

OPERATING SYSTEMS

File Management

Suresh Jamadagni

Department of Computer Science

OPERATING SYSTEMS

File System – File-System Mounting, File Sharing and File Protection

Suresh Jamadagni

Department of Computer Science

OPERATING SYSTEMS

Slides Credits for all the PPTs of this course

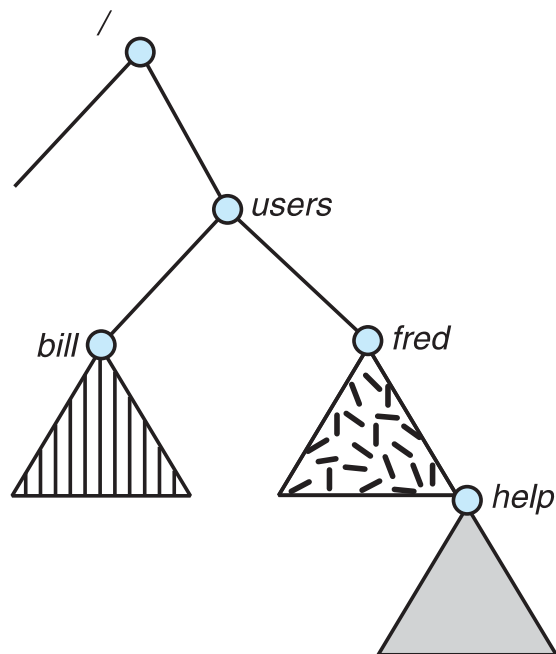


- The slides/diagrams in this course are an **adaptation, combination,** and **enhancement** of material from the following resources and persons:
 1. Slides of Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne - 9th edition 2013 and some slides from 10th edition 2018
 2. Some conceptual text and diagram from Operating Systems - Internals and Design Principles, William Stallings, 9th edition 2018
 3. Some presentation transcripts from A. Frank – P. Weisberg
 4. Some conceptual text from Operating Systems: Three Easy Pieces, Remzi Arpaci-Dusseau, Andrea Arpaci Dusseau

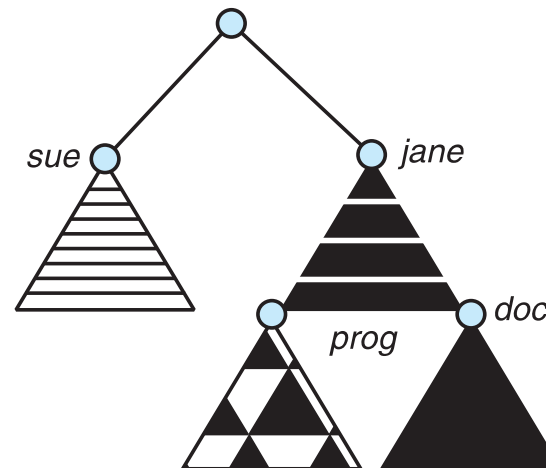
OPERATING SYSTEMS

File System Mounting

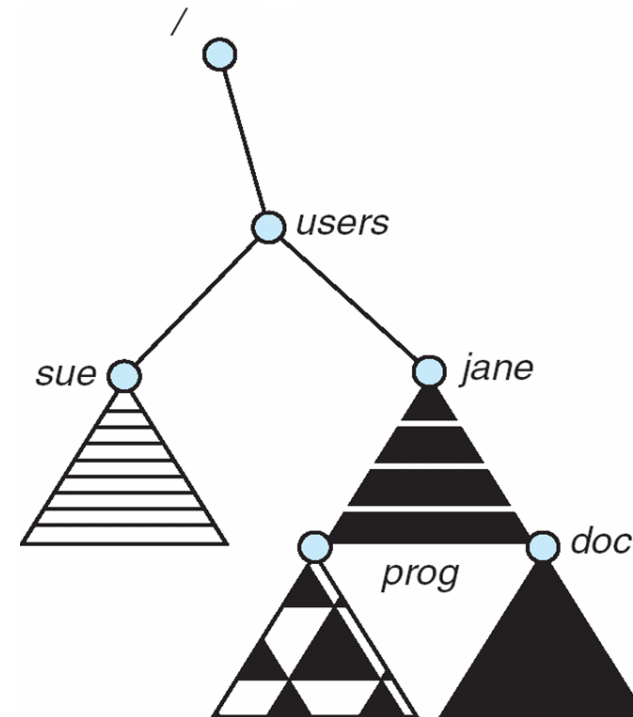
- ❑ A file system must be **mounted** before it can be accessed
- ❑ Fig (a) shows Existing File System
- ❑ A unmounted file system (i.e., Fig. (b)) is mounted at a **mount point**
- ❑ Fig (c) shows the effect of mounting



