Vika 5

TÖL401G - Stýrikerfi

1. Identify the separate components of a process and illustrate how they are represented and scheduled in an operating system.

- Program counter (PC)
  – The PC is a register that contains the address of the next instruction to be executed.
- Stack
  – The stack is a data structure that contains temporary data such as function parameters, return addresses, and local variables.
- Data section
  – The data section contains global variables.
- Set of further associated resources
  – heap
  – Open files

2. Describe how processes are created and terminated in an operating system, including developing programs using POSIX system calls that perform these operations.

- Process creation
  – fork() system call creates a new process by duplicating the calling process. The new process is referred to as the child process. The calling process is referred to as the parent process.
  – exec() system call used after a fork to replace the process' memory space with new program.
- Process termination
  – wait() returns data from child to parent (return value provided by exit system call)
  – exit() process executes last statement and voluntarily requests from the OS to be deleted.

3. Describe and contrast interprocess communication using shared memory and message passing.

- Shared memmory:
  – Once the shared memory are has been established, ordinary memory access techniques can be used to exchange information between processes without further OS support.
  – Processes need to synchronize their access to the shared memory to avoid conflicts.
  – Preferable when large amounts of data need to be exchanged between processes.
- Message passing:
  – OS has an internal buffer that can be accessed by different processes via send and receive operations to exchange data.
  – Operating system provides a set of system calls to create and manage the message buffers and to send and receive messages.
  – Preferable when smaller amounts of data need to be exchanged between processes.

4. Describe programs that use POSIX pipes and POSIX shared memory to perform interprocess communication.

- Pipe:
  – Pipes are a form of IPC that allow data to be transmitted between processes in a linear,

unidirectional manner. A pipe consists of a read end and a write end. When one process writes data to the write end of the pipe, another process can read that data from the read end. Pipes can be used to implement filters, where the output of one process serves as input for another process.

- Shared memory:
  – Shared memory is a region of memory that can be accessed by multiple processes simultaneously. It's an efficient form of IPC because it doesn't require copying data between processes; instead, they read and write directly to the shared memory region. Synchronization primitives like semaphores or mutexes are often used to ensure the integrity of the data in shared memory.

5. Describe client–server communication using sockets and including how to create client/server programs using the Java socket API.

- Sockets:
  – A socket is one endpoint of a two-way communication link between two programs running on the network. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to. An endpoint is a combination of an IP address and a port number. Every TCP connection can be uniquely identified by its two endpoints. That way you can have multiple connections between your host and the server.
- Java sockets:
  – Java API supports interprocess communicatiion using sockets. The java.net package provides classes that represent sockets and server sockets. The Port numbers are represented by Integers. IP addresses are represented by InetAddress objects. And for translating domain names to IP addresses, the InetAddress class provides a static method called getByName().