Ques - 5

Ans - 5

The relative advantages and disadvantages of implementing a threads package in user space and in the Kernel :-

(1) The most important advantage of the user-level approach is that a threads package can be implemented on the top of existing OS that does not support threads. It's not possible in the Kernel-level approach because in this approach the concept of thread must be incorporated in the design of the Kernel of an OS.

(2) In the user-level approach, due to the use of ~~the~~ two-level scheduling, users have the flexibility to use their own customized algorithm to schedule the threads of a process.

(3) Switching context from one thread to another is faster in the user-level approach than in the Kernel-level approach. This is because in the former approach context switching is performed by the ~~the~~ runtime system, while in the latter approach Kernel is needed for it.

) In the Kernel-level, the status information for hreads is maintained within the Kernel. Due to s, the scalability of the Kernel-level approach s poor as compared to the user-level roach.

(5) A serious drawback associated with the user-level approach is that with this approach the use of ~~the~~ round-robin scheduling policy to timeshare the CPU cycles among the threads on a quantum-by-quantum basis is not possible. This is due to the lack of clock interrupts within a single process. Therefore, once a thread is given the CPU to run, there is no way to interrupt it, and it continues to run unless it voluntarily gives up the CPU.

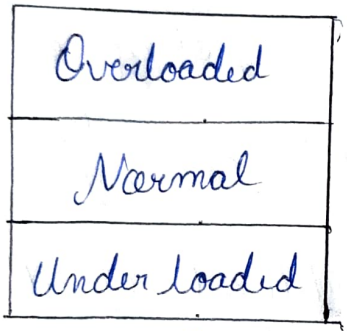
(6) Another drawback of the user-level approach is associated with the implementation of blocking system calls. In the kernel-level approach, implementation of blocking system calls is straightforward because when a thread makes such a call, it traps to the kernel, where it is suspended, and the kernel starts a new thread.

Ques-2

Ans-2

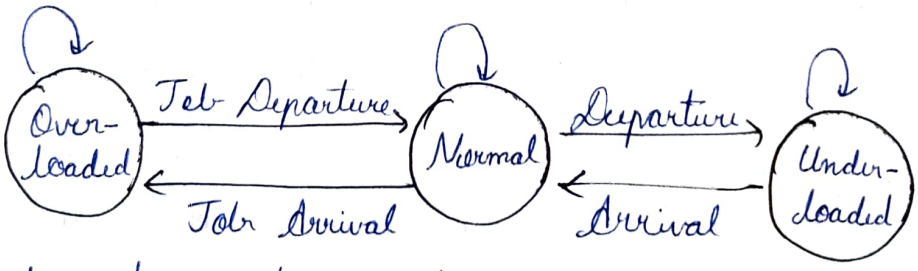
High-low policy is a double threshold policy that uses the threshold values called high mark and low mark, which divide the space of possible load states of nodes into the three following regions :-

* Overloaded - above the high mark and low mark value.
* Normal - above the low mark value and below the high mark value.
* Underloaded - below both values.



Double Threshold policy

A node's load state switches dynamically from one region to another, as shown below



State transition diagram of load of node in high-low policy.

High-low policy gaurantees a pre-defined level of performance to the model owners. It accounts for the overhead that the load-balancing algorithm may incur in transferring and recieving a remote process.

process will not be transferred to other node unless it is worth while, and remote process will not be transferred to another node unless it is worth while, and remote process will not be accepted unless there is enough excess to handle it.

The low-mark and high-mark can be determined by either static policy or dynamic policy. In dynamic policy, the threshold values of node is calculated as the product of average workload of all nodes and predefined constant C_i^l (low mark) and C_i^h (high mark).

In this policy nodes exchange state information, which may involve overhead. whereas, in static policy threshold is predefined and does not depend on the state of other nodes. The threshold value may be different for different processors depending on the processing capabilities of the processors.

Ques-6

Ans-6

A stateful file server maintains clients state information from one access request

to the next. That is, for two subsequent requests made by a client to a stateful server for accessing a file, some state information pertaining to the service performed for the client as a result of the first request execution is stored by the server process. This state information is subsequently used when executing the second request.

Problems associated with stateful file servers:

(1) If a server crashes and then restarts, the state information that it was holding may be lost and the client process might continue its task unaware of the crash, producing inconsistent results.

