SHRIMATI INDIRA GANDHI COLLEGE

Affiliated to Bharathidasan University| Nationally Accredited at 'A' Grade(3rd Cycle) by NAAC

An ISO 9001:2015 Certified Institution

Thiruchirrappalli

DISTRIBUTED OPERATING SYSTEMS

QUESTION BANK

DEPARTMENT OF COMPUTER SCIENCE, INFORMATION TECHNOLOGY AND COMPUTER APPLICATIONS

Prepared by

P ANANTHI

ASSISTANT PROFESSOR IN COMPUTER SCIENCE

SHRIMATI INDIRA GANDHI COLLEGE,

TIRUCHIRAPPALLI - 2

UNIT-I

Define distributed system.

A distributed system is a collection of independent computers that appears to its users as a single coherent system. A distributed system is one in which components located at networked communicate and coordinate their actions only by passing message.

Tightly coupled systems:

In these systems, there is a single system wide primary memory (address space) that is shared by all the processors . If any processor writes, for example, the value 100 to the memory location x, any other processor subsequently reading from location x will get the value 100. Therefore, in these systems, any communication between the processors usually takes place through the shared memory.

Loosely coupled systems:

In these systems, the processors do not share memory, and each processor has its own local memory .If a processor writes the value 100 to the memory location x, this write operation will only change the contents of its local memory and will not affect the contents of the memory. In these systems, all physical communication between the processors is done by passing messages across the network that interconnects the processors.

List the characteristics of distributed system?

- Programs are executed concurrently,
- support for resource sharing.
- Openness
- Concurrency
- Scalability
- Fault Tolerance (Reliability)

- Transparency
- Components can fail independently (isolation, crash)

Mention the examples of distributed system.

- The internet,
- Intranet.
- Department computing cluster
- Corporate systems
- Cloud systems (e.g. Google, Microsoft, etc.)
- Mobile and ubiquitous computing

Mention the challenges in distributed system.

1. Heterogeneity
2. Openness
3. Security
4. Scalability
5. Failure handling
6. Concurrency
7. Transparency

What are the Advantages of Distributed Systems?

1. Performance

2.Distribution

3.Reliability (fault tolerance)

4.Incremental growth

5.Sharing of data/resources

6.Communication

What are different types of transparencies required in distributed systems?

- Access Transparency
- Execution Transparency
- Replication Transparency
- Performance Transparency
- Configuration Transparency

Write a short note on quality of service in distributed systems.

Quality of Service (a.K.a. QoS) refers to performance and other service expectations of a client or an application.

- Performance
- Reliability and availability
- security Examples where this is important.
- Voice over IP (VOIP) and telephony
- Video (e.g. Netflix and friends)

Operating Systems are of different types and some of them are:

Single User OS: An operating system designed for use by a single individual, for example MS-DOS (Microsoft Disk Operating System) is a single user operating system.

Multi User OS: An operating system that can be used by more than one person. Multi User operating system can be accessed simultaneously by several people through communications facilities or via network terminals. Windows operating system is one of the examples of multi user operating system.

Distributed OS: A form of information processing in which work is performed by separate computers linked through a communications network. Distributed operating system is usually categorized as either plain distributed processing or true distributed processing. The distributed operating system has been discussed in detail, later in this unit.

The various design issues in the development of distributed systems are stated as follows:

Transparency

Flexibility

Reliability

Performance

Scalability

A distributed system is said to be transparent if the users of the system feel that the collection of machines is a timesharing system and belongs entirely to him.

Transparency can be achieved at two different levels. In the first level, the distribution of the system is hidden from the users.

In the other level, the system is made to look transparent to the programs. Flexibility in distributed systems is important because this system is new for engineers, and thus there may be false starts and it might be required to backtrack the system.

Distributed systems are more reliable than single-processor systems because if one system in a distributed system stops functioning, other systems can take over.

Building a system, which is flexible and reliable, is of no use if the system is slower than a single-processor system. To measure the performance of the system, various performance metrics are used, such as number of jobs per hour, system utilization and amount of network capacity consumed.

Types of Networks

A computer network can be as small as several personal computers on a small network or as large as the Internet.

Depending on the geographical area they span, computer networks can be classified into two main categories, namely, local area networks and wide area networks.

Local Area Networks

A Local Area Network (LAN) is the network restricted to a small area such as an office or a factory or a building. It is a privately owned network that is confined to an area of few kilometers.

Wide Area Network (WAN)

A Wide Area Network (WAN) spreads over a large geographical area like a country or a continent. It is much bigger than a LAN and interconnects various LANs. This interconnection helps in a faster and more efficient exchange of information at a higher speed and low cost. These networks use telephone lines, satellite transmission and other long-range communication technologies to connect the various networks. For example, a company with offices in New Delhi, Chennai and Mumbai may connect their individual LANs together through a WAN. The largest WAN in existence is the Internet.

Network Topology A network topology refers to the way a network is laid out either physically or logically.

The various network topologies include bus, ring, star, tree, mesh, and graph.

Bus/Linear Topology

The bus topology uses a common single cable to connect all the workstations. Each computer performs its task of sending messages without the help of the central server. Whenever a message is to be transmitted on the network, it is passed back and forth along the cable from one end of the network to the other. However, only one workstation can transmit a message at a particular time in the bus topology.

Ring/Circular Topology

In the ring topology, the computers are connected in the form of a ring without any terminated ends. Every workstation in the ring topology has exactly two neighbours. The data is accepted from one workstation and is transmitted to the destination through a ring in the same direction (clockwise or counter clockwise) until it reaches its destination.

Star Topology

n the star topology, the devices are not directly linked to each other but are connected through a centralized network component known as the hub or the concentrator.

Tree Topology

The tree topology combines the characteristics of the bus and star topologies. It consists of groups of star-configured workstations connected to a bus backbone cable.

Mesh Topology

In the mesh topology, each workstation is linked to every workstation in the network. That is, every node has a dedicated point-to-point link to every other NOTES Self-Instructional Material 23 Introduction to Computer Networks node.

The messages sent on a mesh network can take any of the several possible paths from the source to the destination. A fully connected mesh network with n devices has n(n-1)/2 physical links'

What is mobile agent?

A mobile agent is a running program (including both code and data) that travels from one computer to another in a network carrying out a task on someone's behalf, such as collecting information, and eventually returning with the results.

What is layering?

The concept of layering is a familiar one and is closely related to abstraction.In a layered approach, a complex system is partitioned into a number of layers, with a given layer making use of the services offered by the layer below.

What are Software and hardware service layers in distributed systems? o Applications, services

- Middleware
- Operating system
- Computer and network hardware

What is RPC?

A remote procedure call (RPC) is an inter-process communication that allows a computer program to cause a procedure to execute in another address space (commonly on another computer on a shared network) without the programmer explicitly coding the details for this remote interaction.

What is the difference between RMI and RPC?

Remote Procedure Call or the RPC and the Remote Method Invocation or RMI are both message passing techniques in the Inter Process Communication (IPC). But there are two basic differences between the two methods: 1. RPC supports procedural programming. i.e. only remote procedures can be invoKed. Whereas RMI is object-based. As the name suggests, it is invoKed on remote objects. 2. In RPC, the parameters that are passed are ordinary data structures. Whereas in RMI, objects can be passed as parameters.

Define Stub.

A stub in distributed computing is a piece of code used for converting parameters passed during a Remote Procedure Call.

Client Stub:

Used when read is a remote procedure. Client stub is put into a library and is called using a calling sequence. It calls for the local operating system. It does not ask for the local operating system to give data, it asks the server and then blocks itself till the reply comes.

Server Stub:

When a message arrives, it directly goes to the server stub. Server stub has the same functions as the client stub. The stub here unpacks the parameters from the message and then calls the server procedure in the usual way.

What is the use of RMI registry?

The RMI registry is used to store a list of available services. A client uses the registry to make it's proxy object, and the Registry is responsible for giving appropriate information to the client so that it can hook up with the server that implements the service.

What is a Message queue?

