UNIX

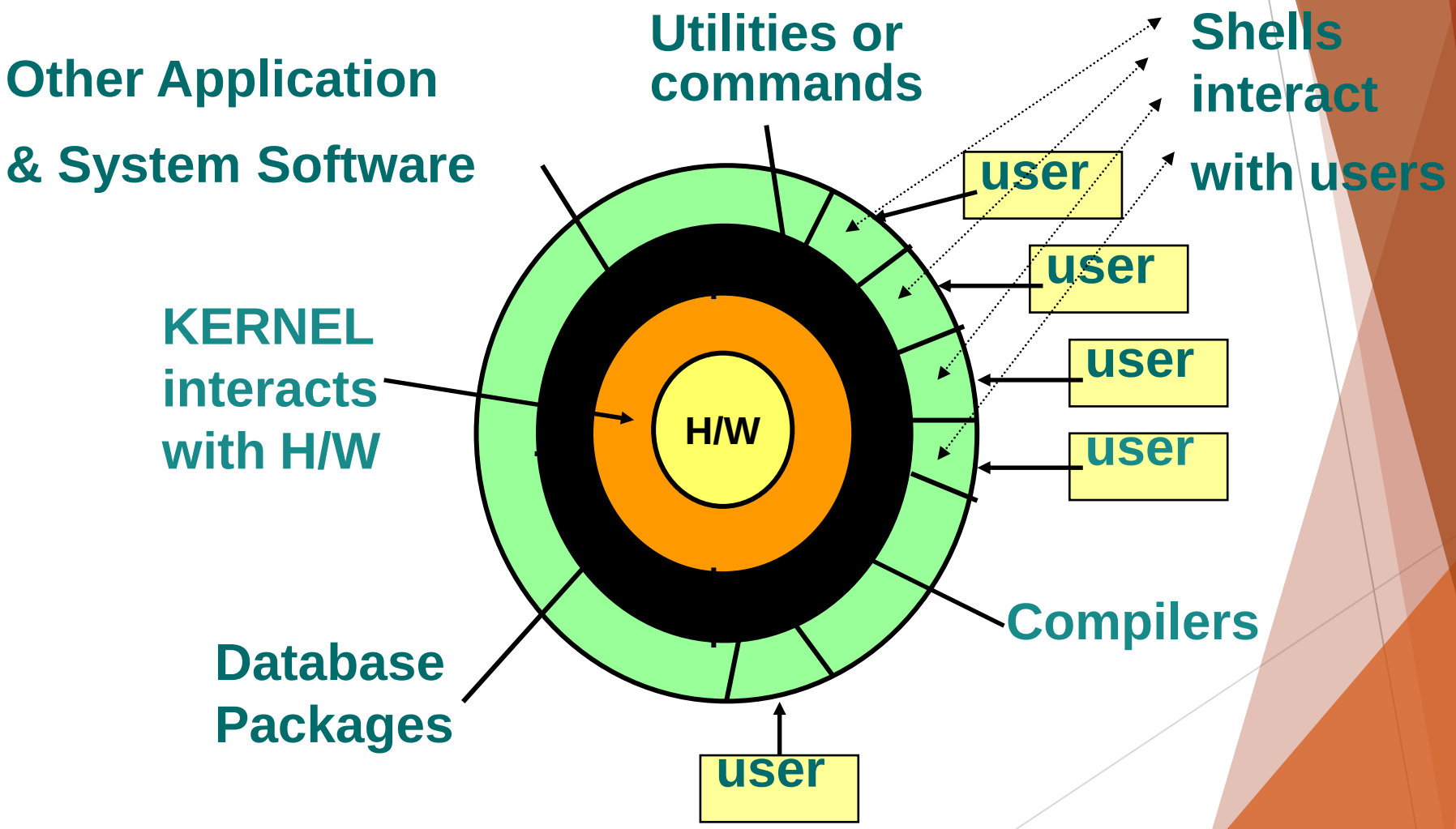
Sujata Batra

Introduction to UNIX

- It is a time-sharing, multi-user OS with elegant, featureless file system, a command interpreter (shell), and a set of utilities (tools/commands, over 200 programs);
- Developed by Ken Thompson and Ritchie originally in assembly, and later in C, thus making it portable to other machines.
- Supports C, Fortran, Basic, Pascal, COBOL, Lisp, Prolog, Java, Ada compilers.
