BLOCKCHAIN

Unit3: Class 8

6. Consensus Algorithm – – Byzantine Fault Tolerance System

Byzantine Generals Problem

- We imagine that several divisions of the Byzantine army are camped outside an enemy city, each division commanded by its own general. The generals can communicate with one another only by messenger. After observing the enemy, they must decide upon a common plan of action. However, some of the generals may be traitors, trying to prevent the loyal generals from reaching agreement.

The generals must have an algorithm to guarantee that

1. All loyal generals decide upon the same plan of action.
2. A small number of traitors cannot cause the loyal generals to adopt a bad plan.

A leader general and a group of other generals surround a castle, the only way to ensure victory is by coordinating a simultaneous attack. Since they're dispersed around the castle, they must send messages to relay the attack time. There is however, loyal and traitor generals, and there's no way to find out.

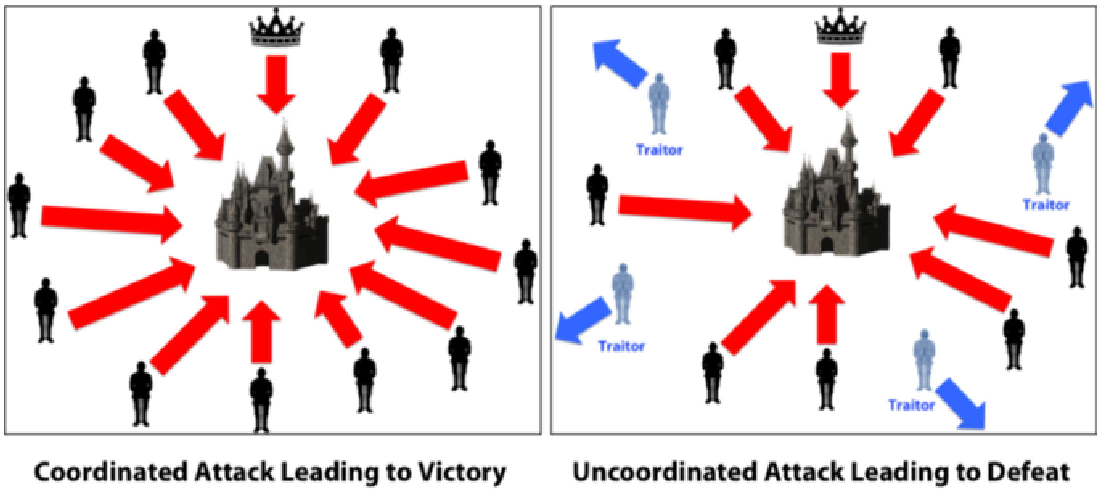
Traitor
Traitor
Traitor
Traitor
Coordinated Attack Leading to Victory
Uncoordinated Attack Leading to Defeat

Fig. 1

A Byzantine fault is a condition of a computer system, particularly distributed computing systems, where components may fail and there is imperfect information on whether a component has failed. In a Byzantine fault, a component such as a server can inconsistently appear both failed and functioning to failure-detection systems, presenting different symptoms to different observers. It is difficult for the other components to declare it failed and shut it out of the network.

Byzantine node

- A node which misbehaves in a distributed decentralized network is called as Byzantine node. Node sends different messages to different peers.

Byzantine fault tolerance

- All nodes participating in distributed network should be fault tolerant to such Byzantine nodes and the fault tolerance is achieved via various algorithms like byzantine fault tolerant and Practical Byzantine Fault tolerance algorithm.
- Paxos and Raft can tolerate up to N/2− 1 number of crash faults