Message queues offer a point-to-point service whereby producer processes can send messages to a specified queue and consumer processes can receive messages from the queue or be notified of the arrival of new messages in the queue. Queues therefore offer an indirection between the producer and consumer processes.

Define thrashing.

Thrashing is said to occur when the system spends a large amount of time transferring shared data blocks from one node to another.

Thrashing may occur in the following situations:

1. When interleaved data accesses made by processes on two or more nodes causes a data block to move back and forth from one node to another in quick succession (a ping – pong effect)

2. When blocks with read only permissions are repeatedly invalidated soon after they are replicated.

UNIT-II

What are the types of communication paradigm in DS?

- Interprocess communication;
- Remote invocation;
- Indirect communication.

Differentiate persistent and non-persistent connections?

HTTP can use both non-persistent connections and persistent connections. A non-persistent connection is the one that is closed after the server sends the requested object to the client. In other words, the connection is used exactly for one request and one response.

Non-persistent connections are the default mode for HTTP/1.0 With persistent connections, the server leaves the TCP connection open after sending responses and hence the subsequent requests and responses between the same client and server can be sent. The server closes the connection only when it is not used for a certain configurable amount of time.

What is meant by inter process Communication?

Inter process communication is concerned with the communication between processes in a distributed system, both in its own right and as support for communication between distributed objects. The Java API for inter process communication in the internet provides both datagram and stream communication.

What is Encoding?

It is the process of converting the original data packet into a stream that is compatible with the communication channel.

What is Decoding?

The process of retrieving the received message to the original message at the receiver node.

What is meant by group communication?

Group communication is a multicast operation is more appropriate- this is an operation that sends a single message from one process to each of the members of a group of process, usually in such a way that the membership of the group is transparent to the sender.

Message Passing

The message that is sent from a sender to receiver will either use synchronous or asynchronous communication method. However, when a message is sent from a sender it needs to be temporarily stored in a memory area until the receiver node receives the message. The message can be stored in a memory area that is available at the sender node or at the memory area which is managed by the operating system. This memory area which is used to store the message till the receiver node receives it is known as buffer and the process is known as buffering.

What do you mean by Inter-process communication?

Inter-process communication refers to the relatively low-level support for communication between processes in distributed systems, including message-passing primitives, direct access to the API offered by Internet protocols (socket programming) and support for multicast communication.

Write about Remote invocation?

Remote invocation represents the most common communication paradigm in distributed systems, covering a range of techniques based on a two - way exchange between communicating entities in a distributed system and resulting in the calling of a remote operation, procedure or method

Define atomicity

Atomicity refers to a condition when a message is sent to a group and it is received correctly either by all the members of the group or by none of the member of the group.

Explain the different types of buffering

Buffering are used based on the requirement of a process within a distributed operating system and some of them are given below:

1. Null Buffering: This type of buffering doesn't use any buffer rather the send process remains in suspended mode till the receiver node in a position to receive the message. Once the process of send message starts the receiver starts the receiving the message and accordingly an acknowledgement is sent once the message is delivered. The sender node on receipt of acknowledgement sends a message to the received in order to unblock the receiver node for further processing.

2. Single Message Buffering: This type of buffering uses a single buffer either at the receiver node address space in order to ensure that the message is readily available to the receiver as and when the receiver node is ready to accept the same. The single message buffer performs better in some situations as the message is available in the buffer which helps the while system in reducing the blocking duration at different nodes. The single message buffer method reduces the delays in communication in comparison with the Null buffering method of communication.

3. Multiple Message Buffering: The multiple message buffering communication mechanism is generally used in asynchronous type of communication in inter process communication within distributed operating system. The multiple message buffer as shown in the figure below works as a mail box which is either stored at the receiver's address space or operating system address space. A sender executes the send process in order to send a

message and the same is received by the receiver from the mail box as and when the receiver processes the receive message process.

Multidatagram Messages

The inter-process communication between different nodes within a distributed operating system is an essential part of a network based operating system. The messages transferred from a sender to a receiver in a network is in the form of packets where the data packets correspond of different information attributes like process identifier, address, sequence number, structural information and actual data.

A datagram is a self-sufficient and independent packet of data associated with a packet switched network. It carries adequate information to be routed from the source to the destination and the network. Datagrams provide a connectionless communication service across a packet-switched network. Every network allows a maximum allowable size of a datagram that can be transmitted from one node to the other and the same is known as MTU maximum transfer unit.

In case the size of the message is smaller than the maximum transfer unit then the datagram is known as “single datagram” message. However, in case the size of the datagram is more than the maximum transfer unit then the datagram is divided into smaller datagrams in order to communicate these multiple datagrams from sender to receiver.

The multiple datagrams communicated from sender to receiver Buffering and Multidatagram . These multi-datagrams include extra attributes within

a packet which carry the information about the sequence of the datagrams and mechanism used for fragmenting. This extra information is used at the receiver end to combine all the multiple datagrams into a single block of information.

Explain the requirement of encoding and decoding in communication system.

Encoding: A message transmitted from the source node to destination node is in the form of single datagram or multi-datagram. The message delivered at the destination or receiver node should be complete and correct. Therefore, the structure of the datagram should also be known at the receiver node in order to understand the complete properties of the datagram received. The receiver node should have the complete information related to the datagram available at sender node in order to maintain the consistency and integrity of data. To ensure this the datagram to be sent to the receiver node should be converted into a form which can be transmitted through the communication channel and accordingly on receipt of the packet the same should be converted back to the original form at the receiver node. The process of converted the original data packet into a stream that is compatible with the communication channel is known as encoding of the message and the process of reverted the received message to the original message at the receiver node is known as decoding. The received node encodes the original message and sends the same through the communication channel or buffer to the receiver node where the encoded message is decoded to get the original message back at the receiver node. Different methods are used for encoding and decoding and the two basic representation of encoding and decoding process are Tagged representation and untagged representation.

Decoding: In the tagged representation all the details about the object along with the data value is encoded and then send through any communication channel or buffer to the receiver. At the receiver node the

encoding done using tagged method reverted back to its original form. The data packet received by the receiver node is simple to decode and implement as the information about all the properties of the data packet along with the data is available. However, in untagged representation program objects do contain only data which does not make the data packet delivered at the receiver node as self-explanatory. The receiver node must have the information about the encoding method or mechanism used at the sender node in advance to decode the data packet to its original form.

What are the different ways of process addressing?

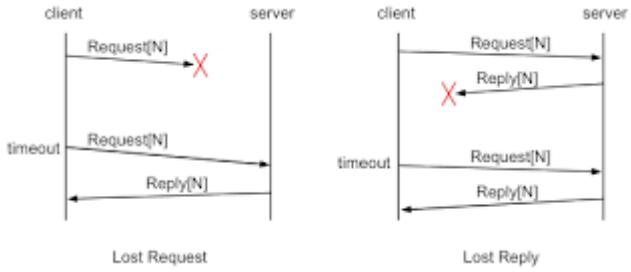
Explicit Addressing: when the message is explicitly destined for a process then the message is sent using the explicit mode of addressing. In this addressing mode, the process-identification (x) along with message (y) is sent to the receiver node (z). The receiver node (z) will only receive the message from process-identification (x). If any other message from process-identification (k) is available it will not be received by the receiver node (z).

Implicit Addressing: when a message is destined for any receiver node that requires the services of the message then implicit mode of addressing is used. In this addressing mode, the service-identification (x1) along with message (y) is intended to be sent to any receiver node where the serviceidentification(x1) is offered. The implicit mode of addressing allows a sender node to send the message to more than one node within a network provided the sender has to name a service rather than a process. This type of addressing mode is feasible for client-to-server communication where a client requests for a service and sends the message to all the available servers. The server in turn will receive the message and treat it as a process. Similarly a server can also send a message to all clients to access a service and inturn only clients which are allowed to use the service can receive the message from the server and acknowledge the same.