- Facility of interconnecting commands through *pipes* and *filters* permit the user to create complex programs from simple programs
- Facility of background processing helps the user in effective utilization of time
- It is an OS for Programmers - shell provides the programming facility
- Security is in-built through the user name and password, combined with the access rights associated with files
- Types: Sco UNIX, Sun UNIX, Xenix, LINUX

UNIX Architecture

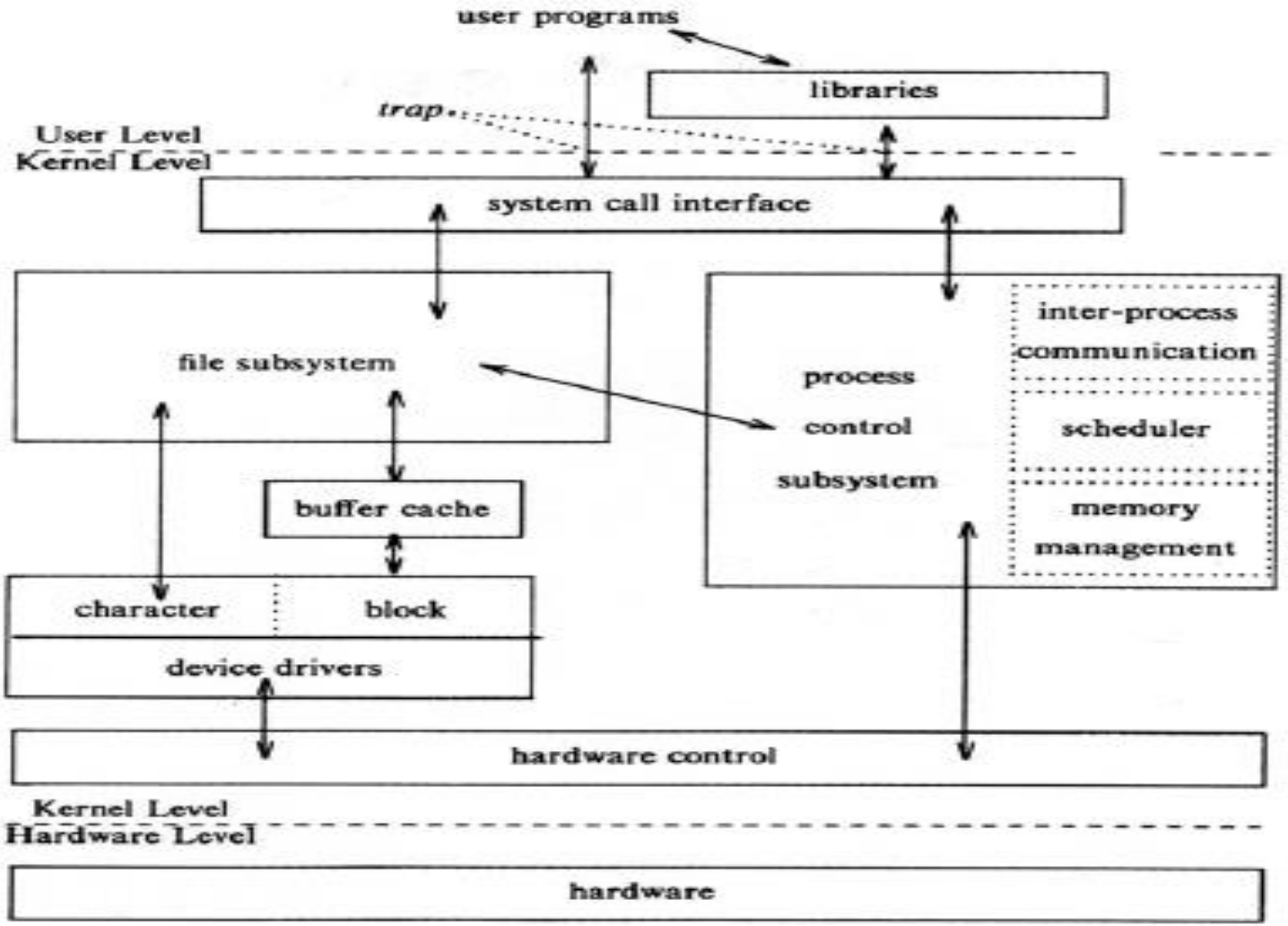
- Shell and kernel together make UNIX system work.