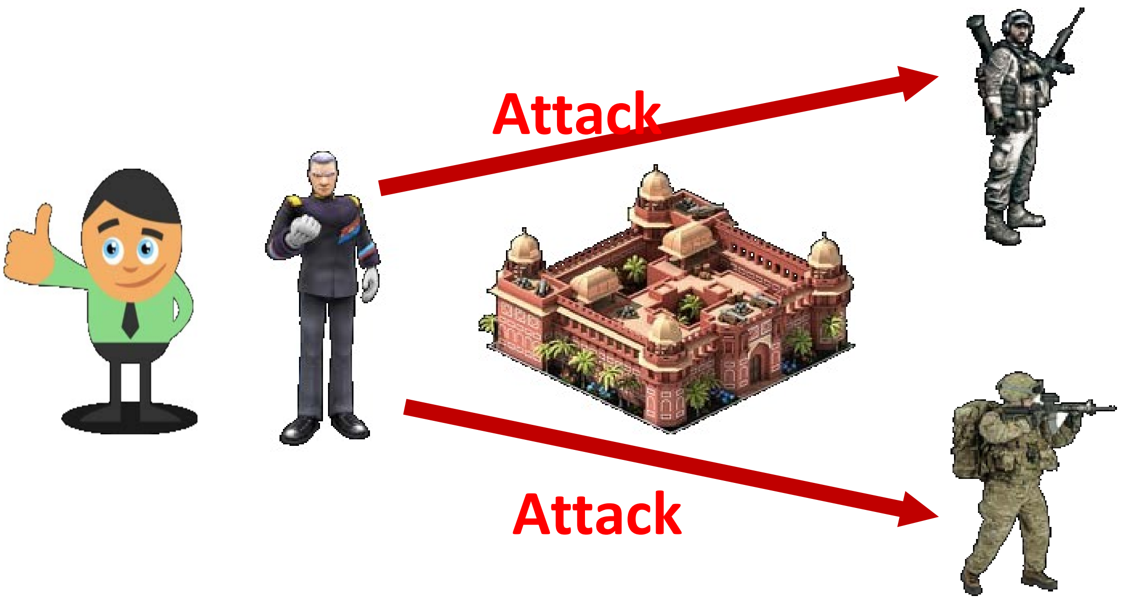

Fig.2

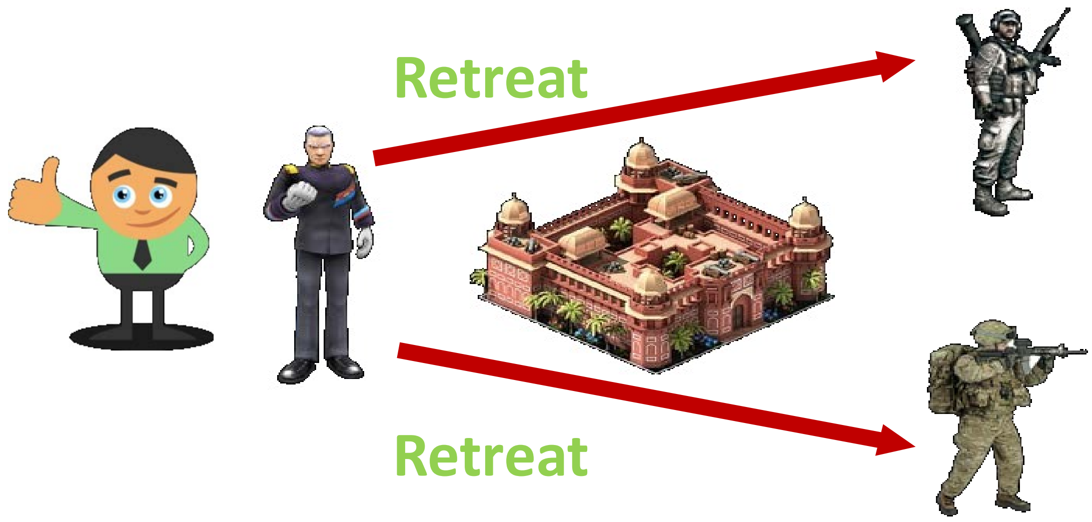

Fig.3

Generals send the attack message to one troop and generals send the retreat message to another troop as in Fig.3. if the general becomes faulty, then it becomes difficult for a system to reach consensus. We call this problem as a byzantine general problem or byzantine fault tolerant problem. We will try to develop a fault tolerant architecture where the system will be able to tolerate this kind of byzantine faults.