FAILURE HANDLING

The messages sent from the sender node are not received at the receiver node due to different failures which may occur in a distributed operating system. The scalable and robust design is always pro-active in order to give seamless services to usersinteracting with a distributed operating system. The failures that can occur while communicating a message within any two or more nodes of a network are discussed below.

. Request Message Lost: This problem can occur if the sender node sends a message and the receiver is down due to breakdown of the communication link between the sender node and the receiver node. In case the receiver node is down or gets crashed the communication between the sender node and the receiver node will not be possible and the same is shown in figure given below: In case the receiver node receives the message but crashes prior to sending the acknowledgement will result failure of communication as the sender node will not receive the acknowledgement. The scenario of receiver crash after receiving the message is shown in figure given below:



Node/Computer Crashes: The problem occurs when either a sender node or the receiver node crashes in the process of communication. The kernel of the operating system initiates the process of freeing the resources after waiting for timeout. In case the sender node crashes after sending the message and the receiver node does not receive the message, the kernel of the operating system waits for the timeout limit. After the expiry of the timeout the kernel frees the communication channel and clears the message and the processes

associated with the message. Similarly, in case the receiver node crashes after sending the acknowledgement to the sender node which in turn has to reply to the receiver node in order to complete the process and free the resources acquired by the process at the receiver end. The kernel of the operating system will initiate the process of freeing the resources acquired by the process after the timeout period is over.

The two main cases where a node may crash are given below:

(a) Receiver node crashes after receiving message

b) Sender node crashes after sending the message

GROUP COMMUNICATION:

A group consists of a collection of processes, which perform the task together in the system. If a message is sent to the group, all the processes receive it.

Different members of a group can communicate in two ways, one-to-many communication and one-to-one communication. In the one-to-many communication, one sender and many receivers are involved. One-to-many communication involving one sender and many receivers

Various Issues of Group Communication

Design Issues related to Group communication Different types of group communications can be designed to establish communication among multiple users of the system. There are a lot of design possibilities used in the designing of the group communication system. The regular message passing and primitives based communications are examples of such design possibilities.

Designing of the group communication system is entirely dependent on the internal organization of a group.

The types of group communication systems are stated as follows:

Closed group: This group refers to the group in which outsiders are not allowed to send messages to the group as a whole.

Open group: This group refers to the group in which an outsider can send message to any group involved in the network.

Peer group: This group refers to the group in which every member of the group is connected to the other members of that group.

Hierarchical group: This group refers to the group in which one member of the group acts as a coordinator of the other members of that group.

Difference between synchronous and asynchronous communication?

In synchronous form of communication, the sending and receiving processes synchronize at every message. In this case, both send and receive are blocking operations. Whenever a send is issued the sending process is blocked until the corresponding receive is issued. Whenever receive is issued, the process blocks until a message arrives.

In asynchronous form of communication, the use of the send operation is non-blocking in that the sending process is allowed to proceed as soon as the message has been copied to a local buffer and the transmission of the message proceeds in parallel with the sending process. The receive operation can have blocking and non-blocking variants.

Unit-III

Define distributed shared memory.

Distributed shared memory (DSM) is an abstraction used for sharing data between computers that do not share physical memory. Processes access DSM by reads and updates to what appears to be ordinary memory within their address space.

What is dirty read?

The dirty read problem is caused by the interaction between a read operation in one transaction and an earlier write operation in another transaction on the same object.

What is clock skew and clock drift?

The instantaneous difference between the readings of any two clocks is called their skew. Clock drift means that they count time at different rates, and so diverge.

What is clocks drift rate?

A clock's driftrate is the change in the offset (difference in reading) between the clock and a nominal perfect reference clock per unit of time measured by the reference clock.

Define Polling.

A method of continuously checking the buffer status for any messages by the receiver is followed in non-blocking synchronization method and is known as polling.

State ACID properties.

ACID properties are a) Atomicity, b) Consistency, c) Isolation and d) Durability

What are the three approaches for concurrency control?

The three approaches for concurrency control are a) Locking, b) Optimistic concurrency control and c) Timestamp ordering

Explain the Berkley algorithm.

The Berkley algorithm eliminates readings from faulty clocks. Such clocks could have a significant adverse effect if an ordinary average was taken so instead the master takes a fault tolerant average. That is, a subset is chosen of clocks that do not differ from one another by more than a specified amount, and the average is taken of readings from only these clocks.

What is mutual exclusion?

A **mutual exclusion** (mutex) is a program object that prevents simultaneous access to a shared resource. This concept is used in concurrent programming

with a critical section, a piece of code in which processes or threads access a shared resource.

Deadlocks

What do you mean by Deadlocks ?

A process request for some resources. If the resources are not available at that time , the process enters a waiting state . The resources was held by other processes .The waiting process may never able to get the resource. This situation is called deadlock.

What are the necessary conditions for deadlocks?

• Mutual exclusion: only one process at a time can use a resource. If another process requests the same resource, the requesting process must wait until the resource is released.

• Hold and wait: Processes currently holding resources granted earlier , can request for new resources , that are currently held by other. 3 • No preemption: a resource can be released by the process holding it only after that process has competed its task.

• Circular wait: The circular chain of two or more processes must exist such that each of them is waiting for a resource held by next member.

Explain the concept of Deadlock recovery

• Process termination

• Resource pre-emption

• Check point / roll back mechanism

Process termination

• Abort all deadlocked process

• Successively abort each deadlocked process until the deadlock no longer exists.

Resource pre-emption

Roll back A process that has a resource pre-empted from it must be roll back to the point to its acquiring of that resource.

Total roll back – Abort the process and restart it.

What is safe state?

state is safe if the system can allocate resources to each process (up to its maximum) in some order and still avoid a deadlock. More formally, a system is in a safe state only if there exists a safe sequence.

What is a phantom deadlock?

A deadlock that is ‘detected’ but is not really a deadlock is called a phantom deadlock.

What is wait-for-graph?

A wait-for graph can be used to represent the waiting relationships between current transactions. In a wait-for graph the nodes represent transactions and the edges represent waitfor relationships between transactions.

What are the phases of transactions?

a) Working phase,

b) b) Validation phase

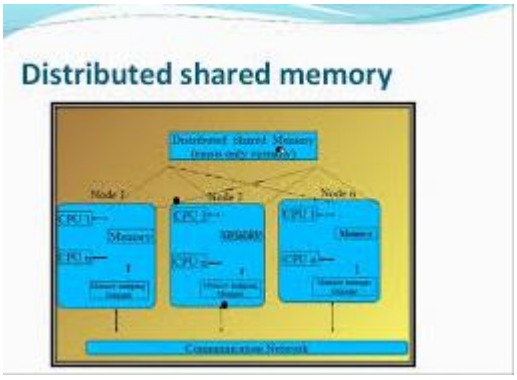
c) c) Update phase.

Define Starvation.

The prevention of a transaction ever being able to commit is called starvation.

GENERAL ARCHITECTURE OF THE DSM SYSTEM

The general architecture of distributed shared memory system is the representation and arrangement of various components that constitute the working paradigm used by different processes on different nodes to communicate by sharing a common virtual address space to fulfill services and operations.



The distributed shared memory can be built by interconnecting or organizing Introduction to DSM various nodes or simply systems in a distributed network arrangement. Each node has one or more processing units that are CPU and associated local memory.

All the constituent nodes or systems are connected with one another by a dedicated communication link which in turn is connected to a high-speed communication network. The distributed shared memory in contrast to physical memory is tightly coupled with the processor and can also be looked like a virtual address space build from various individual memories coupled with various connected processors. This shared memory acts like a global address space for all the processors of the distributed network and this address space can be as large as the collection of all the individual memory spaces that is local to each processor in each node.

A memory-mapping manager that is in each node of the architecture maps local memory onto the shared memory to constitute a virtual address space, which is global and accessible to all the connected nodes. This virtual memory space is partitioned into various blocks to facilitate better mapping. In order to overcome the latency issues associated with multiple shared accesses, the local memory of each node is also treated as a big cache memory of the shared address space accessible to each individual processor of a node. This local cache is mapped using the memory-mapping manager routine.