(a)



(b)



(c)

- ❑ Sharing of files on multi-user systems is desirable
- ❑ Sharing may be done through a **protection** scheme
- ❑ On distributed systems, files may be shared across a network
- ❑ Network File System (NFS) is a common distributed file-sharing method
- ❑ If multi-user system
 - ❑ **User IDs** identify users, allowing permissions and protections to be per-user
 - ❑ **Group IDs** allow users to be in groups, permitting group access rights
 - ❑ Owner of a file / directory
 - ❑ Group of a file / directory

OPERATING SYSTEMS

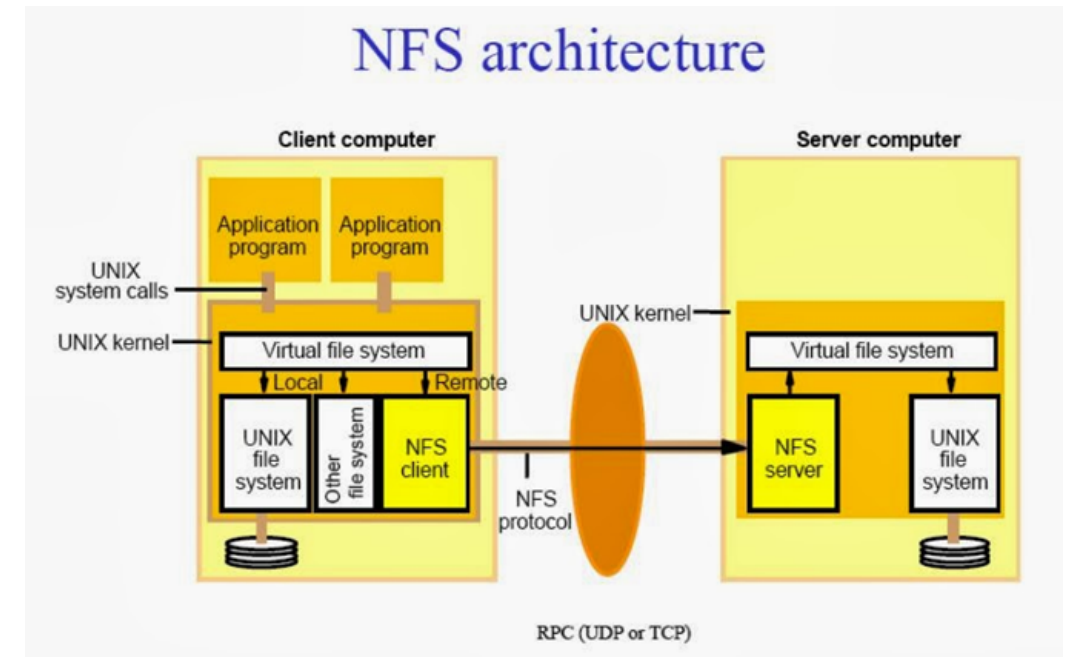
File Sharing – Remote File Systems

- ❑ Uses networking to allow file system access between systems
 - ❑ Manually via programs like FTP
 - ❑ Automatically, seamlessly using **distributed file systems**
 - ❑ Semi automatically via the **world wide web**

