**Operating Systems – Theory Assignment 1**

1. (a) **Alternate mechanisms to enforce memory protection:**

* **Memory Protection using Segmentation** – This is a method of **dividing the system memory** into different segments. A reference to a memory location includes a **value that identifies a segment** and an **offset within that segment**. For example, the x86 architecture has multiple segmentation features that help use protected memory in this architecture.
* **Memory Protection using Keys** - A memory protection key (MPK) mechanism **divides physical memory into blocks** of a particular size, each of which has an **associated numerical value** called a protection key. **Each process also has a protection key** value associated with it. On memory access, the hardware checks that the current process's protection key **matches the value associated** with the memory block being accessed; **if not, an exception occurs**.

**References:**

* <http://www.cs.uni.edu/~diesburg/courses/cs3430_sp14/sessions/s12/s12_memory_protection.pdf>
* <https://www.geeksforgeeks.org/memory-protection-in-operating-systems/>
* <https://en.wikipedia.org/wiki/Memory_protection#Segmentation>

(b) **Examples where programs require memory to be shared between them**:

* We can observe the applications of shared memory using a simple **producer-consumer problem**:

The producer and the consumer process both share a “buffer,” which they use as a “queue”. The job of the Producer Process is to **generate the data and fill it into the buffer**. The job of the consumer process is to consume the data and empty the buffer. The **problem** is that the **producer must not generate data if the buffer is full,** and the **consumer must not consume data if the buffer is empty**. The **solution to this problem is shared memory**.  The producer and consumer processes must have a common buffer that resides in the region of memory.

* **During a login process:**

The Service Manager checks shared memory, finds that the login application is **not currently stored** there**, retrieves the code record** for the application from the database, and **then loads it into shared memory**. At this point, the Service Manager will **begin to execute** the login application. Now, the prompt form is displayed; it will check shared memory for the form, find that it is **not currently stored** there, **retrieve the login prompt** form from the database and **load it into shared memory**. Once the form is displayed, the user enters their user ID and password, the application then attempts to validate this information against Service Manager’s operator file.

* Shared memory is used to **transfer images** between the **application and the X server** on Unix systems or inside the **IStream object** returned by CoMarshalInterThreadInterfaceInStream in the **COM libraries** under Windows.

**References -**

* <https://en.wikipedia.org/wiki/Shared_memory>
* <https://docs.microfocus.com/SM/9.61/Hybrid/Content/performance/shared_memory/creation_and_use_of_shared_memory.htm#:~:text=Shared%20memory%20is%20a%20common,of%20sharing%20data%20between%20processes>.
* <https://binaryterms.com/shared-memory-system-in-ipc.html#Example>

2) **Fields in the Process Control Block of a process in the latest Linux OS:**

1. **Process ID** - It is a **unique identification number** which is assigned by the OS at the time of process creation.

2. **Process Status** - It contains the **current state information** in which the process is residing (generally new, ready, waiting, running, terminated)

3. **Program Counter** - It contains the address of the next instruction to be executed.

4. **CPU registers** – A CPU register is a **quickly accessible small-sized location** available to the CPU. These registers are **stored in virtual memory** (RAM).

5. **Context Switching** – This takes care of **switching the CPU from one process/ task to another**. It stores the state of the process so that it can be restored and its **execution can be resumed** at a later point. This allows multiple processes to share a single CPU.

6. **CPU scheduling information** - Scheduling information is used to **set the priority of different processes**. This is very useful information which is set by the process control block. In a computer system there are many processes running simultaneously, and each process has its own priority. The priority of the primary feature of RAM is higher than other secondary features, for example. Scheduling information is very useful in managing any computer system.

7. **Memory-management data** – It manages the record of how much memory is available and how much memory has been allocated to that process. It consists of **page and segment tables** and also stores the data contained in **base and limit registers**.

8. **Accounting data** - This section of the process control block stores the details related to **CPU utilization and execution time of a process**. The time limits, account numbers, amount of CPU used, process numbers etc., are all a part of the PCB accounting information.

9. **I/O Status –** This section pertains to the **list of I/O devices** allocated to that process.

10**. List of Open Files –** This contains information on **all the files that are used by that given process**. It helps the OS to close all opened files at the termination state of the process.

**References:**

* <https://www.cs.fsu.edu/~zwang/files/cop4610/Fall2016/chapter3.pdf>
* <https://www.scaler.com/topics/operating-system/process-control-block-in-os/>
* <https://www.includehelp.com/operating-systems/process-control-block-in-operating-system.aspx>
* <https://www.tutorialspoint.com/what-is-process-control-block-pcb#:~:text=Process%20Control%20Block%20is%20a,in%20terms%20of%20the%20PCB>.