Network File System

Michael Tsai
2020/5/25

Network File Services

- Goal:
  - Share filesystems among computers
  - Transparent to users
    (as if the files is stored locally)
  - No information is lost when the server crashes
- Get better in the last 25 years
  (very complex, bugs in unusual situations)

Related Terms

- Storage Area Network (SAN): serve blocks (not files)
  - Example: iSCSI
- Network Attached Storage (NAS):
  file-level data storage attached to the network
- Server Message Block / Common Internet File System (SMB/CIFS)
  - Microsoft Windows Network —> Active Directory
  - Samba: Implementation on Unix-like systems
- Apple Filing Protocol (AFP)

State

- State:
  e.g., files that each client has open.
- State**less**: if the server does not track the status of the users & files
- Stateful: otherwise.
- Stateful is more complex, but allow more control over files and their management.

Performance Concerns

- Goal:
  transparent == no difference from local access
- Challenge: network latency (over WAN)
- Techniques:
  - Read-ahead caching:
    preload the file into local memory
  - Batch writes
    cache writes in memory and send them in batches

NFS history

- NFS v2:
  client write op. is completed until ack from the server (read: HIGH DELAY!), UDP only

- NFS v3:
  asynchronous writes, TCP or UDP
  (NFS v2 should not be used now)

- NFS v4: numerous enhancements, TCP only! (congestion control)
  Compatibility with firewall & NAT devices
  Lock & mount —> core NFS protocol
  ACLs
  Unicode filenames
  Good performance on low-bandwidth connections

- NFS FAQ: http://nfs.sourceforge.net
  Read A1, A4, A6. Google for the terms you don't understand and try to learn these.

Security

- Bottom line: allow files to be accessed (read/write) **from the network** (read: entire world)!
- Access control
- How to authenticate the users?
- Modern solution: centralized authentication system

NFS Security

- Flavors of authentication:

  1. AUTH_NONE: no authentication

  2. AUTH_SYS: UNIX-style user & group access control

  3. RPCSEC_GSS: a powerful flavor that ensures integrity and privacy plus authentication

NFS Security

- AUTH_SYS: how to attack?
  - Take control of a client machine that is allowed to access the NFS
  - **Pretend** that a user that owns the file is authenticated, and tell NFS server to serve the file to the client machine

Root access

- In NFS-mounted filesystems, usually root is changed to run as nobody (or other similar account)
- Prevent files owned by root to be accessed by the world
- But cannot protect user files
  (since root of the client can su to become other user)

Server configuration

- /etc/exports: specify the directories to share
- Each line: <path(local)> <client>(options)
  Example:
  /mnt/backup 140.112.31.40(rw,async)
- See manpage exports(5) (man 5 exports)
- Possible client ID: IP (192.168.1.1), IP networks (192.168.1.0/24), hostnames (www.csie.ntu.edu.tw), wildcards (*.csie.ntu.edu.tw) — hostname determined via reverse DNS lookup
- Common options: ro/rw, sync/async, root_squash/no_root_squash/all_squash
- Package: nfs-kernel-server

Client configuration

- Example:
  mount nfs.csie.ntu.edu.tw:/mnt/backup /mnt/backup_nfs_csie
- Put it in /etc/fstab for mounting at start-up
- What's the downside of putting it in fstab?
- Package: nfs-common

Why Automount

- Downsides of putting NFS filesystems in fstab:

    1. Maintaining fstab file on 100+ machines is tedious (different needs on different machines)

    2. Dependency on multiple servers (hung commands)

    3. No backup provisions

Automount

- Mounts a virtual filesystem at locations for automatic mounting
- Mount the filesystems “on-demand”
- Unmount when a filesystem is not used for a time duration (time-out)
- Check out packages: autofs, autofs5