The UNIX Kernel

- Kernel is a collection of programs mostly written in C, that runs directly on the hardware - so parts of kernel must be customized to each system's hardware features.
- Kernel is directly loaded into memory when the system is booted. Low level jobs are its basic services:
  - System Initialization
  - Process/Memory/File/I-O Management
  - Programming Interface
  - Communication Facilities

Block Diagram of Unix Kernel



The Shell

- The shell acts as an interface between the user and the kernel.
- When a user logs in, the login program checks the username and password, and then starts another program called the shell.
- The shell is a command line interpreter (CLI). It interprets the commands the user types in and arranges for them to be carried out.
- The commands are themselves programs: when they terminate, the shell gives the user another prompt

UNIX has a variety of Shell

- Bourne Shell
  - Developed by Dr. Steve Bourne at AT&T Bell Laboratory. Its features are:
    - Use of wildcards.
    - Input and output redirection.
    - A set of shell variables for customizing the shell environment.
    - Background execution of commands.
    - Command set , loop constructs and conditional statements for writing shell scripts.
- C Shell
    - Developed by Bill Joy at University of California, Berkeley. Its features are:
      - Use of wildcard.

- Input and output redirection.
- A set of shell variables for customizing the shell environment.
- Integer arithmetic.
- History mechanism that remembers previously executed commands.
- Aliasing for abbreviating frequently used commands.
- A built-in set of operators based on the 'C' programming language for writing shell scripts.

- Korn Shell
  - Developed by David Korn at AT&T Lab. Its features are:
    - History mechanism that remembers previously executed commands.

- Aliasing for abbreviating frequently used commands.
- Pattern matching for filenames.
- Interactive editing of the command line with either emacs or vi editor.
- Better function definitions providing local variables and the ability to create recursive functions.

- Bash Shell
  - Released by Brian Fox and Chet Ramey as part of the free software foundation GNU project. It provides all the features found in Bourne , C and Korn Shells .

The UNIX Process

- A process in UNIX is a program in execution with definite life-time and well-defined hierarchy.
- The context of a process is a snapshot of its current run-time environment that consists of the current state of the processor registers and –
- User program image - Process execution on user data structure that represents an application and is manipulated by its own functions (user mode execution).
- System image - Process execution on system's data structures that represent the resources (memory, files, devices) and supported by the kernel routines. Depending upon resource requirement and availability, process's states are controlled by executing the kernel routines accessed through system calls (system mode execution).
- The kernel maintains a process table to manage all processes. The two data structures per process are the user structure and the process structure.

The UNIX Process...

- The ***kernel process*** is the first (*root*) process that comes into existence when the system is booted. Its *process_id* and *group_id* are both 0.
- In establishing a multi-programming environment, the kernel process then creates the ***init process*** with its *process_id* of 1 and *group_id*, 0, showing that process 1 is part of process 0. The init process creates and manages terminal processes for active terminals in the system.
- At the time of creation, each terminal process belongs to process group 0 and is made to execute a program called ***getty***. Once each terminal process (now called a ***getty process***) establishes communication with the associated terminal, it displays a login message and waits for a user to input a user name