(2) If a client process crashes and then restarts, the server is left holding state information that is no longer valid but cannot easily be withdrawn.

Therefore, the stateful service ~~part~~ paradigm requires complex crash recovery procedures. Both servers and clients need to reliably detect crashes.

The servers need to detect client crashes so that it can discussed any state it is

holding for the client, and the client must detect server crashes so that it can perform necessary error-handling activities.

Despite having problems, in some cases the stateful service becomes necessary. For example,

(a) In internetworks, messages may not be received in the same order in which they are sent. A stateful service is preferable in such a case, since by maintained state it is possible to order the messages correctly.

(b) Similarly, if the file system uses the server-initiated cache validation approach, it cannot use the stateless service paradigm since the server has to maintain a record of which file are cached by which clients.

Ques-4

Ans-4

The server process can be designed to benefit from the concurrency made possible by threads in the following way:-

As a group of threads model

This model supports parallelism with blocking system calls. In this method the server process is comprised of a single dispatcher thread and multiple worker threads. Either the worker threads can be created dynamically, whenever a request comes in, or a pool of threads can be created at start up time to deal with as many simultaneous requests as there there are threads. The dispatcher thread keeps waiting in a loop for requests from the clients. When a client request arrives, it checks it for access permission. If permission is allowed, it either creates a new worker thread or chooses an idle worker thread from the pool and hands over the request to the worker thread. The control is then passed on to the worker thread and dispatcher threads state changes from running to ready. Now the worker thread checks to see if a disk access is needed for the request or if it can be satisfied from the block cache that is shared by all the threads.

If disk access is needed, it sends a disk access request to the disk server and blocks while waiting for a reply from the disk server. At this point, the scheduler will be invoked and a thread will be selected to run from the group of threads that are in the ready state. The selected thread may be the dispatcher thread ~~may~~ or another worker thread that is now ready to run.

This was how server process can be benefitted from concurrency made ~~tto~~ possible by threads.

Now, ~~Threads~~ the client process can be designed to benefit from the concurrency made possible by threads in the following way :-

A client process may use a divide-and-conquer algorithm to divide data into blocks that can be processed seperately. It can then send each block to a server to be processed and finally collect and combine the results. In this case, a seperate client thread may be used to handle each data block to interact with the different ~~server~~ server processes.

Ques-10

Ans-10

Jini technology is a servic oriented architecture that defines a programming model which both exploits and extends Java technology to enable the construction of secure, distributed systems consisting of well behaved network services and clients.

Jini is a simple system defining a small, simple set of conventions allowing services and clients to form a flexible distributed systems tolerant to change in the environment. The best way to illustrate Jini as a system is to separate its various components into :-

1) Infrastructure
2) A programming model
3) Clients and Services

The issues with Jini network it offers a small set of rules defining how different devices should communicate in a Java based environment.

Another major drawback with in Jini network is that each device must run a Java Virtual Machine - JVM. To run a JVM there is a need for substantial computing resources.

Another drawback with Jini lies in the scalability problem of a system based on generative communication using multicasting to either insert a tuple into Java space or remove would lead to serious networking problems, congested networks if implemented in wide area networks.

There are not yet so many Jini enabled products but companies use Jini connection technology for application specific. List of problems are :-

* Lack of Centralized Management :-
manage a large of number of devices in an Jini.

* Limited security scalability :-
Jini's transaction support is limited can be complex and difficult to configure for large network.

* Limited Transcation Scalability :-
Jini's transaction support is limited, which can effect some apptication.

* Poor Performance in Large Networks :-
Jini's performance can degrade in large network due to increased overhead of multicast communication.

Ques-3

Ans-3

Issues in freezing and restarting operations:-

(1) Blocking of process:-

Before a process is frozen, it must be blocked to execute. This can be immediate or delayed blocking. If process in not executing a system call or ~~extracting~~ executing but sleeping at interruptible priority then it can be immediately blocked while if it is sleeping at uninterruptible priority then it is blocked after completion.

(2) Fast and slow I/O Operations:-

The nat step is to work for fast I/O Operations to complete But it is not feasible to wait for slow I/O Operations for which proper mechanisms are necessary to do it correctly.

