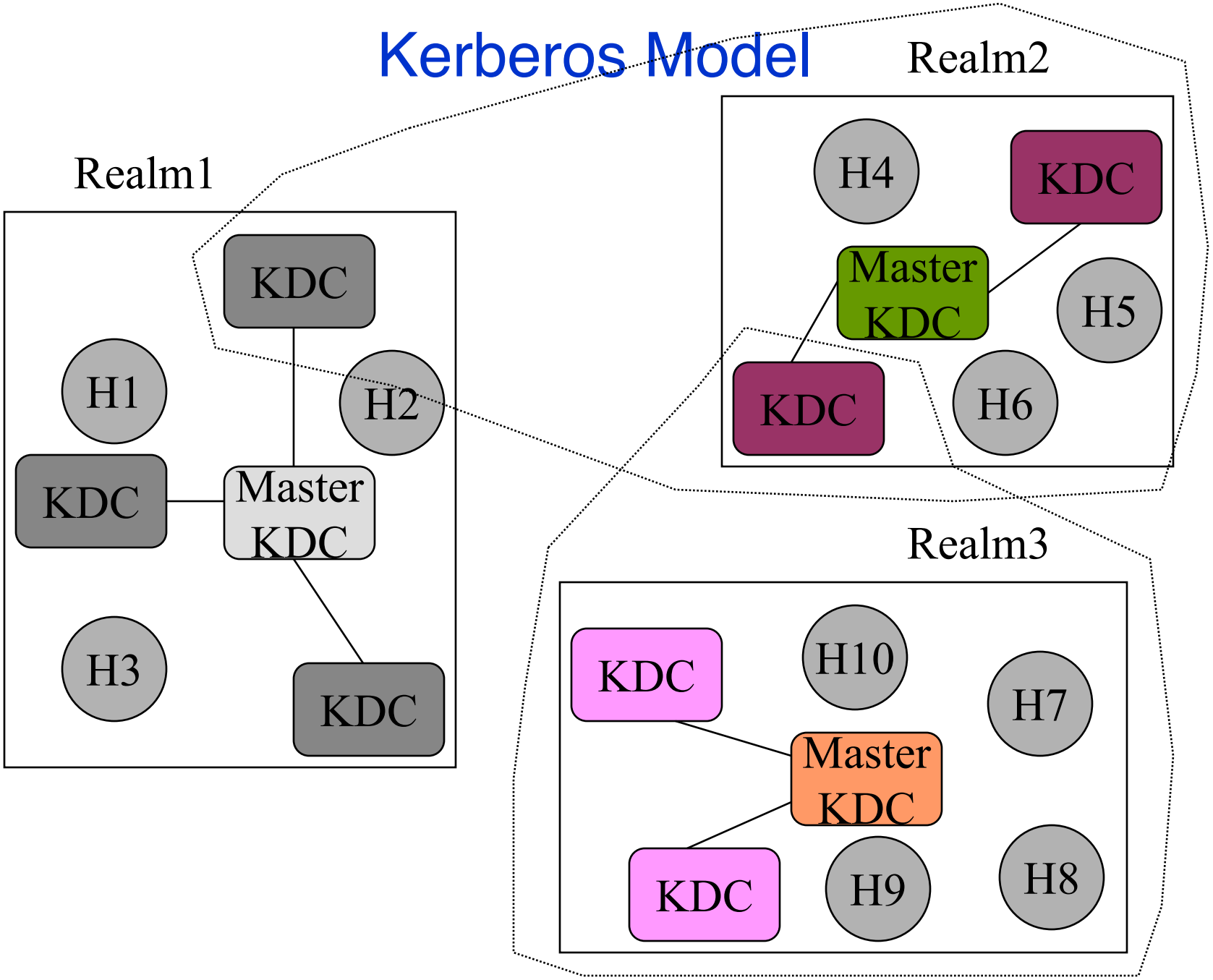
Session Key and TGT...