The UNIX Process…

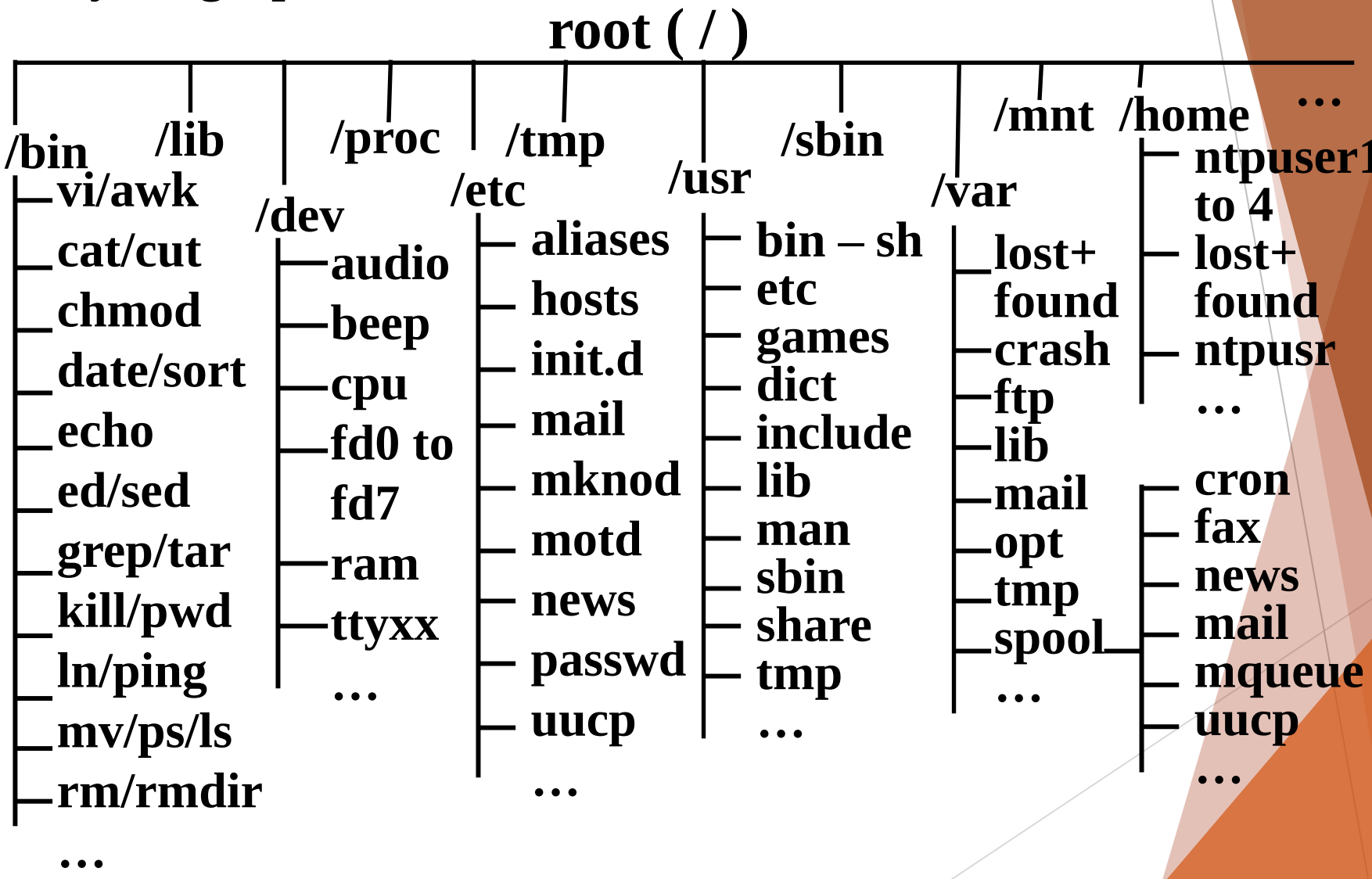
- When a user starts a dialogue, the ***getty process*** receives the user name and leaves the task of validating the user's password to another program called ***login***. The same terminal process is now called a ***login process***.
- The ***login process,*** after validating the user's password, calls a command line program, the ***login shell*** to run in the same process. The same terminal process is now called a ***shell process***.
- Each ***shell process*** now establishes a new process group and becomes ready to process the user commands. A *shell process* is the initiating process by which each terminal maintains the user session.
- While interpreting a command, the shell creates an execution thread and then assigns the requested command program to this new process.
- Both the shell and the new process proceed independently in separate execution threads. The parent shell process normally waits until child process completes its execution.