The access strategy of DSM architecture works when a process emerging from any node of the network tries to access some data from the shared memory. The request of data access is accepted by memory-mapping manager routine and the data requested is first searched on the cache that is on the local memory. If the data is found on the local address space the access is fulfilled without any latency.

However, if the data is not available on the local cache then the memory-mapping manager triggers a network fault and the control is passed to the underlying operating system. The operating system generates a request to the desired node that is the node whose local memory contains the desired data. The data found is later transferred to the requester nodes cache. This pattern of request-identify-retrieve or transfer is performed whenever required to fulfill access operations. However, this underlying architectural shift is not visible to user processes as for them the memory is like a shared global address space.

The caching of copies of data on the local memory avoids or reduces the access latency.

DESIGN AND IMPLEMENTATION ISSUES OF DSM

There are various issues and challenges associated with the design and implementation of a distributed shared memory system. The most prominent issues and challenges are mentioned as under:

Granularity: Whenever a request for data transfer/access on the DSMS is made, the extent of the data quota that can be standardized to be accessed or moved across the processors of various nodes on the network if there is network block fault. The selection of a particular standardized unit to be designated as a memory/data block is an important component in DSMS design

Structure of Shared-Memory Space:

The structure of the data block shared on the network among various nodes typically is influenced by the application type that DSMS is expected to support.

Memory Coherence and Access Synchronization:

When similar copies of the data are moved across the shared memory of DSMS. The coherence or the persistence of the data across different memories remains an issue because, whenever a modification is made it is necessary to update all the copies to ensure consistency in DSM. In DSMS, there can be concurrent data access requests from several processes in order to address consistent data accessibility over the DSMS, the access mechanism needs to be synchronized.

Data Location and Access: The mechanism of locating, retrieving and transferring of requested data in DSM by user process needs to be properly designed to respond to block faults effectively by availing consistent data image.

Replacement Strategy: In the situation like whole local cache is fully occupied and the process has to respond to any data transfer request generated by any node on DSMS. If the data requested is not the one that is currently in the local cache, the requested data needs to be retrieved from DSM and brought into local cache by replacing previously held content. Therefore, the replacement of the cached content in this replacement approach is a challenging issue in DSMS.

Thrashing: This is the problem that emerges in DSMS when two processes from two different nodes request the same data block. In this situation the data block is moved back and forth among these competing processes quickly, the quickness is as fast as it seems no data transfer was carried out.

Heterogeneity: If the nodes and their underlying architecture used to design DSMS are homogeneous then no heterogeneity issues can emerge. However, if the underlying environment is different then the heterogeneity takes place. The DSMS architecture design should facilitate positive and productive data communication when DSMS is heterogeneous in nature.

Parameters influencing Block Size Selection

Paging overhead: The feature of “locality of reference” attributed with shared memory programs to perform a data transactions on DSMS, the process is expected to access a larger component on the shared address space within a small quantum of time. This paging overhead associated to access large memory is comparatively less than the paging overhead required for small block size.

Directory Size: In order to maintain the log to record all the data transactions on the DSMS and the overhead associated with it, the larger block size is preferred in contrast to small block size. Small block size means more data transfers therefore, more log maintenance in comparison to large block size

with fewer transfers and less log maintenance on DSMS. **Thrashing:** As many processes may be concurrently accessing the same data reference present in a particular data block. This competing process from different nodes may be trying to update the data instance causes an exponential increase in data transfers over the network stalling the program execution efficiency. This current updating requests made by processes on the same data block causes thrashing. More data block has more changes of more thrashing overheads in comparison to small data blocks as small data may result in fewer data operations.

False Sharing: When two different processes from separate nodes requests to access two different data instances residing on the same data block causes no data to be transferred across the network have a direct impact on the productive aspect of DSMS. More data block size has more chances for false sharing in comparison to small data block size.

STRUCTURE OF SHARED MEMORY SPACE

The commonly used approach to build a shared memory space of a DSM system are:-

1. No Structuring
2. 2. Structuring by data type
3. 3. Structuring as a database

1. No Structuring: In most of the DSM systems the shared memory space is not structured but is simply an arrangement of liner array of words. The main advantage of this unstructured DSM space is its connivance in choosing any suitable page size to represent a particular data block. When there is no fixed structure for page size it is, therefore, easy to implement in designing DSMS.

2. Structuring by data type: In this approach, the shared memory layout is structured and the memory space is organized either as a collection of objects or as a collection of variables in the source language. Therefore, the granularity of the unit is also defined either as an object or a variable. Since

the behavior of object or variable is not fixed but is varying in nature and depends on the fundamental of application and its underlying language manifestations therefore, DSM systems use variable grain size. This behavior or dependency of grain size on the application nature creates overheads in the design and implementation of DSMS.

3. Structuring as a Database: In this particular approach of designing a DSMS, the shared memory is well structured like a database. The shared space in this approach is organized into ordered structures called as tuple space. Tuples are commonly designed as a row/column format. The data within the tuple space is accessed and addressed by the content that the tuples are holding. In order to perform any transaction over the network, the processes select the tuples by directly targeting the tuples by specifyingthe number of their fields and their values or types. Apart from tuple organization, the access mechanism is non-transparent which in contrast is transparent in other DSMS approaches.

CONSISTENCY MODELS

The consistency model describes the degree of consistency that is being maintained for the successful and correct data access. Consistency models are designed on the basis of certain protocols that the competing processes must follow to ensure consistency of the share data instances in DSMS. To provide consistent data to a different application in DSMS several approaches were considered to design various consistency models among them the most popular ones are discussed below:

Strict Consistency Model: The consistency models in real essence enforce “principle of coherence” on the DSMS to grant mechanism for consistent data. Strict Consistency model is considered as the strongest model approach that

strictly adheres to “principle of coherence”. The DSMS is said to be a strict consistency model if the process intended to read any variable from some data block on shared memory, the data variable to be accessed is the latest copy of the data variable written in shared memory. In a general context, it can be said that if there is any kind of update operation on any data variable that the latest copy of the data is instantly updated at all places of its existence. The implementation of strict consistency model requires the absolute global clock to synchronize the processes, variables or objects with persistence.

Sequential Consistency Model: It is proposed by Lamport is the design mechanism of shared memory where all the processes of the network has got a similar order to access the shared memory to execute different operations. However, there is no restriction on interleaving among the different access operations like read, write. Let’s consider that there are three processes to perform read, write and read operations. The DSMS that supports the implementation of sequential consistency model ensures that no operation on the shared memory that lasts till other previous operation is completed. Sequential consistent memory provides one-copy/single copy semantics as all the processing sharing the memory location will encounter the same data contents stored sequentially on DSMS.

Causal Consistency Model: It is proposed by Hutto and Ahamad represents a weakening of sequential consistency approach that it makes a refinement between events that are possibly causally related and those that are most certainly not. In other words, causal consistency represents that all the execution are the same as if causally-related read/write operations were executed in an order that reflects their causality. All concurrent operations may be seen in different orders. Memory reference operations that are not potentially causally related may be seen by different processes in a different order. Any two memory reference can be treated as casual if the first processes get influenced by another process in any way. A shared memory system is, therefore, said to support the causal consistency model if all the specific operation on shared memory are potentially causally related and are seen by all other processes in the same order. The implementation of Casual Consistency Model takes into account the dependability of processed with each

other which is achieved by maintaining a dependency graph for shared operations. **Pipelined Random-Access Memory Consistency Model:** It is proposed by Lipton and Sandberg provides again a weaker consistency semantics and is also known as FIFO consistency. All the processes see that all the write operation order made by some process looks different to another process only ensures that all write operations performed by a single process are seen by all other processes in the order in which they were performed as if all the write operations performed by a single process are in a pipeline. Write operations performed by different processes may be seen by different processes in different orders". In this model, the consistency is preliminarily posed on write operations.