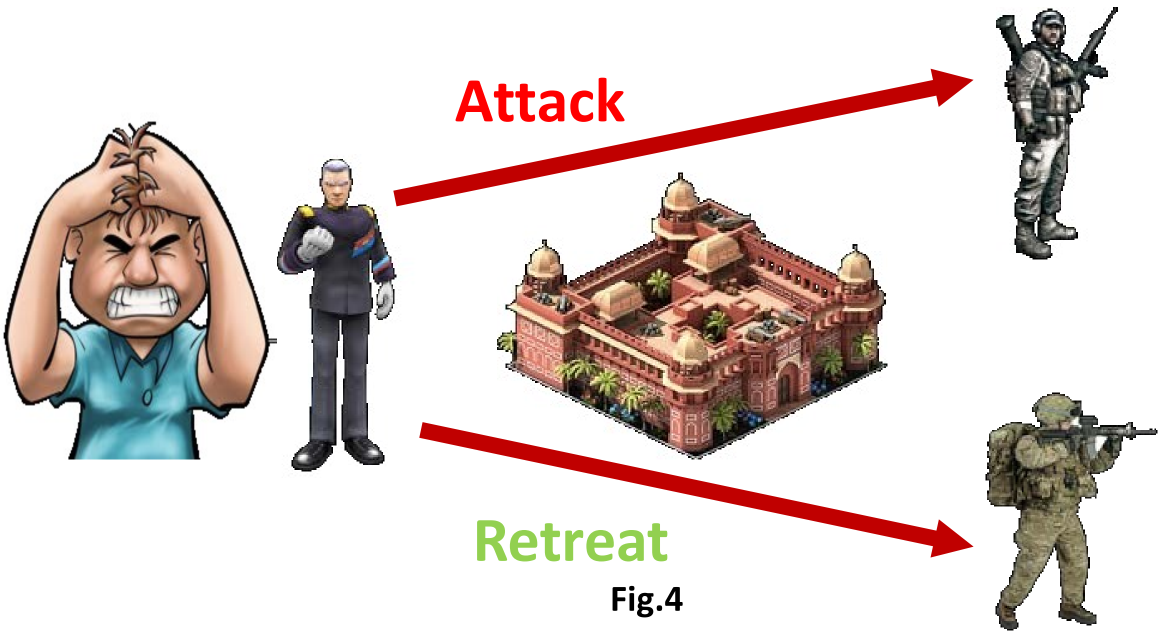

Fig.4

Three Byzantine Generals Problem: Lieutenant Faulty

The commander sends the message to the lieutenants. And the lieutenant can share the messages among themselves and try to find out whether the commander is faulty or whether the lieutenant is faulty. In this architecture we assume that the lieutenant is faulty. The lieutenant may send different messages from what it hear. But the commander is a correct commander.

- Round1:
  - Commander correctly sends same message to Lieutenants
- Round 2:
  - Lieutenant$_1$ correctly echoes to Lieutenant$_2$
  - Lieutenant$_2$ incorrectly echoes to Lieutenant$_1$

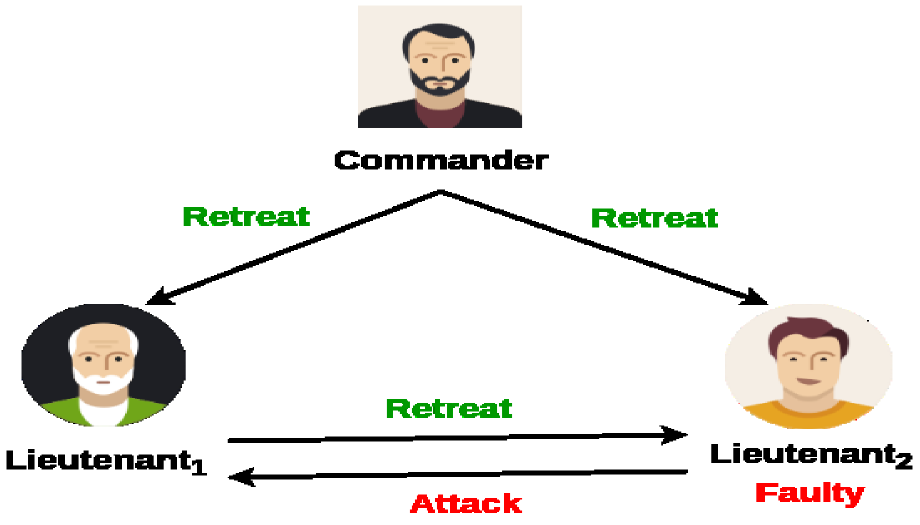

Fig. 5

- Lieutenant$_1$ received differing message
- By integrity condition, Lieutenant$_1$ bound to decide on Commander message
- What if Commander is faulty?

