Network Programming Interaction Session

Madan Kadariya
NCIT

Some useful terminologies in UNP

1. **File descriptor:** File descriptors are nonnegative integers that the kernel uses to identify the file being accessed by a particular process. Whenever the kernel opens an existing file or creates a new file it returns the file descriptor.
2. What are the **kernel data structures** are used for open files?
    - Process table entry
    - File table entry
    - V node structure

Some useful terminologies in UNP...

3. How will you retrieve the **file status**?

- The file status information are retrieved by using the functions **stat**, **fstat**, **lstat**.
- The **stat** function returns a structure of information about the named file.
- The **fstat** function obtains information about the file that is already open.
- The **lstat** function similar to stat but when the named file is a symbolic link.

Some useful terminologies in UNP...

4. What are the different types of **buffering** is supported by UNIX?

- **Fully buffered** : In this case the actual I/O takes place when the standard I/O buffer is filled.
- **Line buffered:** In this case the standard I/O library performs I/O when a new line character is encountered on input or output.
- **Unbuffered**: The standard I/O library does not buffer the characters.

Some useful terminologies in UNP...

5. Functions those who are return a **process ID** are:

- **getpid ()** : returns the current process ID
- **getppid ()** : returns the parent process ID of the calling process.
- **getuid()** : returns the real user ID of the calling process.
- **geteuid()**: returns the effective user ID of the calling process.
    - effective user ID gives the process additional permissions during execution of “set-user-ID" mode processes
- **getgid ()**: returns the real group ID of the calling process
- **getegid()**: returns the effective group ID of the calling process.

Some useful terminologies in UNP...

6. zombie process: The process that has terminated but whose parent has not yet waited for it called zombie process.

7. Wait() / Waitpid() :

- Either of wait or waitpid can be used to remove zombies.
- wait (and waitpid in it's blocking form) temporarily suspends the execution of a parent process while a child process is running.
- Once the child has finished, the waiting parent is restarted.

wait() and waitpid()

Declarations:

```
#include <sys/types.h>
#include <sys/wait.h>
   pid_t wait(int *statloc);                    /* returns process ID if OK, or -1 on error */
   pid_t waitpid(pid_t pid, int *statloc, int options);   /* returns process ID : if OK,
                0       : if non-blocking option && no zombies around
                -1      : on error
```

The statloc argument can be one of two values:

- **NULL pointer**: the argument is simply ignored
- **pointer to an integer**: when wait returns, the integer this describes will contain status information of the terminated process.

wait() and waitpid()

Wait()	Waitpid()
wait blocks the caller until a child process terminates	waitpid can be either blocking or non-blocking: • If options is 0, then it is blocking • If options is WNOHANG, then is it non-blocking
if more than one child is running then wait() returns the first time one of the parent's offspring exits	waitpid is more flexible: • If pid == -1, it waits for any child process. In this respect, waitpid is equivalent to wait • If pid > 0, it waits for the child whose process ID equals pid • If pid == 0, it waits for any child whose process group ID equals that of the calling process • If pid < -1, it waits for any child whose process group ID equals that absolute value of pid

Some useful terminologies in UNP...

8. Purpose of exec() functions

- When a process calls one of the exec functions that process is completely replaced by the new program.
- The new program starts execution from main function.
- The process does not change across an exec because a new process is not created.
- But this function replaces the current process with new program from disk.

Some useful terminologies in UNP...

9: Process group

- A process group is a collection of one or more processes.
- Each process group has a unique process ID.
- A function **getpgrp()** returns the process group id of the calling process.
- Each process group have a leader.
- The leader is identified by having its process group ID equal its process ID.
- A process joins an existing process group or creates a new process group by calling **setpgid()**.

Some useful terminologies in UNP...

