Linux Internals & Networking

System programming using Kernel interfaces

Team Emertxe

Contents

Linux Internals & Networking
Contents

EMERTXE

System Call
Dup

Relationship Between

File Descriptors and Open Files

- Data Structures maintained by Kernel
  - The per-process file descriptor table
  - The system-wide table of open file descriptions
  - The file system i-node table
- For each process, the kernel maintains a table of open file descriptors. Each entry in this table records information about a single file descriptor, including
  - a set of flags controlling the operation of the file descriptor
  - a reference to the open file description
- The kernel maintains a system-wide table of all open file descriptions(Open file table / Open file handles).
  - the current file offset updated by read, write, lseek
  - status flags specified when opening the file i.e the flags argument to open()
  - the file access mode i.e read-only, write-only, or read-write, as specified in open()
  - a reference to the i-node object for this file

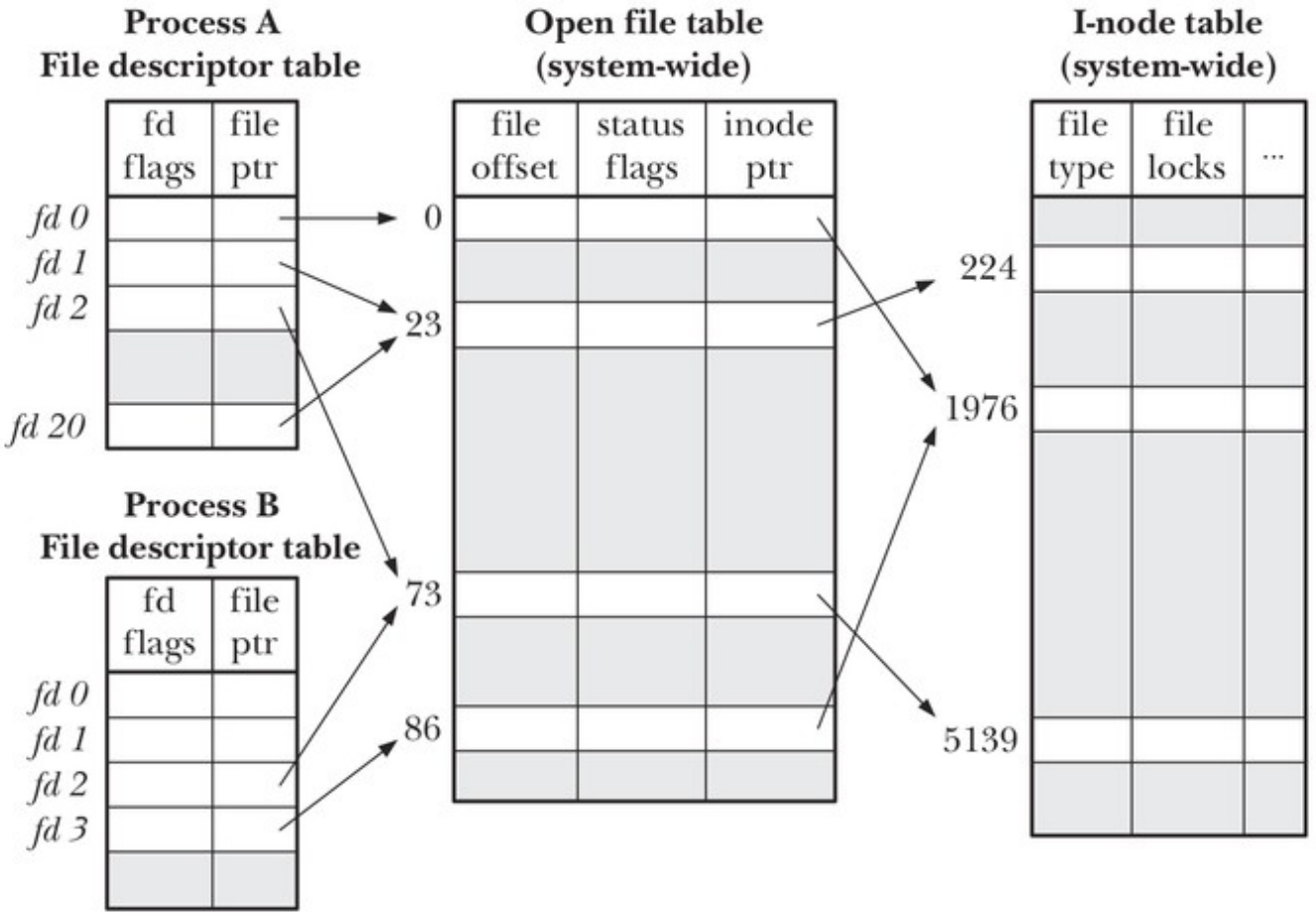
Relationship Between

File Descriptors and Open Files

- Each file system has a table of i-nodes for all files residing in the file system
  - file type (e.g., regular file, socket, or FIFO) and permissions
  - a pointer to a list of locks held on this file
  - various properties of the file, including its size and timestamps

Relationship Between

File Descriptors and Open Files



Relationship Between
File Descriptors and Open Files

- In process A, descriptors 1 and 20 both refer to the same open file description.
  - This situation may arise as a result of a call to dup(), dup2(), or fcntl()
- Descriptor 2 of process A and descriptor 2 of process B refer to a single open file description (73).
  - This scenario could occur after a call to fork() (i.e., process A is the parent of process B, or vice versa), or if one process passed an open descriptor to another process using a UNIX domain socket
- Descriptor 0 of process A and descriptor 3 of process B refer to different open file descriptions, but that these descriptions refer to the same i-node table entry (1976) in other words, to the same file.
  - This occurs because each process independently called open() for the same file.
  - A similar situation could occur if a single process opened the same file twice.

Thank You