OPERATING SYSTEMS

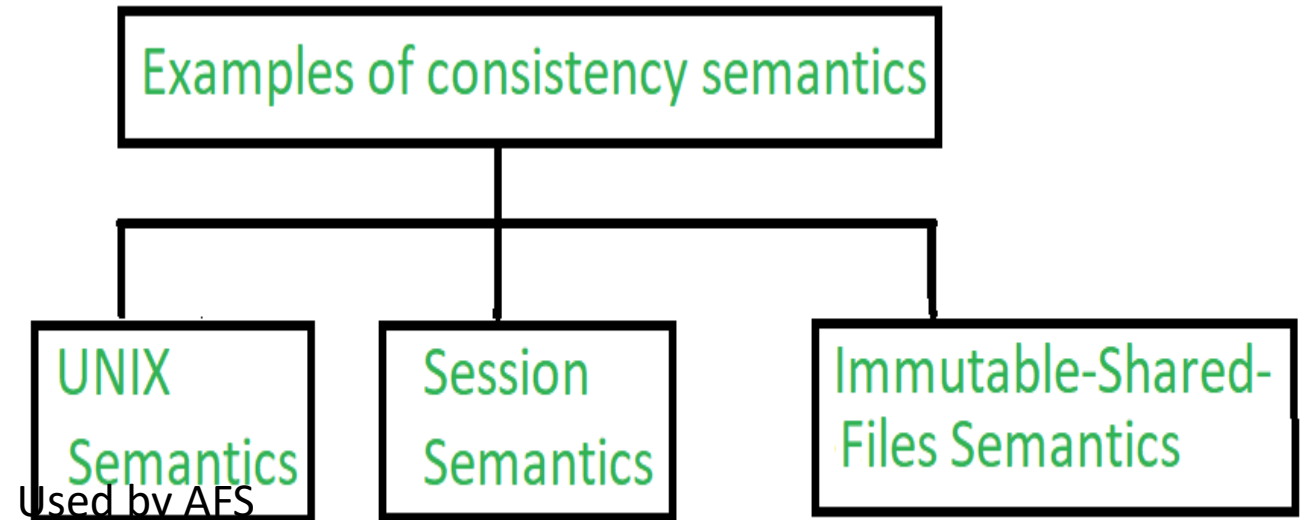
File Sharing – Remote File Systems (Cont.)

- ❑ **Client-server** model allows clients to mount remote file systems from servers
 - ❑ Server can serve multiple clients
 - ❑ Client and user-on-client identification is insecure or complicated
 - ❑ **NFS** is standard UNIX client-server file sharing protocol
 - ❑ **CIFS** is standard Windows protocol
 - ❑ Standard operating system file calls are translated into remote calls
- ❑ Distributed Information Systems (**distributed naming services**) such as LDAP, DNS, NIS, Active Directory implement unified access to information needed for remote computing



- ❑ All file systems have failure modes
 - ❑ For example corruption of directory structures or other non-user data, called **metadata**
- ❑ Remote file systems add new failure modes, due to network failure, server failure
- ❑ Recovery from failure can involve **state information** about status of each remote request
- ❑ **Stateless** protocols such as NFS v3 include all information in each request, allowing easy recovery but less security

- **Consistency Semantics** is a concept which is used by users to check file systems which are supporting file sharing in their systems.
- Basically, it is a specification to check how in a single system multiple users are getting access to same file and at same time.
 - like when will modification by some user in some file is noticeable to others.



- ❑ Specify how multiple users are to access a shared file simultaneously
 - ❑ Similar to process synchronization algorithms
 - ▶ Tend to be less complex due to disk I/O and network latency (for remote file systems)
 - ❑ Andrew File System (AFS) implemented complex remote file sharing semantics
 - ❑ Unix file system (UFS) implements:
 - ▶ Writes to an open file visible immediately to other users of the same open file
 - ▶ Sharing file pointer to allow multiple users to read and write concurrently
 - ❑ AFS has session semantics
 - ▶ Writes only visible to sessions starting after the file is closed

? File owner/creator should be able to control:

? what can be done

? by whom

? Types of access

? **Read**

? **Write**

? **Execute**

? **Append**

? **Delete**

? **List**

? Mode of access: read, write, execute

? Three classes of users on Unix / Linux

			RWX
a) owner access	7	⇒	1 1 1
			RWX
b) group access	6	⇒	1 1 0
			RWX
c) public access	1	⇒	0 0 1

? Ask manager to create a group (unique name), say G, and add some users to the group.