The UNIX File System

- It is a hierarchical collection of 3 types of files: ordinary, directory and special files (device, pipe, fifo, socket).
- A UNIX file is *featureless* because it is simply an array of bytes.
  - Dominant file type in UNIX is the text file.
  - System related files are also stored in text form.
  - Separate device can be added by creating a file for it.
- Root is the supremo and is represented by the '/'. Every subdirectory must have a parent.
- File names can contain both upper and lower case alphabets, digits, a dot, hyphen (-), underscore (_) anywhere; should not have a blank or tab; are case-sensitive.
- Path names are a sequence of directory names separated by '/'. They are used to access files.
- Absolute pathname - file location is determined with respect to the root.
- Relative pathname - file location is determined with respect to the current directory.

The UNIX File System …

Though the UFS looks hierarchical, it is actually a directed acyclic graph because files can be shared.



Sujata Batra

Important Directories in Linux File System

/home - It holds user's home directories. In other UNIX systems, this can be /usr directory.

/bin - It holds many of the basic Linux programs; bin stands for *binaries*, files that are executable.

/usr - It holds many user-oriented directories:

- bin - It holds user-oriented Linux programs.
- sbin - It holds system administration files.
- spool - It has several subdirectories:
  - . mail holds mail files
  - . spool holds files to be printed
  - . uucp holds files copied between Linux machines.
- docs - various documents including useful Linux info
- man - man pages accessed by typing the man <command>
- games - the fun stuff!

/sbin - It holds system files that are usually run automatically.

/etc - It and its subdirectories hold many of Linux *config files*.

/dev - It holds *device files*. All info sent to /dev/null is thrown into trash. Your terminal is one of the /dev/tty files.