Weak Consistency Model: It is proposed by Dubois et al. is the consistency approach where the "Synchronization accesses (accesses required to perform synchronization operations) are sequentially consistent. Before synchronization access can be performed, all previous regular data accesses must be completed. Before regular data access can be performed, all previous synchronization accesses must be completed. This essentially leaves the problem of consistency up to the programmer. The memory will only be consistent immediately after a synchronization operation". The write operations made by some processes is not necessary to be shown to another process. In order to fulfill the operational mechanism of weak consistency model following recommendations must be adopted.

All accesses made to synchronization variables must be performed under sequential consistency semantics.

All the necessary update or write operations on some data variable in shared memory must be completed before access to synchronization variable is permitted. All operations on the synchronization variable must be furnished before making access to any non-synchronized variable.

Release Consistency Model: Release consistency is essentially the same as weak consistency, but synchronization accesses must only be processor consistent with respect to each other. Synchronization operations are broken down into acquiring and release operations. All pending acquires (e.g., a lock

operation) must be done before a release (e.g., an unlock operation) is done. Local dependencies within the same processor must still be respected. Release consistency is a further relaxation of weak consistency without a significant loss of coherence.

Entry Consistency Model: Like other variants of release consistency model, it requires the programmer (or compiler) to use acquires and release at the start and end of each critical section, respectively. However, unlike release consistency, entry consistency requires each ordinary shared data item to be associated with some synchronization variable, such as a lock or barrier. If it is desired that elements of an array be accessed independently in parallel, then different array elements must be associated with different locks. When an acquisition is done on a synchronization variable, only those data guarded by that synchronization variable are made consistent.

Processor Consistency Model: Writes issued by a processor are observed in the same order in which they were issued. However, the order in which writes from two processors occur, as observed by themselves or a third processor, need not be identical. That is, two simultaneous reads of the same location from different processors may yield different results. General Consistency Model: A system supports general consistency if all the copies of a memory location eventually contain the same data when all the writes issued by every processor have completed.

DISCUSS ABOUT THRASHING

Thrashing is said to occur when the system spends a large amount of time transferring shared data blocks from one node to another.

Various conditions in DSMS that can result in page faults or may cause thrashing are discussed below: 1. Interleaved data access by a processor on

multiple nodes results into the movement of data block back and forth across these nodes. 2. Invalidation of data blocks with read-only permission just after their replication in other nodes.

Strategies to overcome the overhead caused by thrashing in DSMS

1. Implementation of an efficient local replacement algorithm.

2. Implementation of application-controlled locks to lock the data block from being accessed by other nodes for a short duration.

3. Locking the data block associated/accessed by some node in DSMS by barring another node to accesses or take away the data block until a specific or designated time period completes.

4. By modifying or training the cache coherence algorithm in DSMS as per the context related to a particular data block. In order works, it means that there is a need to have separate coherence algorithm/protocols for each data block based on its characteristics to reduce the overheads caused by thrashing.

ADVANTAGES OF DSM

Hide data movement and provide a simpler abstraction for sharing data. Programmers don’t need to worry about memory transfers between machines like when using the message passing model.

Easier to implement than RPC since the address space is the same Nodes implementing data blocks using sequential programming constructs/ paradigms directly run on DSMS.

In DSMS complex data structures or data blocks are migrated across nodes on network by simply passing reference by simplifying the algorithmic approach for distributed applications.

The principle of “locality of reference” facilities the movement the entire page containing the data requested rather than just transferring a small piece of data.

DSMS is comparatively cheaper architectural design approach than multiprocessor systems to perform parallel executions.

In DSMS the much larger virtual memory space is built by combining all the local memory associated with each local processor or node in DSMS. The formation of large shared memory space helps to reduce or overcome the overhead caused by disk latency like swapping in case of traditional distributed systems.

In DSMS large number of heterogeneous nodes can be connected to form DSMS network therefore, much larger shared-memory system accessed by nodes dynamically. While as in case of multiprocessor systems where main memory is accessed via a common bus topology limiting the size of the multiprocessor system.

The programs developed to share or access shared memory space in multiprocessors can run on DSM systems easily.

The migration of data blocks from one node to another node on DSMS is comparatively easy to handle as both the processors/processes are accessing the same shared address/ memory space.

SYNCHRONIZATION The basic building blocks of a distributed operating system are cooperation, exchange of data between different nodes of a distributed operating system. In order to exchange data between any two nodes, the computer system at the destination should accept the data that has been sent by a sender which is possible only when synchronization between the sender and receiver is implemented. The process of synchronization gives the

details of the timing between sender and receiver which helps the sender and receiver to communicate.

The process of synchronization can be categorized in two basic categories like blocking and non-blocking synchronization

Blocking: In every communication process a message is passed from a sender to a receiver. In the blocking synchronization method the sender after sending the message will wait for the acknowledgement from the receiver and during the wait period the sender will be blocked till acknowledgement is received. Similarly, the receiver after sending the acknowledgement will wait for a message from the sender in order to proceed further.

Non-Blocking: In the non-blocking synchronization method the sender after sending the message is not blocked and the sender will not wait for the acknowledgement from the receiver in order to proceed further with execution

Discuss about mutual Exclusion

A **mutual exclusion** (mutex) is a program object that prevents simultaneous access to a shared resource. This concept is used in concurrent programming with a critical section, a piece of code in which processes or threads access a shared resource.

In distributed systems, mutual exclusion is implemented using centralized, decentralized and token ring algorithms.

1 Centralized Algorithm

In a centralized algorithm, one of the many processes is elected as a coordinator. A coordinator can be a machine with the highest network address. When a process wants to enter a critical region, it sends a request message to

the coordinator. The message states the critical region that the process wants to enter and asks for permission. The coordinator replies with a message granting permission, if no other process is using the critical region in question currently.

However, a centralized algorithm also has certain drawbacks. It follows a centralized approach that has a single point of failure, i.e. the coordinator. If the processes block after making a request, then they cannot distinguish between a dead coordinator and permission denied state, as no message is sent in both the cases. A single coordinator can also cause performance bottleneck in large systems.

2. Distributed Algorithm

To avoid a single point of failure, distributed algorithms were developed, such as Ricart and Agrawala. This algorithm requires a complete ordering of the events in the system. A process wanting to enter a critical region sends a message containing its process number and current time to all the other processes, including itself. The messages sent are acknowledged either singly or using group communication.

This algorithm is associated with another problem that it must use either a group communication or every process must maintain a group membership list. The group membership list must include the processes entering a group, the processes leaving the group and the processes that have crashed.

2. **Token Ring Algorithm**

In a token ring algorithm, there is a bus network with unordered processes. In software, a logical ring is created and each process is assigned a space on the ring.

However, there are certain problems associated with this algorithm too. The token must be regenerated if it is lost. Even before that, it is difficult to detect that the token has been lost because the time of successive appearances of

the token on the network is unbounded. Furthermore, it is not necessary that the token is lost if it hasn't been spotted on the network for a long time. This can also mean that some process is still using it.

Discuss about Election Algorithms

Most distributed algorithms require one process to act as a coordinator, distributor or sequencer. Any process can take up this responsibility, but some algorithms need to determine or elect the coordinator process. If every process is the same, with no distinguishing characteristics, there is no way to select a process. Therefore, it is assumed that every process has a unique number or ID, such as the network address. Generally, the election algorithms attempt to trace the process with the highest process number and designate it as the coordinator. It is also assumed that each process knows the process ID of every other process, but they do not know which of the processes are active (currently up) and which are currently down. The aim of any election algorithm is to elect the new coordinator and make all the processes agree to it. Various election algorithms are as follows:

1. The bully algorithm

2. The ring algorithm

Bully Algorithm

In this algorithm, a process initiates an election when it fails to receive response from another process. The election is held by a process, P by performing the steps as follows:

1. An ELECTION message is sent to all the processes with higher process numbers than P.

2. 2. If no process sends a response, P wins the election and becomes the coordinator.

3. 3. If a higher numbered process answers, it takes over the job from P.