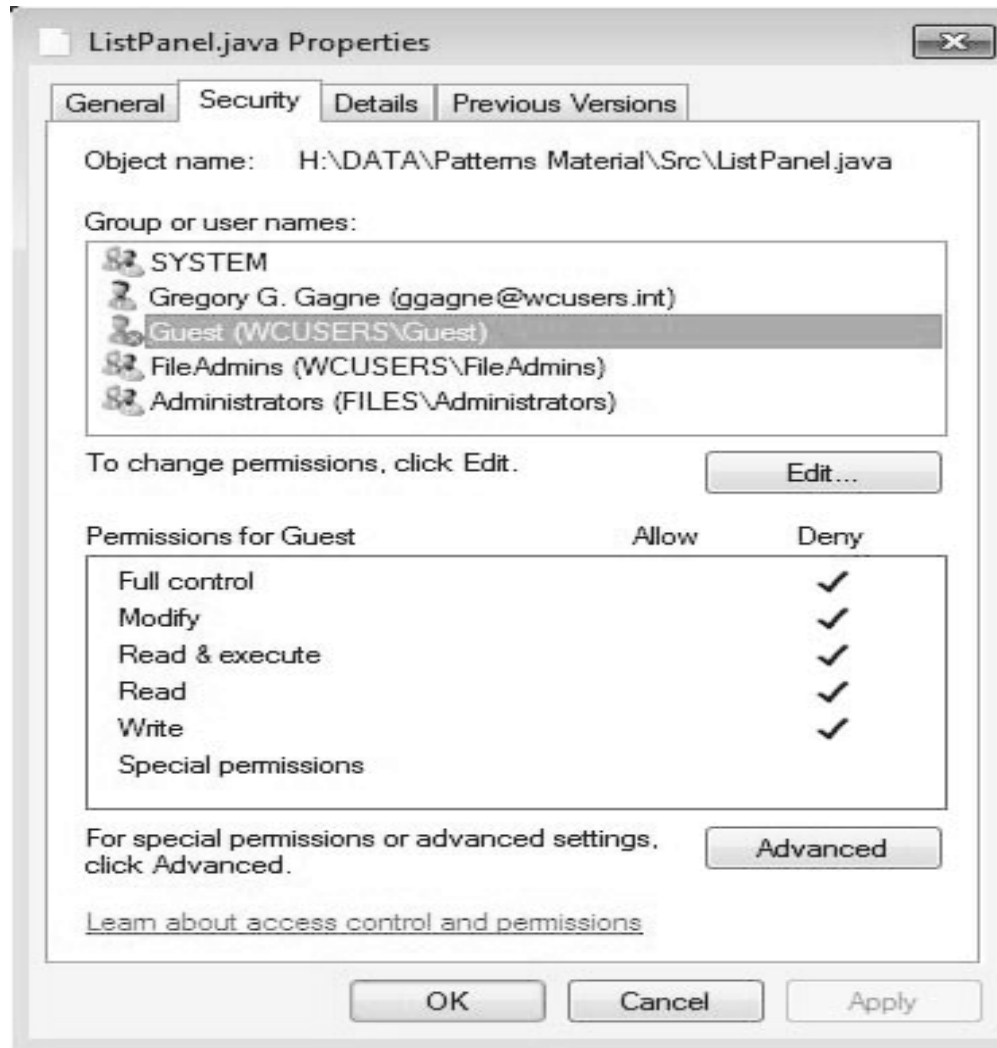
? For a particular file (say *game*) or subdirectory, define an appropriate access.

owner group public
 \ | /
chmod 761 game

Attach a group to a file
chgrp G game

OPERATING SYSTEMS

Windows Access-Control List Management





THANK YOU

Suresh Jamadagni

Department of Computer Science and Engineering

sureshjamadagni@pes.edu