- Bob's Kerberos client decrypts and remembers:
  - S_B, for subsequent message with KDC
  - TGT, for reminding/convincing KDC to use S_B with it as well
  - No need for remembering/storing password
- New request to KDC must include TGT in the request message
- New tickets from KDC must be decrypted with S_B

Details of Kerberos Handshake



Kerberos Model



Replicated KDCs

- Multiple replica of KDC - availability and performance
- Keeping KDC databases consistent
  - Single master KDC as the point of direct update to principals' database entries
  - Updated database is downloaded from the master to all replica KDCs
  - Periodic download or on-demand

Will It Be Effective?

- KDC dynamic state consists of outstanding TGTs and tickets.
- Kerberos puts the burden of “maintaining” them on the clients - hosts/servers/grantees.
  - Convince me that I did this for you...
- KDC is only involved in the initial “mediation” and it stays “out of the picture” once a ticket is issued.
- Only static state information is principals’ database - read only for all replica KDCs.

Database Content Protection

- Encryption is required for sensitive data
- Integrity of the database must be ensured
  - Installation of masqueraded master keys
  - Substitution (replay) of old databases
- Kerberos stores principals' master keys encrypted with KDC master key
- Kerberos transmits a secure hash of the database with encryption, in a separate message during downloads

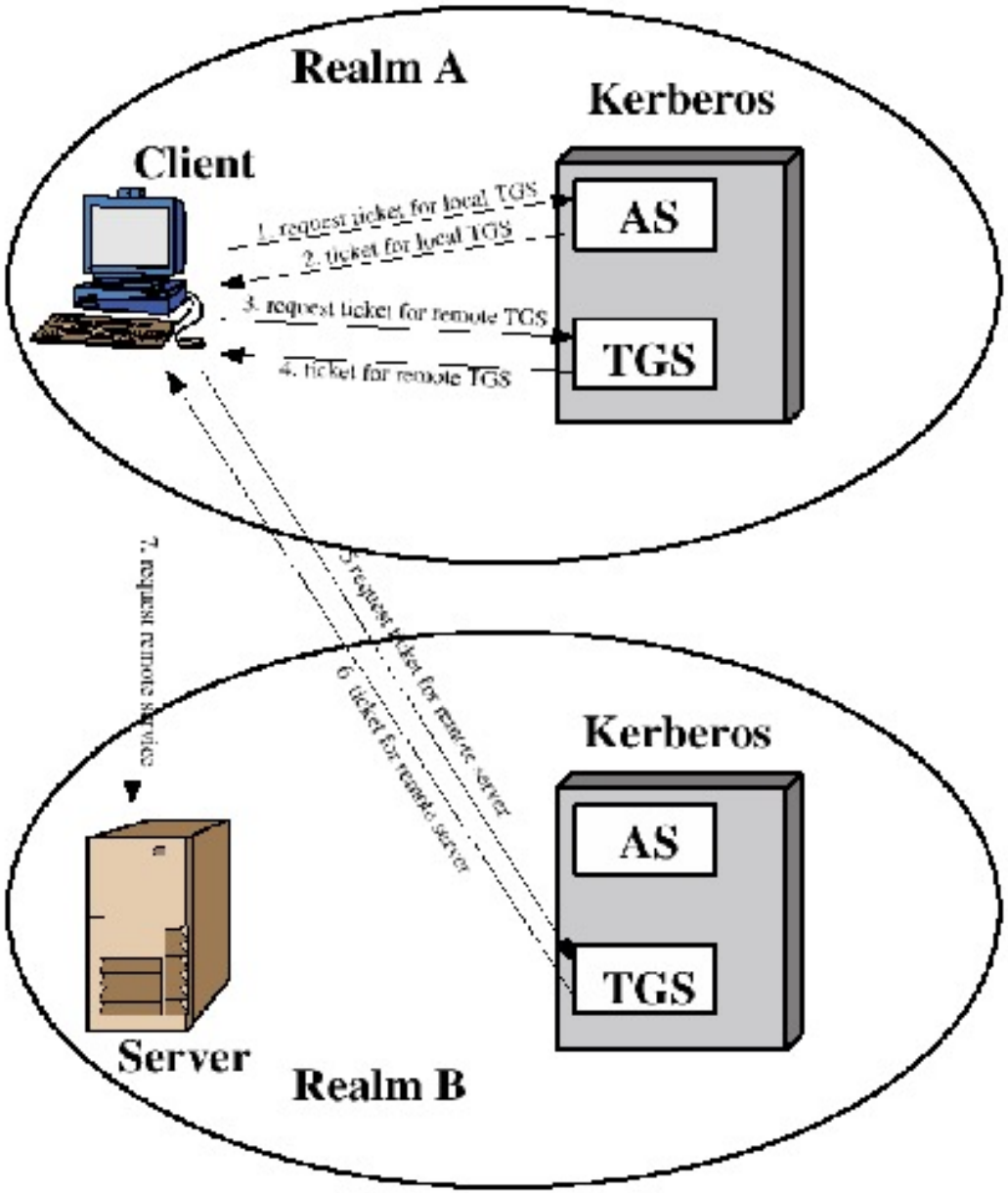
Multiple Trust Domains

- Single master KDC can only stretch so far...
- KDC asks people to put too much trust in it.
  - Should competing commercial entities use the same KDC?
  - .gov, .org, .edu etc, each having a different model of “what is more trustworthy.”
- Single master KDC - greatest temptation - biggest security risk/vulnerability.
- So comes different domains or *realms.*

Kerberos Realms

- Each realm has a different master KDC, with different master KDC key
- Each realm can have many replica KDCs, but all sharing the same KDC master key
- Two KDCs in different realms have different principals' master key databases

Request for Service in Another Realm



