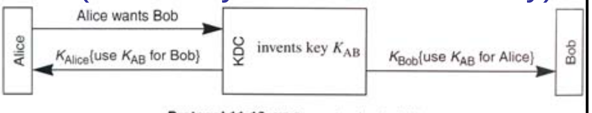
SE-4920: Lecture 13 Kerberos V4

- Reading
  - Chapters 13
- Today's Outcomes
  - Describe the services provided by Kerberos V4
  - Diagram the generation and use of tickets and ticket-granting tickets for authentication and establishment of a shared secret

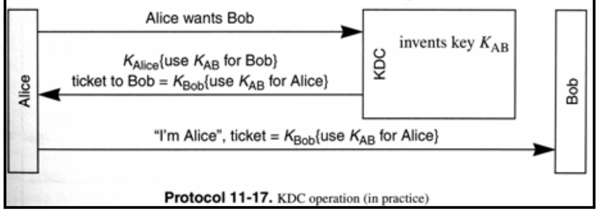
Introduction

- Secret key based service for providing authentication of a user
  - At a workstation
  - To various network resources
  - Uses KDC
- Fourth and fifth version in current use
  - V4 – published in late 1980s
    - More widely used, simpler, has better performance
    - TCP/IP only
  - V5 – published in 1993, revised in 2005, overcame many V4 limitations
- Library of subroutines for application use
  - MIT Kerberos (V4 and V5), KTH Kerberos (V4), Heimdal (V5)
- Software using Kerberos
  - Win2k, XP, Server 2003 use a variant for authentication
  - Mac OS X
  - OpenSSH, NFS, PAM (pam_krb5), SOCKS, Apache (mod_auth_kerb)

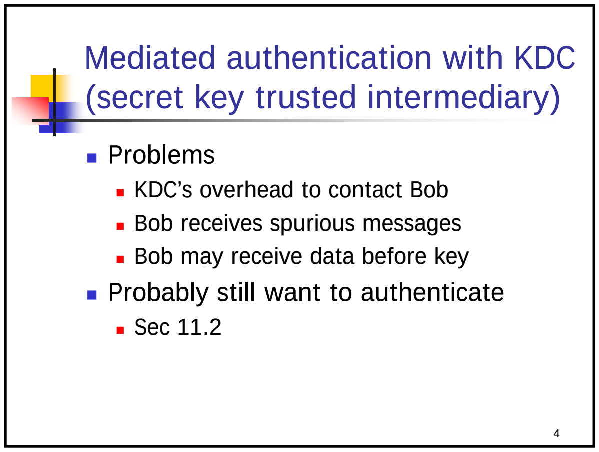
Mediated authentication with KDC (secret key trusted intermediary)


Alice
Alice wants Bob
K_{Alice}\{use K_{AB} for Bob\}
KDC
invents key K_{AB}
K_{Bob}\{use K_{AB} for Alice\}
Bob

Protocol 11-16. KDC operation (in principle)


Alice wants Bob
Alice
K_{Alice}\{use K_{AB} for Bob\}
ticket to Bob = K_{Bob}\{use K_{AB} for Alice\}
KDC
invents key K_{AB}
Bob
"I'm Alice", ticket = K_{Bob}\{use K_{AB} for Alice\}

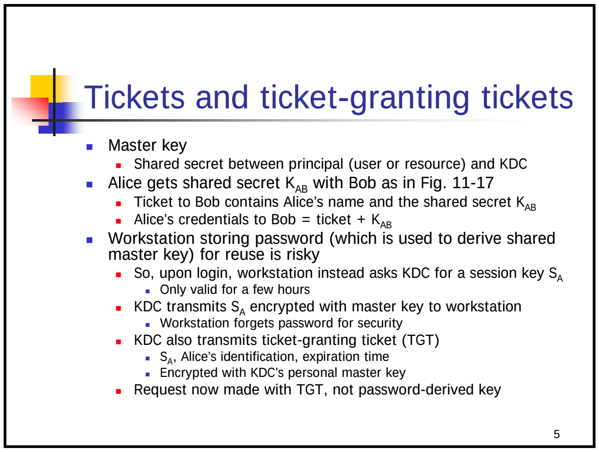
Protocol 11-17. KDC operation (in practice)