When a higher-numbered process receives an ELECTION message from a lower-numbered process, it sends an OK message indicating that it is alive and will take over. The receiver then holds an election in a similar way as explained above. Ultimately, all the processes give up, and one process that is left is the new coordinator. The new coordinator sends a message to all the other processes announcing its victory and telling them to start immediately.

If a process that was previously down comes back, it holds an election and if it is the highest-numbered process running, it will win and become the coordinator. Thus, the algorithm gets its name from the fact that biggest-numbered process wins and bullies the smaller ones; hence the name, bully algorithm**.**

Figure 9.9 shows the working of a bully algorithm.

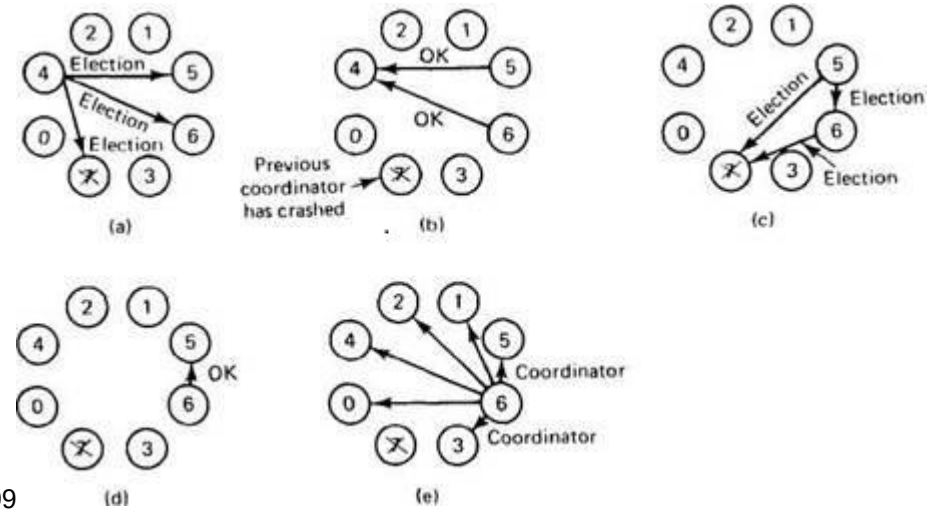


Fig99

In Figure 9.9, there are eight processes (0 to 7) in a group. Process 7 was the coordinator but it has crashed and therefore, a new coordinator needs to be elected. Process 4 is the first one to notice that the coordinator has failed. Therefore, it sends an ELECTION message to all the other process, higher than it, i.e. to processes 5, 6 and 7, as shown in Figure 9.9 (a). Since process 7 was previously dead, it does not respond, while processes 5 and 6 send an OK message, as shown in Figure 9.9 (b). As soon as process 4 gets a response, it

knows that its job is over, so it sits back and waits for the coordinator to be elected. As shown in NOTES Self-Instructional Material 107 Figure 9.9 (c), both processes 5 and 6 hold the election and send an ELECTION Synchronization message to higher processes, i.e. process 5 sends a message to processes 6 and 7, and process 6 sends a message to process 7. In Figure 9.9 (d), process 6 informs process 5 that it will take over and process 6 already knows that process 7 is dead as it does not receives a response from process 7. Now, process 6 knows that it is the winner and when it is ready to take over, it sends a coordinator message to all the other processes. When process 4 receives the coordinator message, it can continue with its operation, which it was previously trying to accomplish. In case, process 7 ever starts again, it will send a coordinator message to all others.

Ring Algorithm

A ring algorithm uses a ring structure without a token and assumes that each process knows its successor. Also, the processes are logically or physically ordered. When a process realizes that the coordinator is dead, it builds an election message, which contains its own process number and sends the message to its successor process. In case the successor is down, the sender process keeps on skipping the processes till it finds a running process. The sender keeps on adding its own process number to the list in the message, at every step.

In a matter of time, the initiating process gets back the message and it recognizes this event when it sees its own process number in the message. At this point of time, a coordinator message is circulated to inform every one of the new coordinator. When the coordinator message has been circulated once, it is discarded and the processes return to their work. Figure 9.10 shows the ring election algorithm.

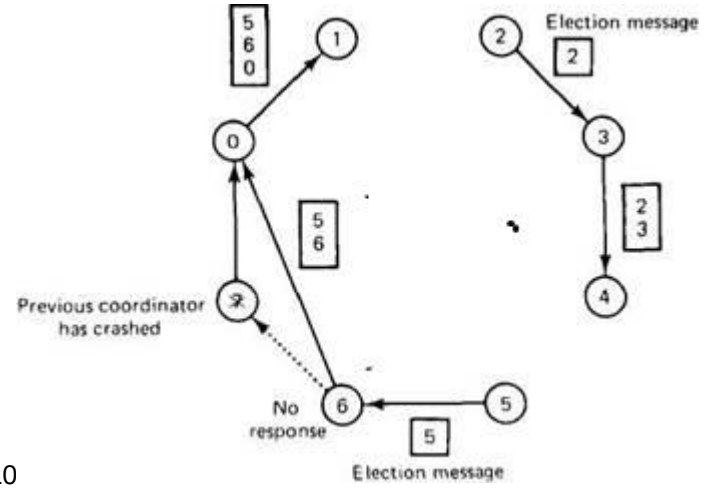


Fig9.10

In Figure 9.10, processes 2 and 5 simultaneously discover that the coordinator has crashed. Therefore, both these processes build an ELECTION message each, with their own process numbers and circulate it. In the end, both the messages will go round the ring and processes 2 and 5 will convert them into coordinator messages. Thus, in this way an extra message will circulate, which is of no harm. It just uses some extra bandwidth.

Explain about Distributed in deadlock

Deadlocks in distributed systems are similar to deadlocks in single-processor systems. There are two kinds of distributed deadlocks: communication deadlocks and resource deadlocks. A communication deadlock is a situation in which each member process is trying to communicate with another member process but is unable to communicate as they both wait for each other to answer a query. Resource deadlock is a situation in which member processes are arguing over exclusive access to I/O devices, files, locks or other resources. The various strategies that are used for handling deadlocks in distributed systems can be classified into four major groups: deadlock detection, prevention and avoidance.. Deadlock detection and recovery techniques allow deadlocks to occur and then apply certain methods to recover from the deadlock. These techniques are very difficult to implement. Due to the presence of atomic transactions, deadlock prevention is also possible in distributed systems. Finally, the deadlock avoidance techniques

acquire information in advance about which resource a process will claim at which stage of execution. Distributed operating system can assume that deadlock will never happen or rarely occur and fully ignore it.

Distributed Deadlock Detection

Since it is very difficult to find out the methods for preventing or avoiding distributed deadlocks, researchers have started dealing in detecting the occurrence of deadlocks in distributed systems. Deadlock detection detects the state of the system to determine whether a deadlock has occurred or not. It also helps processes to recover from the deadlock condition. The presence of atomic transactions in some distributed systems make a major conceptual difference. If a deadlock is detected in a conventional operating system, you can break the deadlock by killing one or more processes. When a deadlock is detected in a system, which is based on atomic transactions, then abort all the deadlocked processes. This means, one of the deadlocked processes is aborted and it is verified whether the deadlock is over or not. This procedure is continued until a system reaches safe state

Centralized deadlock detection

In a centralized deadlock detection algorithm, each machine has a resource graph for its own processes and resources and the deadlock detection coordinator maintains the resource graph for the entire system. When the coordinator detects a cycle, it destroys one process in order to break the deadlock. In distributed systems, due to delay in information many deadlock algorithms generate false deadlocks.

UNIT-IV

What is File Sharing?

When two or more users share the same file at the same time, it is called file sharing.

Define Naming Service.

It refers to the process of mapping user-defined file names with transparency.

What is Static map?

It is a very simple mechanism, which is used where the location of file or directory does not change.

. **What is the role of replication in distributed systems?**

Replication is defined as the maintenance of copies of data at multiple computers. It is a key to the effectiveness of distributed systems in that it can provide enhanced performance, high availability and fault tolerance.

What are the two basic file system used in distributed system?

- The Sun Network File System, NFS.
- The Andrew File System, AFS.