Three Byzantine Generals Problem: Commander Faulty

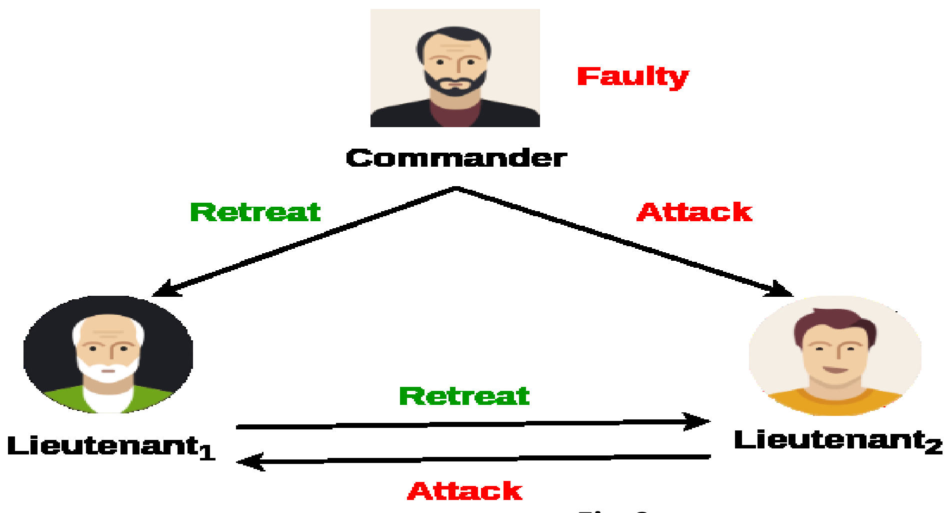

Fig.6

- Round 1:
  - Commander sends differing message to Lieutenants
- Round 2:
  - Lieutenant$_1$ correctly echoes to Lieutenant$_2$
  - Lieutenant$_2$ correctly echoes to Lieutenant$_1$
- Lieutenant$_1$ received differing message
- By integrity condition, both Lieutenants conclude with Commander's message
- This contradicts the agreement condition
- No solution possible for three generals including one faulty.
- if we try to apply this majority voting principle what we see that with 3 byzantine
- The majority principle does not work, you get equal voting for attack and retreat. we will not be able to solve the byzantine general problem with 3 generals where we have one commander and 2 different lieutenants.

