Name - Ankit Goayl

Roll No. 17103011

Branch – CSE 3rd year

ANS 1: A distributed system consists of several nodes, each with its own clock, running at its own speed. Because of the nonzero drift rates of all clocks, the set of clocks of a distributed system do not remain well synchronized without some periodic resynchronization. This means that the nodes of a distributed system must periodically resynchronize their local clocks to maintain a global time base across the entire system. The slow and fast clocks drift in opposite directions from the perfect clock. Therefore, of two clocks, if one is slow and one is fast, at a time after they were synchronized, the maximum deviation between the time value of the two clocks will be 2p*delta t. Hence, to guarantee that no two clocks in a set of clocks ever differ by more than 'delta', the clocks in the set must be resynchronized periodically, with the time interval between two synchronizations being less than or equal to delta/2p.

ANS 2: The RPC mechanism is an extension of the procedure call mechanism in the sense that it enables a call to be made to a procedure that does not reside in the address space of the calling process. The called procedure (commonly called remote procedure) may be on the same computer as the calling process or on a different computer. In case of RPC, since the caller and the callee processes have disjoint address spaces (possibly on different computers), the remote procedure has no access to data and variables of the caller's environment. Therefore the RPC facility uses a message-passing scheme for information exchange between the caller and the callee processes

ANS 3: Stubs, which provide a perfectly normal (local) procedure call abstraction by concealing from programs the interface to the underlying RPC system.
Stubs can be generated in one of the following two ways:
1. Manually. In this method, the RPC implementor provides a set of translation functions from which a user can construct his or her own stubs. This method is simple to implement and can handle very complex parameter types. 2. Automatically. This is the more commonly used method for stub generation. It uses Interface Definition Language (JDL) that is used to define the interface between a client and a server. An interface definition is mainly a list of procedure names supported by the interface, together with the types of their arguments and results. This is sufficient information for the client and server to independently perform compile-time type checking and to generate appropriate calling sequences. However, an interface definition also contains other information that helps RPC reduce data storage and the amount of data transferred over the network.
The function of the stub is to provide transparency to the programmer-written application code. On the client side, the stub handles the interface between the client's local procedure call and the run-time system, marshaling and unmarshaling data, invoking the RPC run-time protocol, and if requested, carrying out some of the binding steps. On the server side, the stub

provides a similar interface between the run-time system and the local manager procedures that are executed by the server.

ANS 4. Lamport's logical clocks – If A -> B then C(A) < C(B) – Reverse is not true!! • Nothing can be said about events by comparing time-stamps! • If C(A) < C(B), then ?? • Need to maintain causality – If a -> b then a is causally related to b – Causal delivery: If send(m) -> send(n) => deliver(m) -> deliver(n) – Capture causal relationships between groups of processes – Need a time-stamping mechanism such that: • If T(A) < T(B) then A should have causally preceded B. causal ordering of messages in vector clocks - maintaining the same causal order of message receive events as message sent - that is: if Send (M1) → Send(M2) and
Receive(M1) and Receive (M2) are on the same process than Receive(M1) → Receive(M2) u example above shows violation - do not confuse with causal ordering of events.

ANS 5. AFS (Andrew File System) is a distributed, networked file system. Most users access AFS through the Unix Timeshare. The UCSC web-hosting service and course lockers are based on AFS storage.
Features and Benefits
■ File backups — AFS data files are backed up nightly. Backups are kept on site for six months.
■ File security — AFS data files are protected by the Kerberos authentication system.
■ Physical Security — AFS data files are stored on servers located in the UCSC Data Center.
■ Reliability and availability — The AFS servers and storage are maintained on redundant hardware.
■ Authentication — AFS uses Kerberos for authentication. Kerberos accounts are automatically provisioned for all UCSC students, faculty and staff. Kerberos uses the CruzID 'blue' password.
■ Quota — The default quota is 500MB per user. Quotas are increased automatically up to 10GB; at that point, a quota increase can be requested through the ITS Support Center. (This is scripted to run several times daily; the quota increase is not instantaneous.)
Sun's Network File System :The earliest successful distributed system could be attributed to Sun Microsystems, which developed the Network File System (NFS). NFSv2 was the standard protocol followed for many years, designed with the goal of simple and fast server crash recovery. This goal is of utmost importance in multi-client and single-server based network architectures because a single instant of server crash means that all clients are unserviced. The entire system goes down.

