AppleShare: Every node is both a server with a disk and a client.

- LAN:
- Client: One server node with all the disks, and a bunch of diskless workstations on a

Examples:
- share disks between nodes as if all the disks were attached to every node.
- given a set of disks attached to different nodes

Basic Idea:

One of the most common uses of distributed systems

Today: Distributed File Systems

OS manages the communication

To use the server, the client does a procedure call

Servers export procedures for some set of clients to call

Last Class: Distributed Systems and RPCs
In modern operating systems, many have location transparency. Most naming schemes used in practice do not have location transparency. File's storage location changes.

**Location independence:** The name of the file need not change if the storage location.

**Location transparency:** The name of the file does not reveal the physical location.

- Do the names change if the user moves?
- Do the names change if the file moves?
- Do the names reveal the file's location?
- How are file names?

### Issues

#### Naming and Transparency

#### Replication

Server with or without

#### Caching

Remote file access

#### Naming and Transparency

### Distributed File Systems: Issues

---

Department of Computer Science, UMass Amherst
Disadvantages:

Advantages:

Examples: AppleShare, Win NT

Absolute names: <machine name: path name>

Naming Strategies: Absolute Names

Mount Points (NFS – SUN’s Network File System)
Partial contents of /etc/ passwd for Edlab machines:

```
```

NFS: Example

```
```

NFS: Example

```
```

```
What happens if multiple clients cache the same file?

When are modifications propagated back to the remote file?

Where and when are the blocks cached?

**Caching Issues:**

1. Remotely on the server machine and then return the results using RPC.
2. Can transfer the file (or part of the file) to the requesting host, and perform local access (called caching).
3. Once the user specifies a remote file, the OS can do the access either:

**Remote File Access and Caching**

**Disadvantages:**

**Advantages:**

- Single name space: CMU’s Andrew and Berkeley’s Sprite
- No matter which node you are on, the file names are the same.
- Set of workstation clients, and a set of dedicated file server machines.
- When a client starts up, it gets its file name structure from a server.
- As users access files, the server sends copies to the workstation and the workstation caches the files.

**Naming Strategies: Global Name Space**
Write back: Yields the quickest response time since the write need only hit cache before the process continues.

White back: Yields the quickest response time since the write need only
exploits caching only for reads.

Performance:

When to write local changes to the server has a central role in determining distributed file system performance.

Remote File Access and Caching

Advantages:

1. Local disk
2. Local memory

Disadvantages:

•
name must be in /etc/exports on the server.
1. /etc/exports lists the global names that the local nodes import. A corresponding global
2. /etc/exports lists the local names the server is willing to export.

- Uses a mount protocol to make a global name local
- Servers have no state.
- Nodes are both servers and clients.
- NFS is designed to run on LANs
- NFS is the standard for distributing UNIX file access.

### Case Study: Sun’s Network File System

- Server-initiated consistency: Server detects potential conflicts and invalidates caches
- Client-initiated consistency: Client contacts the server and asks if its copy is consistent with the server’s copy.
- Can check only upon opening a file.
- Can check at a given interval.
- Can check every access.

- Server needs to know:
  * Which clients are readers and which are writers.
  * Which clients have cached which parts of which files.
NFS Implementation

What they logon:
- Users may need to know different names depending upon the node to

? Support the NFS mount and remote access protocols using RPC.
- Does not rely on node homogeneity - heterogeneous nodes must simply

4. Reading/Writing files
3. Accessing file attributes
2. Manipulating links and directories
1. Directory search, reading directory entries

NFS defines a set of RPC operations for remote file access:

NFS Implementation
Need to write changes back to disk

- Speed up remote file access with caching
- Location dependent names are most prevalent
- Desired name independence, but it is difficult to attain

Naming

Summary