Mediated authentication with KDC (secret key trusted intermediary)

- Problems
  - KDC's overhead to contact Bob
  - Bob receives spurious messages
  - Bob may receive data before key
- Probably still want to authenticate
  - Sec 11.2

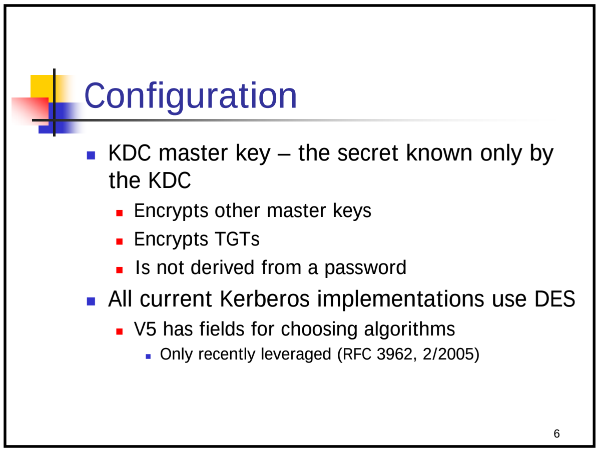
4



Tickets and ticket-granting tickets

- Master key
  - Shared secret between principal (user or resource) and KDC
- Alice gets shared secret K_{AB} with Bob as in Fig. 11-17
  - Ticket to Bob contains Alice's name and the shared secret K_{AB}
  - Alice's credentials to Bob = ticket + K_{AB}
- Workstation storing password (which is used to derive shared master key) for reuse is risky
  - So, upon login, workstation instead asks KDC for a session key S_A
    - Only valid for a few hours
  - KDC transmits S_A encrypted with master key to workstation
    - Workstation forgets password for security
  - KDC also transmits ticket-granting ticket (TGT)
    - S_A, Alice's identification, expiration time
    - Encrypted with KDC's personal master key
  - Request now made with TGT, not password-derived key

5



Configuration

- KDC master key – the secret known only by the KDC
  - Encrypts other master keys
  - Encrypts TGTs
  - Is not derived from a password
- All current Kerberos implementations use DES
  - V5 has fields for choosing algorithms
    - Only recently leveraged (RFC 3962, 2/2005)

6

Logging into the network: Obtaining a session key and TGT

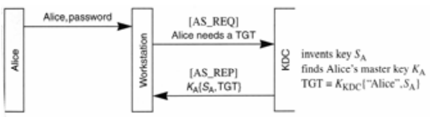


Figure 13-1. Obtaining a TGT

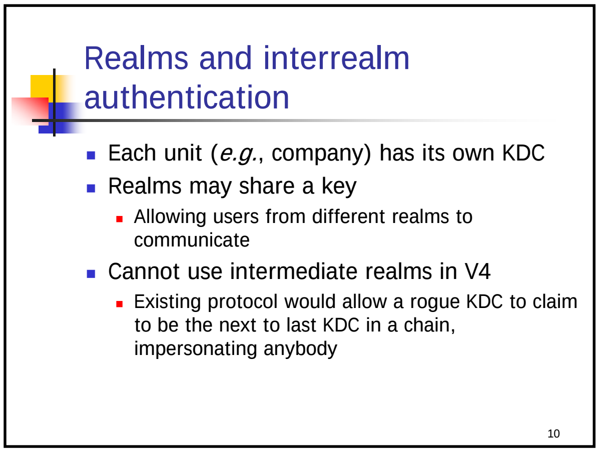
- TGT is doubly encrypted
  - Suboptimal (TGT only of use if you know S_A)
- V4 does not ask for password until server responds
  - Keep password in memory for less time
    - But, gives a quantity for offline guessing
  - V5 has method to prove knowledge of password first
- Big advantage: Everything at KDC is static
  - It can forget that it issued the ticket
    - Everything needed later is in the TGT