(3) Information about Open Files:-

A process status information also consits of information pertaining to files currently open by the process. In UNIX system, it is difficult for a process to obtain a files complete path name owing to its semantics and thus it is necessary to preserve a pointer to file

so that migrated process can access it. One approach to solve the problem is, when snapshot of process is taken, a link (with special name) is created for each file that is in use.

(4) Reinstalling process on its distination node:-
An empty process is created on distination node similar to that allocated in creation. In some implementations, this copy has a process identifier different from migrating process and needs to be changed to original identifier in subsequent step before process starts executing.

Ques-7

Ans-7

(a) the same type of locks for both read and write operations:

The three transactions can be controlled by using the same type of locks for both read and write operations. For example, all three transactions can use exclusive locks. This will ensure that T1 and T2 cannot read the data at the same time and that T3 will have to wait until T1 and T2 are done before it can write to A.

(b) Type-specific locks:

The three transactions can be controlled by using type-specific locks. For example, T1 and T2 can use shared locks, while T3 uses an exclusive lock. This will ensure that T1 and T2 can read the data at the same time, but that T3 will have to wait until they are done before it can write to A.

(c) Intention-to-write locks:

The three transactions can be controlled by using intention-to-write locks. For example, T1 and T2 can use shared locks, while T3 uses an intention-to-write lock. This will ensure that T1 and T2 can read the data at the same time, but that T3 will have to wait until they are done before it ~~was~~ can write to A.

(d) Optimistic concurrency control scheme:-

The three transactions can be controlled by using an optimistic concurrency control scheme. For example, T_1 and T_2 can read the data, while T3 will wait unatil they are done before they can write to A. If T3 attempts to write ~~A~~ to A while T_1 or T_2 are reading it, then it will recieve an error

and will have to retry the write operation.

(e) Timestamp-based concurrency control scheme:-

The thru transactions can be controlled by using a timestamp-based concurrency control scheme. For example, T_1 and T_2 ~~are~~ can read the data, while T_3 will wait ~~and~~ until they are done before it can write to A. T_3 will be assigned a timestamp that is later than the timestamp of T_1 and T_2. This will ensure that T_3's write operation is not lost.

Ques-9

Ans-9

The commonly used approaches for user authentication in computer systems are:-

(1) Proof By Knowledge:-

In this approach, authentication involves verifying something that can ~~be~~ only be known by an authorized principle. ~~An~~ Authentication of a user based on the password supplied by him or her is an example of proof by knowledge. Authentication methods based on the concept of proof of knowledge are again of two types:-

(a) Direct Demonstration Method:-

A user claims his or her identity by supply information (like typing a password) that the verifier checks against prestored information. ~~On other~~

(b) Challenge Response Method:-

A user ~~approves~~ his or her identity by responding correctly to the challenge questions asked by the verifier

(2) Proof By Possession:-

In this approach, a user proves his or her identity by producing some item that can only be possessed by an authorized principal. The system is designed to verify the produced item to confirm the claimed identity.

For Example, a plastic card with a magnetic strip on it that has a user identifier written on it. The user inserts the card in a card slot meant for this purpose in the system's terminal, which then extracts the user identifier number from the card and checks to see if the card produced belongs to authorized user.

(3) Proof by property :-

The system is designed to verify the identity of a user by measuring some physical characteristics of the user that are hard to forge. The measured property must be distinguishing that is, unique among all possible users. For example, a special device (known as biometric device) may be attached to each terminal of the system that verifies some physical characteristics of the user, such as the person's appearance, fingerprints, hand geometry, voice signature. In deciding the physical characteristic to be measured an important factor to be considered is that the scheme must be physcalogically acceptable to user community.

Ques-8

Ans-8

In the optimistic concurrency control approach, a transaction is validated only after it has completed its first phase. If the validation fails because some conflict is detected the transaction is aborted and restarted all over again. Execution of operation of the transaction that follow the operation that caused the conflict is actually a waste. In this case, the overhead involved in executing the operations of the transaction that follow the operation that caused the conflict can be totally avoided if it is possible to detect the conflict right at the time when the operation causing it is executed. This ~~is~~ because the transaction can be ~~avoided~~ aborted immediately at the point where the conflict occurs and it than be restarted. This has been made possible in the timestamp approach, in which each operation in a transaction is validated when it is ~~caused~~ called out. If the validation fails, the transaction is aborted immediately and it can then be restarted.