What are the file system modules?

Directory module: relates file names to file IDs.

File module: relates file IDs to particular files.

Access control module: Check permission for operation requested.

File access module: reads or writes file data or attributes.

Block module: accesses and allocates disk blocks.

Device module: performs disk I/O and buffering.

Write the Characteristics of file systems?

File systems are responsible for the organization, storage, retrieval, naming, sharing and protection of files. They provide a programming interface that characterizes the file abstraction, freeing programmers from concern with the details of storage allocation and layout.

What are the different forms of transparency are partially or wholly addressed by current file services?

Access transparency

Location transparency

Mobility transparency

Performance transparency

Scaling transparency

Define File service architecture of AFS?

An architecture that offers a clear separation of the main concerns in providing access to files is obtained by structuring the file service as three components – a flat file service, a directory service and a client module.

What is Andrew File System?

Andrew is a distributed computing environment developed at Carnegie Mellon University (CMU) for use as a campus computing and information system. The design of the Andrew File System (henceforth abbreviated AFS) reflects an intention to support information sharing on a large scale by minimizing client-server communication.

What is caching?

When a client requests a name lookup, the name resolution software consults its cache. If it holds a recent result from a previous lookup for the name, it returns it to the client; otherwise, it sets about finding it from a server. That server, in turn, may return data cached from other servers.

What are the two types of naming services in a distributed system? .

1. Multi-level mapping
2. Multi-valued mapping

List out the three ways to create file replication.

1. Explicit File Replication
2. Lazy File Replication
3. Group Communication Based Replicas

What is Fault tolerance?

The quality of a system to continue functioning even when some of its components have failed.

Define Atomicity.

It is a property of atomic transaction that ensures that each transaction either occurs completely or not at all.

Define Consistency. This means that if some invariants apply to a system before a transaction, they must hold true even after the transaction is complete.

Write a note on File Models

The basic file model is based on the structure used in a file system within a distributed file sharing environment and the same are categorized as structured and unstructured files. The second categorization of file models within a distributed file sharing environment is based on modifiability are mutable and immutable files.

(a) Structured File System: In this file system, the details of a file are known to the storage server where the files are stored. Every file in the system is a collection of records which are in an ordered sequence. The record is the lower most unit is this file system which can vary in size from one file to the other. The sharing of files in this file system are not as simple as unstructured file system. The structured file system is used using two methods as indexed and non-indexed method. The indexed file system stores the records in an indexed sequence where any record can be accessed by specifying the value of one or more key field and the same can be addressed by giving the values of the key fields. In structured file system method the records of a file are maintained using structures and one of the commonly used architecture is B-Tree. However, in non-indexed file system a record is traversed by mentioning the position of a record in a file.

(b) Unstructured File System:

(c) This file system is the simplest form of a file system where the details about a sub structure are not known to the storing server where the files are stored. The distributed operating system kernel does not require to know about the sub structures of or details of file and data stored on the storage servers which gives applications the access to understand the details of the sub structures of the data stored. Some of the examples where this method of storage was used are MS-DOS and UNIX.

(d) Another categorization of file models based on modifiability are given below:

(e) (a) Mutable Files: Once a file is stored on a storage server, the possibility of modifying the contents of a file is required. When the content of a file are modified the file is not re-created rather the existing file is updated by overwriting the existing file with a new file. This method is known as mutable file method. This method is commonly used in most of the operating system at present as the method reduces the overheads required to manage the number of in case every modification request re-creates a file again and again.

(f) (b) Immutable Files: In this file system the contents are modified by re-creating a file. In this method every modification requests will create a new file and the information about the previous versions of a file is stored as history of the file modified. A separate subroutine is used for managing the versions of a file which helps the operating system to find out the recent updated file. This increases the load on the operating system and increase the quantum of storage space required to store a file.

Write a note on File Accessing Models

The file accessing modes are the methods of accessing a file within a distributed file sharing environment and the list of the generally used file accessing modes is given below:

(a) **Accessing Remote Files:** In order to access a file in a distributed file sharing environment different modes are used one of them is access remote files method where a file is accessed remotely from any location which is having access to the network.

The two different methods of Access Remote Files are given below:

(i) Remote Service model:

In this method of file access the client requests for file access and the file is sent to server. The server processes the client's request and the output is

sent to client. The file is not sent to the client node rather the file is delivered to the server. The server processes all the requests and the output is deliver in the form of a message to the client. The communication between the client and the server is in the form of data packets. In this file access mode method the communication overheads & message overheads are more while communicating a request from a client to server and while communicating the output from a server to the client. The protocols for communication and file access need to be designed appropriately in order to minimize the overheads generated by communication and messages.

(ii) Data caching model: This model of file access mode helps in reducing the load of communication channel in comparison with Remote Service Model method of file access. In this model when a client has a file access requests, the availability of file is first checked locally on the node and if the file is not available locally then a copy of the file from the server is stored locally on the client node. The client node processes the files access request locally and generates the output from the locally stored file. The file is stored locally in the cache of the client node.

(b) Unit of Data Transfer:

The data caching model of file access mode a copy of the file from the server is stored locally on the client node cache. However, the size of the data packet transferred needs to be fixed in the design of the operating system. In this file access mode the categorization is done based on the unit of the size transferred and the same are given below: **(i) File Level Transfer Model:** In this file access mode type when a client requests for access to a file from the server, the complete file is transferred from the server to the client. This method reduces the file access at the server node and improves the performance and scalability of the distributed file sharing environment.

(ii) Block Level Transfer Model:

In this file access mode type when a client requests for access to a file from the server, the file blocks are copied from the server to the client. The file blocks are arranged in contiguous blocks where the size of the file block is fixed.

(iii) Byte Level Transfer Model:

In this file access mode type when a client requests for access to a file from the server, the data transfer happens in bytes which are copied from the server to the client. This file access method is more flexible in comparison with the previous methods of file access. However, the cache management becomes more difficult as the number of bytes in a file are more in number.

(iv) Record Level Transfer Model: , This file access mode will transfer the data of file in unit of records where a complete record will be transferred from the server node to the requesting node. This method of transfer is more suitable for structured file access method.

Write a note on File Replication

In a distributed file system, file replication is broadly used. File replication improves the availability of files to clients as clients can access the replica of the file stored at the nearest site in the network.

File replication is also important in case of system failures. Replicas of files are stored on the machines in which failure does not occur. These machines are linked with other replicas. When any of the machines on which the copy of a file is stored, crashes, then the machine, which contains another copy of this file, continues the processing.

File replication can be created in the following three ways:

Explicit file replication: A programmer controls the complete process of the replication of files. To create multiple copies of a file, a programmer needs the permission of the directory server. This directory server associates the address of each copy of file along with the file name.

Lazy file replication:

In a distributed system, the server controls the replication of files.In this type of replication, a programmer creates the copy of a file on the server. After

the copy is created on the server, the server further creates multiple copies of the same file on other servers also. The server creates multiple copies only when the system is not heavily loaded.

Group communication based replicas: The system calls are sent to all the servers that create multiple copies at the same time when the original was made.

File replication also leads to a consistency problem. Any update in one replica is reflected in all other replicas. A distributed operating system uses update protocol algorithms in order to ensure consistency between the replicas. The following are the two most commonly used protocols:

Primary copy replication algorithm: According to this protocol, updated replica is sent to the primary server that holds the master copy. This primary server makes some permanent changes in the master copy and issues commands to the secondary servers. These secondary servers hold the replicas of this file to make necessary changes.

Voting: According to this protocol, clients require permission from multiple servers that hold the replicas before performing any I/O operation, such as read or write on any of the replica. The file replication scheme improves the load-balancing feature of the distributed operating system: for example, if two processes require the same information, then one of the two processes can be sent to some other client machine to continue processing by using the replica of that information.

Discuss about Directory Structure

Directories are considered as symbolic tables of files that store all the related information about the file it holds along with the content. This information includes file attributes, location type and access privileges. They are also known as containers for files. A directory is itself a file that is owned by the distributed operating system.