Logging into the network: Alice asks to talk to a remote node

- Alice's request contains
  - TGT
  - Name of resource
  - Authenticator
    - Proof that S_A is known
    - Encrypted time of day (max. skew typ. 5 minutes)
- Server response (encrypted with S_A) has
  - Ticket to Bob = $K_B\{$"Alice",$K_{AB}\}$
  - K_{AB}
- Authenticator not really needed
  - Response not useful if you do not know S_A
  - Done for similarity to rest of Kerberos
    - Where authenticator prevents replay attack
- Alice's request to Bob: Ticket + Authenticator
- Bob's action
  - Verify authenticator (skew and duplication within skew)
  - Reply with $K_{AB}\{$timestamp+1$\}$ (provides mutual authentication)
- We now have mutual authentication and a shared secret
  - May proceed in the clear, add integrity, or integrity and encryption

Replicated KDCs

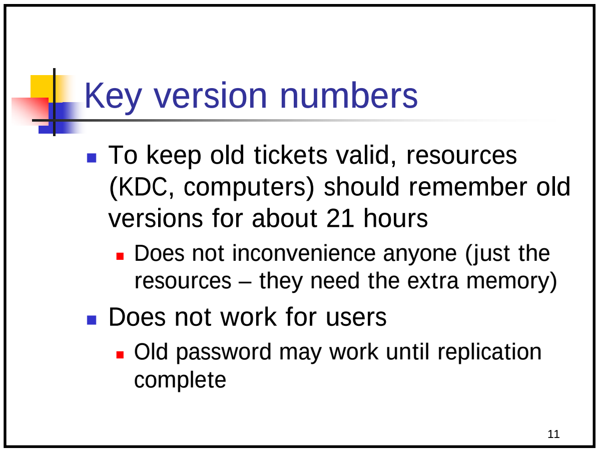
- Avoid single point of failure
- One master copy
  - Target of all updates
    - Add, modify, delete users (nothing else)
    - Failure of master prevents only these operations
  - Updates fairly rare
- Several replicas share load
- Replication
  - Principal master keys encrypted with KDC master key; no further encryption
  - Attacker can still see usernames
  - Cryptographic hash used for integrity
    - *E.g.*, swap my key with yours so I can access your account
  - Also includes timestamp, preventing replay of an old database
    - *E.g.*, before you were fired



Realms and interrealm authentication

- Each unit (*e.g.*, company) has its own KDC
- Realms may share a key
  - Allowing users from different realms to communicate
- Cannot use intermediate realms in V4
  - Existing protocol would allow a rogue KDC to claim to be the next to last KDC in a chain, impersonating anybody

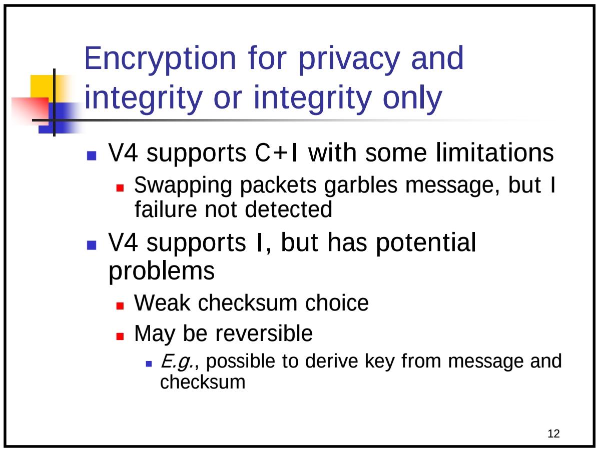
10



Key version numbers

- To keep old tickets valid, resources (KDC, computers) should remember old versions for about 21 hours
  - Does not inconvenience anyone (just the resources – they need the extra memory)
- Does not work for users
  - Old password may work until replication complete

11



Encryption for privacy and integrity or integrity only

- V4 supports C+I with some limitations
  - Swapping packets garbles message, but I failure not detected
- V4 supports I, but has potential problems
  - Weak checksum choice
  - May be reversible
    - *E.g.*, possible to derive key from message and checksum

12