ANS 6: The problem of thrashing may occur when data items in the same data block are being updated by multiple nodes at the same time, causing large numbers of data block transfers among the nodes without much progress in the execution of the

application. While a thrashing problem may occur with any block size, it is more likely with larger block sizes, as different regions in the same block may be updated by processes on different nodes, causing data block transfers that are not necessary with smaller block sizes.
The relative advantages and disadvantages of small and large block sizes make it difficult for a DSM designer to decide on a proper block size. Therefore, a suitable compromise in granularity, adopted by several existing DSM systems, is to use the typical page size of a conventional virtual memory implementation as the block size of a DSM system. Using page size as the block size of DSM system has the following advantages [Li and Hudak 1989]:
1. It allows the use of existing page-fault schemes (i.e., hardware mechanisms) to trigger a DSM page fault. Thus memory coherence problems can be resolved in page fault handlers.
2. It allows the access right control (needed for each shared entity) to be readily integrated into the functionality of the memory management unit of the system.
3. As long as a page can fit into a packet, page sizes do not impose undue communication overhead at the time of network page fault.
4. Experience has shown that a page size is a suitable data entity unit with respect to memory contention.

ANS 7: Chandy-Misra-Haas's distributed deadlock detection algorithm is an edge chasing algorithm to detect deadlock in distributed systems. In edge chasing algorithm, a special message called probe is used in deadlock detection. A probe is a triplet (i, j, k) which denotes that process P_i has initiated the deadlock detection and the message is being sent by the home site of process P_j to the home site of process P_k. The probe message circulates along the edges of WFG to detect a cycle. When a blocked process receives the probe message, it forwards the probe message along its outgoing edges in WFG. A process P_i declares the deadlock if probe messages initiated by process P_i returns to itself.

Other terminologies used in the algorithm:

1. **Dependent process:**
   A process P_i is said to be dependent on some other process P_j, if there exists a sequence of processes P_i, P_{i1}, P_{i2} , P_{i3} …, P_{im} , P_j such that in the sequence, each process except P j is blocked and each process except P_i holds a resource for which previous process in the sequence is waiting.

2. **Locally dependent process:**
   A process P_i is said to be locally dependent on some other process P_j if the process P_i is dependent on process P_j and both are at same site.

Data structures:
A boolean array**,** dependent i **.** Initially, dependent i [j] is false for all value of i and j. dependent i [j] is true if process P_j is dependent on process P_i.

Algorithm:

Process of sending probe:

1. Process of sending probe:
2. Else for all P_j and P_k check following condition:
    - Process P_i is locally dependent on process P_j
    - Process P_j is waiting on process P_k
    - Process P_j and process P_k are on different sites.

If all of the above conditions are true, send probe (i, j, k) to the home site of process P_k .

On the receipt of probe (i, j, k) at home site of process P_k:

1. Process P_k checks the following conditions:
    - Process P_k is blocked.
    - dependent k [i] is false.
    - Process P_k has not replied to all requests of process P_j.

If all of the above conditions are found to be true then:

1. Set dependent k [i] to true.
2. Now, If k == i then, declare the P_i is deadlocked.
3. Else for all P m and P n check following conditions:
    - Process P_k is locally dependent on process P_m and
    - Process P_m is waiting upon process P_n and
    - Process P_m and process P_n are on different sites.

4. Send probe (i, m, n) to the home site of process P_n if above conditions satisfy. Thus, the probe message travels along the edges of transaction wait-for (TWF) graph and when the probe message returns to its initiating process then it is said that a deadlock has been detected.

ANS 8: When two users (persons or programs) of two different nodes want to communicate securely by using a symmetric cryptosystem, they must first share the encryption! decryption key. For this, the key must be transmitted from one of the two users to the other user. However, there is no special transmission medium for the key transfer and the key must be transmitted using the same insecure physical medium by which all exchanged messages are transmitted. This requires that the key must itself be encrypted before transmission because if the key is compromised by an intruder while being transmitted over the insecure medium, the intruder can decrypt all encrypted messages exchanged between the two users. Therefore, a circularity exists in symmetric cryptosystems. This circularity can only be broken through prior distribution of a small number of keys by some secure means. The usual approach is to use a server process that performs the job of a key distribution center (KDC). Each user in the system shares with the KDC a prearranged pair of unique keys.

ANS 9: In a symmetric cryptosystem, either both the encryption key (Ke) and decryption key (Kd) are the same or one is easily derivable from the other. Usually, a common key (K) is used for both enciphering and deciphering. For security, it is important that the key of a symmetric cryptosystem be easily alterable and must always be kept secret. This implies that the key is known only to authorized users. Symmetric cryptosystems are also known shared-key or private-key cryptosystems.In an asymmetric cryptosystem, on the other hand, the decryption key (Kd) is not equal to the encryption key (Ke) . Furthermore, it is computationally impractical to derive Kd from Ke . Because of this property, only Kd needs to be kept secret and K, is made publicly known. Asymmetric cryptosystems are also known as public-key cryptosystems.