10. Signal:

- Signals are software interrupts. Signals provide a way of handling asynchronous events: a user at a terminal typing the interrupt key to stop a program or the next program in the pipeline terminating prematurely.
- **What are the conditions to generate a signal?**
  - The terminal generated signals occur when users press some terminal keys.
  - The hardware exceptions generate signals
  - The kill(2) function allows a process to send any signal to another process or process group.
  - The kill(1) function allows a process to send Software conditions generate signals

Some useful terminologies in UNP...

Pending signal

- A signal is generated for a process when a process event that causes the signal occurs.
- The signal is delivered to a process when the action for a signal is taken.
- During the time between the generation of signal and its delivery is referred as pending.

Purpose of kill and raise functions

- The **kill** function sends a signal to a process or a group of processes. The **raise** function allows the process to send a signal to itself.

Some useful terminologies in UNP...

11. fork(): It causes creation of a new process. The new process (child process) is an exact copy of the calling process (parent process) except for the Following:

- The child process has a unique process ID. The child process has a different process ID than the process ID of the parent process.
- The child process has its own copy of the parent's descriptors. These descriptors reference the same underlying objects
- file pointers in file objects are shared between the child and the parent.
- Upon successful completion, **fork()** returns a value of 0 to the child process and returns the process ID of the child process to the parent process.
- Otherwise, a value of -1 is returned to the parent process, no child process is created, and the global variable **errno** is set to indicate the error.

Some useful terminologies in UNP...

Problems of fork() function

- Fork is expensive. Because memory and all descriptors are duplicated in the child.
- Inter process communication is required to pass information between the parent and the child after the fork.
- A descriptor in the child process can affect a subsequent read or write by the parent.
- This descriptor copying is also used by the shell to establish standard input and output for newly created processes as well as to set up pipes.

Some useful terminologies in UNP...

12. Sockets

- A socket is a construct to provide a communication between computers.
- It hides the underlying networking concepts and provides us with an interface to communicate between computers.

13. Value result arguments

- The way in which the length of the structure passed is depends on which direction the structure is being passed from process to kernel or vice versa.
- The size parameter is passed by value when the function is called and passed by result when the function returns.
- This type of argument is called value result argument.

Some useful terminologies in UNP...

14. Little byte endian order and Big byte endian order

- **Little byte endian order**:
  - In sixteen bit integer number, the low order byte at the starting address and higher order byte at the increasing memory address.
- **Big byte endian order**:
  - In sixteen bit integer number, the high order byte at the starting address and lower order byte at increasing memory address.

Some useful terminologies in UNP...

15. Purpose of address conversion functions

- The functions **inet_aton()** ,**inet_ntoa()** and **inet_addr()** convert IPv4 address between dotted decimal string and its 32-bit network byte ordered binary value.
- The functions **inet_pton()**, **inet_ntop** handle both IPv4 and IPv6 address. The letters **p** and **n** stands for presentation and numeric or network.

16: Purpose of connect and bind function.

- The connect function is used by a TCP client to establish a connection with a TCP server.
- The bind function assigns a local protocol address to a socket.

Some useful terminologies in UNP...

17. Different address family and type values available for creating a socket

- The different option values are
  - **AF_INET** or **PF_INET**: Address family for internet (IPv4), PF: Protocol family
  - **AF_INET6** or **PF_INET6** : Address family for internet (IPv6)
  - **AF_LOCAL** or **AF_UNIX**: Address family for Local or unix domain socket.
  - **AF_ROUTE**: Address family for Routing sockets. A routing socket is a special type of socket that is not specific to any particular network protocol, but allows “privileged" user processes to write information into the kernel. User processes use this type of socket to add and remove information from the routing table.
  - **AF_KEY**: Address family for Key management protocol, originally developed for usage with IPsec. This has no relation to keyctl and the in-kernel key storage facility

Some useful terminologies in UNP...