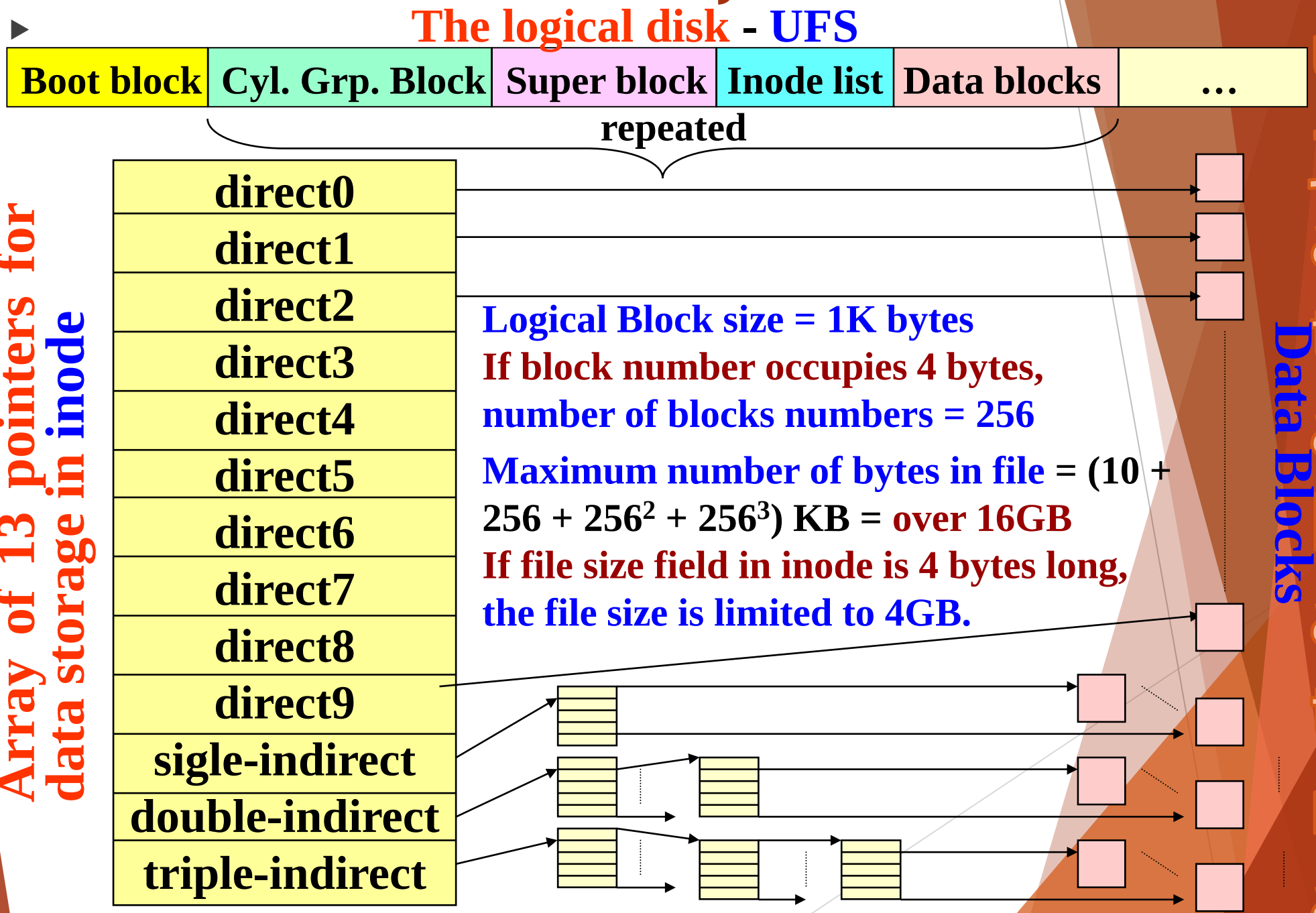
The UNIX File System …

- The UFS resides on a single logical disk. A logical disk is a disk partition comprising of a set of consecutive cylinders.
- UFS further subdivides a partition into one or more cylinder groups and attempts to allocate ***inodes*** and related data blocks from the same cylinder group, thus minimizing the disk head movements.
- At the beginning of the logical disk lies the boot block of UNIX operating system containing the ***bootstrap program***.
- It is followed by **repetitive cylinder groups** each one containing a ***super block***, ***cylinder group block***, ***inode list*** and the ***data area***.
- Each cylinder group contains a duplicate copy of the super block. The super block contains the size of file system, number of free blocks, index of next free block in free block list, size of inode list, number of free inodes, index of next free inode in free inode list.
- The cylinder group block contains a number of inodes and corresponding data blocks for that cylinder group. The block size is a power of 2 (>=4096).

Internal File Maintenance - UNIX File System …

- For each file created in the system, an ***inode*** is also created. ***Inode*** is a disk file record of 64 bytes that maintains the permanent attributes of a file.
- An ***inode*** is permanent and it exists until the corresponding file is removed from the system. Sample details of an *inode* -
  - Owner and group identifiers
  - File type and file size
  - Number of links for this file
  - Times of file creation, last file access and modification, and last inode modification
  - List of access rights - read/write/execute permissions
  - Reference count showing number of times file is opened
  - Physical address of file on the disk: array of 13 pointers for data storage
- Whenever a file is opened, its inode is brought into main memory. The active inode is kept there until the file is closed and is used to locate the beginning of an open file on disk and to verify that every I/O request comes from a valid user as per specified access permissions.

The *inode* - UNIX File System ...



Sujata Batra