DISTRIBUTED OPERATING SYSTEMS
ASSIGNMENT

VASU NEGI
UFID: 8495-3933

Generals Problem:

The two generals have to agree to attack the enemy at the same time. The enemy is however sitting in the middle valley between two generals and also can capture the message sent by the generals. The generals need to attack at the same time, where comes two scenarios, one where the message can get captured by the enemy or the other message is received. However, the sender general also needs to know whether the message is received or not. Unless you have some sort of communication, there can never be an agreement. So, the problem is to how these might come to an agreement to launch the attach at the same time, whilst also tracking that any message was received or not.

Synchronous/**Asynchronous**:

You always communicate at a particular time. That can be used to reach consensus. Synchronous is harder to implement, as different cores can have different clocks in the same CPU. In asynchronous, the communication does not happen at the same time.

Bounded/ **Unbounded**:
The hardware is designed on bounded constraints and these chips/systems work on these constraints. However, these bounded constraints are expensive for example racks in supercomputers.

Ordered/Unordered:
Just like in TCP, if the packets are not in order, there is retransmission. Hence, ordered is essential for communication.

Unicast/Multicast:
Multicast communication is tricky and harder to implement.

Flat Groups versus Hierarchical Groups:

Flat groups: all group members communicate with every group member. However, the larger the network, the more complicated the communication gets.

Hierarchical Groups: The members form a hierarchy, in which a group member becomes the coordinator for specific set of members.

Agreement in Faulty Systems:

In distributed systems, where components may fail, and the information depends on whether the component is faulty. Here, a component can arbitrarily seem defective to the system and may present different situations to the network.

K-fault tolerance:

The system can sustain the fault tolerance up to k number of faulty components. However, deciding the component is faulty or not as voting also becomes a problem when there are malicious components who can also vote and isolate the component.

Reliable Multicast:

The message has to delivered to a set of participants. The harder the constraints the harder the implementation of the system. What multicast means, there is best effort to send the message to send the messages to all the participants in the group. When discussing the multicast, we define how the components join a group, and how the messages are delivered in the set.

Hierarchical Feedback Control:

Imagine you don't have two generals but many generals in a distributed system. In these large systems, the system needs to have some sort of relationship between each component. This type of system can be used to have fault tolerance using leaders for each small group.

Virtual Synchrony:

Virtual Synchrony provides a model for managing repeated set of machines and communication between them. A component in each group can either join it or leave it.

Message Ordering:

Unordered: No specific order

FIFO-ordered: First In First Order: Individual sequence number may be used. However, more memory is required to hold the packets for longer period of time.

Causally ordered: Any sort of causality is maintained, where messages are delivered in a fashion to maintain causality.

Totally ordered: same order everywhere.

Three Phase commit:

This is an iteration over the two-phase commit, where it handles the edge case not handled by the two-phase commit. Each component requires to have no single state which can make a direct transition to a commit or abort and, also not to a final decision from which direct transition to a commit.

Checkpointing:

Checkpointing provides fault tolerance in distributed systems. It is similar to game's checkpoints where a checkpoint representing the saved state is stored to provide fault tolerance in the system.