- The different type values are
  - **SOCK_STREAM** : A **SOCK_STREAM** type provides sequenced, reliable, two-way connection based byte streams. An out-of-band data transmission mechanism may be supported.
  - **SOCK_DGRAM** : A **SOCK_DGRAM** socket supports datagrams (connectionless, unreliable messages of a fixed (typically small) maximum length).
  - **SOCK_RAW**: **SOCK_RAW** sockets provide access to internal network protocols and interfaces. The type **SOCK_RAW**, which is available only to the super-user.
  - **SOCK_SEQPACKET**: Provides a sequenced, reliable, two-way connection based data transmission path for datagrams of fixed maximum length; a consumer is required to read an entire packet with each input system call.

Some useful terminologies in UNP...

18: Actions performed by listen function:

- The listen function is called by TCP server and it performs the following actions:
  - It converts the unconnected(closed) socket into a passive socket, indicating that the kernel should accept incoming connection requests directed to this socket.
  - It specifies the maximum number of connections that the kernel should queue for this socket.

Some useful terminologies in UNP...

19. Concurrent servers and Iterative servers.

- The servers that can handle multiple clients simultaneously are called concurrent servers.
- The servers that can handle multiple clients serially are called concurrent servers.

Some useful terminologies in UNP...

20. Different I/O models

- **Blocking I/O**: The control does not return to the application until the I/O is complete.
- **Nonblocking I/O**: It instruct the kernel as “when an I/O operation that the process requests cannot be completed without putting the process to sleep, do not put the process to sleep, but return an error instead.”
- **I/O multiplexing:** Check for multiple descriptor to be ready.
- **Signal driven I/O**: A SIGIO signal is used to tell the kernel when the descriptor is ready.
- **Asynchronous I/O**: These functions work by telling the kernel to start the operation and to notify us when the entire operation.

Some useful terminologies in UNP...

21. Synchronous and asynchronous I/O operations

- A synchronous I/O operations causes the requesting process to be blocked until I/ O operations complete.
- An asynchronous I/O operation does not cause the requesting process to be blocked.

22. UDP sockets

- UDP is a connectionless unreliable, datagram protocol.
- In UDP the client does not establish connection with a server, Instead the client just sends a datagram to the server.

23. TCP sockets

- TCP sockets provide a simple and effective way to provide connection oriented client server networking.

Some useful terminologies in UNP...

24. Generic socket options

- **SO_DEBUG**: enables recording of debugging information
- **SO_REUSEADDR:** enables local address reuse
- **SO_REUSEPORT:** enables duplicate address and port bindings
- **SO_KEEPALIVE:** enables keep connections alive
- **SO_DONTROUTE:** enables routing bypass for outgoing messages
- **SO_LINGER:** linger on close if data present
- **SO_BROADCAST**: enables permission to transmit broadcast messages
- **SO_OOBINLINE**: enables reception of out-of-band data in band
- **SO_SNDBUF**: set buffer size for output
- **SO_RCVBUF:** set buffer size for input
- **SO_SNDLOWAT:** set minimum count for output
- **SO_RCVLOWAT:** set minimum count for input
- **SO_SNDTIMEO:** set timeout value for output
- **SO_RCVTIMEO**: set timeout value for input

Some useful terminologies in UNP...

25: Functions available for set and get the socket options

- The functions for set and retrieve socket options are
- **getsockopt()** and **setsockopt():** manipulate the options associated with a socket. Options may exist at multiple protocol levels; they are always present at the uppermost “socket” level.
- **fcntl():** provides for control over descriptors
- **ioctl():** function manipulates the underlying device parameters of special files

Some useful terminologies in UNP...

26. Different functions done by TCP echo client server program

- The client reads a line of text from its standard input and writes it to the server.
- The server reads the line from the network input and echoes the line back to the client.
- The client reads the echoed line and prints it on its standard output.

Some useful terminologies in UNP...

27. Purpose of netstat program

- It shows the status of networking endpoints
- It shows the multicast groups that a host belongs to on each interface
- It shows the pre protocol statistics with the –s option
- It displays the routing table with the –r option and the interface information with the –i option

To Be continue... ∞