What's new in Version 5?

- Can use encryption algorithms other than DES (only choice in V4)
- Can support protocol other than the Internet Protocol (IP)
- A more standardized coding format for the messages (based on ASN.1)
- Support "Delegation" aka "Authentication forwarding", i.e. , a client can transfer a ticket it obtained to a different host/client and let the latter to access the resource authorized by the ticket
  - e.g. an end-user asks the print-server (program) to fetch her file from the file-server to print
- Support more scalable Inter-Realms (or multiple KDC domains) authentication
  - V4 requires N^2 direct KDC-to-KDC trust relationships ; V5 allows one to go thru intermediate KDC(s) to get a ticket from a third Realm without letting the intermediate KDC to commit impersonation
- Arbitrary Ticket Lifetime (V4 limit ticket-lifetime to 21 hrs)
  - Need to repeatedly asked for password ; not good for long-running batch job like simulation
- Techniques to evade password-guessing attacks
  - Need to enter your password before receiving a ticket-granting challenge

and more…

Kerberos - in practice

- Kerberos V5 is an Internet standard, specified in RFC4120, 6113,6251
- **To use Kerberos:**
  - need to have a KDC on your network
  - need to have Kerberised applications running on all participating systems
  - major problem before year 2000 - US export restrictions
    - Kerberos cannot be directly distributed outside the US in source format (& binary versions must obscure crypto routine entry points and have no encryption)
    - else crypto-libraries must be re-implemented locally
- In Microsoft Windows 2000, authentication information is done via “Active Directory” which serves as both a KDC and directory service. The protocol is a modified version of Kerberos V5, (documented Iin RFC3244, 4757) e.g.
  - Kerberos was designed to authenticate based on names of communicating principals while and Operating systems typically authenticate based on userid (UID) and group-id

Want to learn more about Kerberos ?

- http://web.mit.edu/kerberos/www/dialogue.html
- **The Kerberos Consortium**
  - **http://www.kerberos.org/**