**ASSIGNMENT -3**

1. **Explain the Difference Between Mutex and semaphore with an example code ?**
2. **Explain the Difference Between Process and Threads in Linux with an example code ?**
3. **what are Different types of IPC mechanism involved in Linux Explain In detail?**
4. **Explain and illustrate IPC Shared Memory using an example code ?**
5. **Explain and Illustrate the Difference between Unnamed Pipe and FIFO with and example code ?**
6. **Explain and Illustrate IPC mechanism using sockets (Server and client Program ) with an example code ?**
7. **Explain and Illustrate IPC mechanism Message Queue with an example code?**
8. **Explain and illustrate with how to create a Kernel Module with and example code?**
9. **Explain and Illustrate How kernel threads are created with an example code ?**
10. **Write a Brief summary of the concepts that in involved in Linux system programming and your Understanding w.r.t to each Module.**

**BONUS POINT QUESTION**

**What skills will be required:**

**1.**[**Pthreads programming may be required**](http://www.google.com/url?q=http%3A%2F%2Fwww.thegeekstuff.com%2F2012%2F04%2Fcreate-threads-in-linux%2F&sa=D&sntz=1&usg=AOvVaw2UyFjX48mc0RYIAF4eD2jp)

**2.**[**Learn usage of pcap library for receiving of packets, this will make program simple**](http://www.google.com/url?q=http%3A%2F%2Fwww.thegeekstuff.com%2F2012%2F10%2Fpacket-sniffing-using-libpcap%2F&sa=D&sntz=1&usg=AOvVaw1ByO5IjsfA8RCP1wmB7Lhu)

**3.**[**basic socket programming for transmission of packets**](http://www.google.com/url?q=http%3A%2F%2Fwww.binarytides.com%2Fsocket-programming-c-linux-tutorial%2F&sa=D&sntz=1&usg=AOvVaw25uAtZ_x_2OI_6U0cR1n3M)

**4. Linked list creation and management**

**5. Structure creation and access for interpreting different types of packets**

**Work on localhost and run the server and client programs on different terminals. With more client program instances being run on independent terminal, the single server program instance should enumerate incoming client.**

1. **Program definition:**

**Program has two pieces in it:**

**1. Client-side protocol**

**2. Server-side protocol**

**Use Linux socket programming for communication over connected local network.**

**Server-side protocol,**

**1. Respond to any JOIN packet being received and save its details from the incoming packet**

**2. Send an INVITATION packet to the client with,**

**generated Unique ID with which server will be sending commands to this client**

**Assign a unique name to the client**

**3. Keep sending DISCOVERY packet to all clients every 3 seconds to keep track of if they are alive. If a DISCOVERY packet is unanswered 3 times, then remove client details from the connected list. In case, PRESENT packet for a client is received within the timeout, the 3 second timer along with count (3) is reset and discovery continues.**

**4. Server will be generating following commands to all connected clients in loop,**

**- Command set-1,**

**- commands: play, pause, stop**

**- Command set-2,**

**- direction-up, direction-down, direction-left, direction-right**

**Client-side protocol,**

**1. Send a JOIN packet which reaches server-side application.**

**2. Expect to receive a INVITATION packet with necessary details as follows to be given,**

**- Unique ID**

**- Assigned name**

**3. Once client is on server's connection list and after client responds to the first DISCOVERY packet from server, commands will start coming into the client (the DISCOVERY packets at the given interval will be coming in parallel and client should respond to that as well).**

**4. Commands to the client can be seen in the server-side description above.**

**5. Print details of each received command.**

**6. Server will be sending a DISCOVERY packet at an interval of 3 secs to which client should respond with PRESENT packet with its unique ID in payload. If Client fails to respond to 3 successive DISCOVERY packets, then Server drops the connection and stops sending any commands to client for execution.**