The following operations can be performed on different entries in a directory:

Searching a file:

Whenever a file is referenced, the directory must search for the related entry.

Create a file: An entry for every newly created file needs to be added in the directory.

Delete a file: Whenever a file is deleted, related entry should be removed from the directory.

List directory: List of files in a directory is shown whenever a user requests for it.

Rename a file: The name of the file should be changeable when the use of file changes or its location changes.

Update directory: Whenever a file attribute changes, its corresponding entry needs to be updated. Based on these entries and its operations, the structure of directories can be organized in different ways.

The three most common structures for organizing a directory are as follows:

- Single-level structure
- Two-level structure
- Hierarchical structure

Discuss the fault tolerant design techniques

Redundancy

Redundancy is often used to build fault-tolerant systems. Take the case of a passenger aero plane which typically has more than one engine. In the event of an engine failure, the other takes over. In other words, a redundant engine is installed to be used in the event of primary engine failure.

Physical redundancy

This type of redundancy, where more than one instances of a critical system is deployed in known as physical redundancy. Patterns exist that are employed to address fault tolerance using physical redundancy.

Temporal Redundancy

It involves recording a series of events that happen in a system, and playing them back in case of a failure.. Temporal redundancy is used in scenarios where transient faults or sporadic faults occur.

Information redundancy

In this case, extra data is added to the transmission which helps in detecting and correcting errors at the receivers end.

Fault Categories

Faults can be categorized into the following types:

Permanent: A permanent fault is one that remains until the defective part is changed. As an example, a failure in a database server due to a hard disk fault remains uncorrected until the bad disk is replaced with a good one followed by restoration of data.

Transient: A transient fault happens once and then does not reappear; for example, a cellular phone fails to detect the mobile network in places such as tunnels and underground railway systems, but once the person comes out in the open, the system re-establishes connection with the nearest base station.

Sporadic:

As the term suggests, these types of faults occur on and off.

Failure Models

In accordance with a specific failure classification scheme, failures can be categorized by their types into crash failures, omission failures, timing failures, response failures and arbitrary failures.

Crash Failure: This occurs when a server stops due to a system malfunction. Before the crash happened, the server was functioning correctly. A server program termination by the operating system due to an illegal memory address access would be a typical example of a crash. To recover from such situations, the server program needs to be restarted.

Omission Failure: Such failure takes places when a server does not send its response to a request. It might very well be the case that the server never received the request to start with. It might also be due to a transient failure in the communication media, which resulted in the total loss of network traffic. An omission failure can also occur in a situation when the server has failed to transmit the response after processing the request.

Timing Failures: These are noticed when a system is not able to respond to the requestor within a predefined time period.

Response Failures: These are more severe forms of failures. When a system responds incorrectly to a request.

Arbitrary Failures: These are the most serious forms of failures. These failures are also known as **Byzantine failures**. Byzantine refers to the Byzantine General's problem, in which a number of generals separated by distances need to decide upon whether to attack the enemy or retreat.

Discuss about Atomic design principles.

Atomic transactions provide synchronization at a higher level of abstraction.

Transaction Model

A transaction model consists of some processes capable of failing at random. The communication is unreliable in the sense that the messages can be lost.

Stable storage

In the transaction model, a stable storage is required that can survive any crashes except for calamities, such as floods and earthquakes. A pair of ordinary disks can be used to implement a stable storage.

Transaction primitives

Special primitives are required for programming transactions. These primitives must either be supplied by the operating system or by the language run-time system. Some of the examples of primitives are as follows:

BEGIN_TRANSACTION: It marks the start of a transaction. END_TRANSACTION: It terminates a transaction and tries to commit it. ABORT_TRANSACTION: It is used to kill a transaction and restore the previous values.

READ: It is used to read data from a file or object.

WRITE: It is used to write data from a file or object.

Explain the properties of Transactions.

Atomic: It ensures that each transaction either occurs completely or not at all. Also, if at all it happens, it should occur in a single indivisible action. This means that other processes must not be able to view the intermediate states of the transaction.

Consistent: It says that a system should be consistent before and after a transaction. This means that if some invariants apply to a system before transaction, they must hold true even after the transaction is complete: for

example, in a banking system, the law of conservation of money must always hold true, i.e. money should not be lost during a transaction. □

Isolated: This means that transactions should seem to be occuring serially independent of each other. If two transactions are running simultaneously to each other and to the other proceses, the final result should seem as if the transactions occurred sequentially.

Durable: It specifies that if a transaction has been committed, no matter what, the transaction proceeds forward and its results become permanent. Thus, the results cannot be undone, once a transaction has been committed.

UNIT-V

What are the key principles of security?

The key principle of security is the following:

1. Make sure you have the latest security updates & patches
2. Install anti-virus software
3. Install anti-spyware software
4. Use a personal firewall
5. Password advice

What is the difference between a mono-alphabet cipher and a polyalphabetic cipher?

Mono-alphabetic cipher is a mono-alphabetic cipher is a substitution cipher in which the cipher alphabet is fixed through the encryption process. All of the

substitution ciphers we have seen prior to this handout are mono-alphabetic; these ciphers are highly susceptible to frequency analysis.

Polyalphabetic Cipher is a polyalphabetic cipher is a substitution cipher in which the cipher alphabet changes during the encryption process.

What is Login Spoofing?

This is a technique for collecting usernames and passwords of users of the system by an attacker who is an ordinary user of the system.

Define Virus

A virus is a program fragment that is attached to legitimate popular programs like games or other utilities with the intention of infecting other programs.

List the various viruses.

Companion virus

Executable program virus

Memory resident virus

Boot sector virus

Device driver virus

Macro virus

Sources code virus

What is Worm?

A worm is also like virus, but it can automatically spread to other computers through the Internet.

What is Cryptography?

Cryptography is the process of representing information using secret codes for providing security and confidentiality of information in a system.

Define Encryption.

Encryption is the process of transforming information (referred to as plaintext) using an algorithm (called cipher) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key.

What is authentication?

When a user logs on to a computer, the operating system wants to determine who the user is. This process is called user authentication.

Define Authorization.

Authorization is that task after authentication to ensure whether the subject has the right to access a secure entity in the system.

Attacks from Inside the System

An insider is a person who has logged into a computer using legitimate username and password. In a system with long and special symbol based passwords, breaking them to login may be difficult.

A person who has logged in can also exploit the system vulnerabilities (bugs or loop holes) to gain entry into other users' area including administrators and work in the system with their privileges. These attackers are also called crackers, as they break the password system to gain unauthorized entries into other users' area. Attacks by insiders include:

Trojan Horses

Login Spoofing

Logic Bombs

Trap Doors

Buffer Overflow

Trojan Horses A Trojan Horse is a program that appears legitimate and innocent but performs illicit activity when it is run, such as stealing passwords, making the system more vulnerable to future entry or simply destroying programs or data on the hard disk.

When users download and execute the program, the Trojan is executed do all nasty things like removing files or reading passwords or send information to the attacker's site.

Login Spoofing

This is a technique for collecting usernames and passwords of users of the system by an attacker who is an ordinary user of the system. The attacker or cracker logs in to the system and executes a program which displays a login window exactly looking like that of normal login window of the system.

If the users start the login sequence by pressing a key combination that the user program cannot catch, this spoof attack can be bypassed or prevented. Windows system uses control-alt-del keys combination for this purpose.

Logic Bombs

This is a piece of code that programmers (who are current employees) of a company secretly inserted into the companies production operation system or companies applications.

For example, as long as the programmer logs in everyday or alternate days, the system functions normally. If the programmer did not login for, say two continuous days, the logic bomb goes off leading to things like erasing files at random and malfunctioning of the whole system.

Trap Doors

Trap Door is another security hole caused by the programmer. This is done by secretly inserting some code to the operating system (or application) code that bypass some normal check.

For example, a programmer may add code to the login program to login using a name ‘SAHARA’ whatever be the password string. So, the programmer can login to computers of any company that loads this operating system or application.