Four Byzantine Generals Problem: Lieutenant Faulty

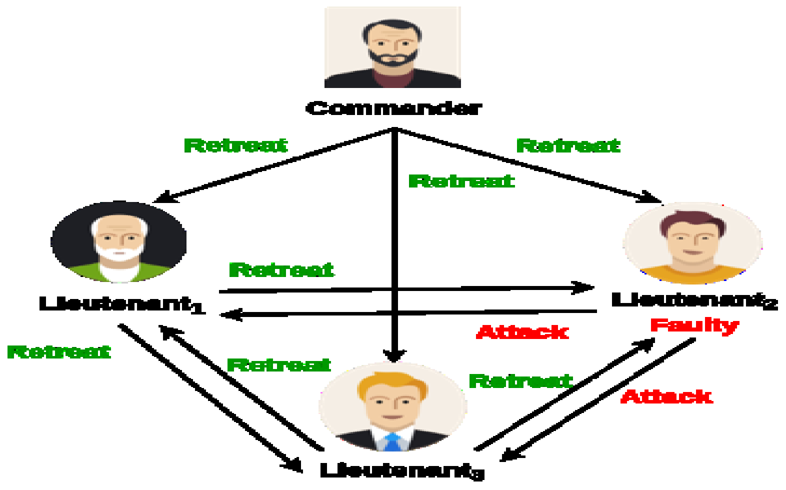

Fig.7

- Round 1:
  - Commander sends a message to each of the Lieutenants
- Round 2:
  - Lieutenant$_1$ and Lieutenant$_3$ correctly echo the message to others
  - Lieutenant$_2$ incorrectly echoes to others
  - Lieutenant$_1$ decides on *majority*(Retreat,Attack,Retreat)= Retreat
  - Lieutenant$_3$ decides on *majority*(Retreat,Retreat,Attack)= Retreat
- out of the 3 lieutenants if one lieutenant is faulty, then we will be able to correctly decode the message.

Four Byzantine Generals Problem: Commander Faulty

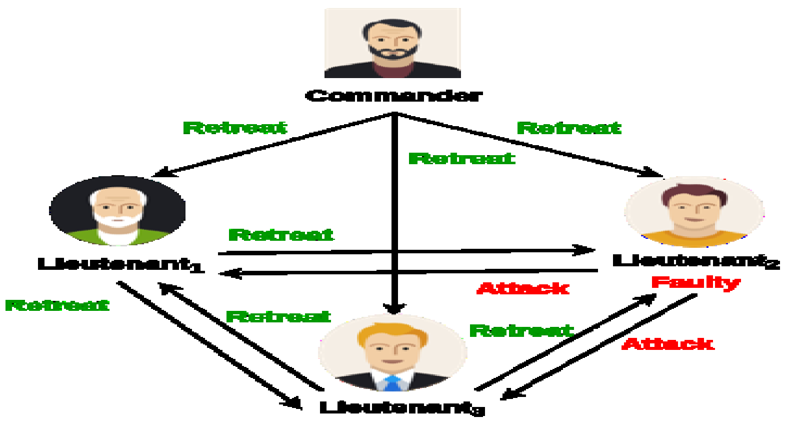

Fig.8

- Round 1:
  - Commander sends differing message to Lieutenants
- Round 2:
  - Lieutenant$_1$, Lieutenant$_2$ and Lieutenant$_3$ correctly echo the message to others
  - Lieutenant$_1$ decides on *majority*(Retreat,Attack,Retreat)= Retreat
  - Lieutenant$_2$ decides on *majority*(Attack,Retreat,Retreat)= Retreat
  - Lieutenant$_3$ decides on *majority*(Retreat,Retreat,Attack)= Retreat
- if the commander is faulty, in that case the system will come to the consensus with the value that the commander has proposed to majority of the lieutenants.

Byzantine Generals Model

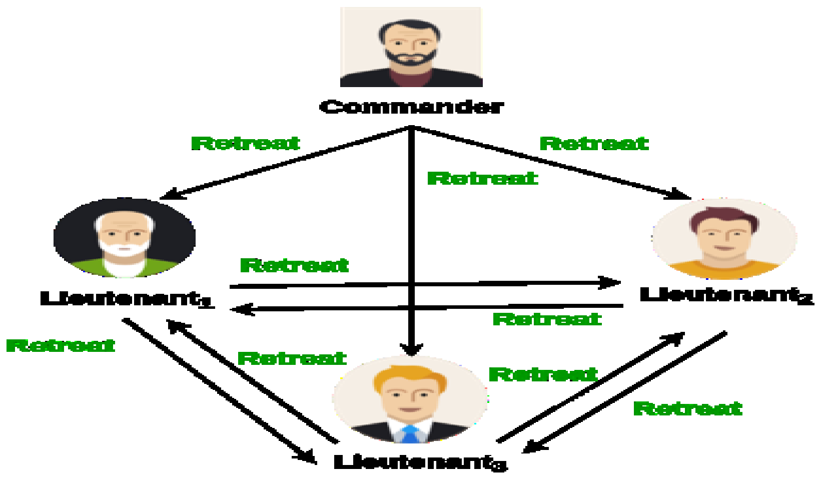

Fig.9

- *N* number of process with at most *f* faulty
- Receiver always knows the identity of the sender
- Fully connected, Reliable communication medium
- Synchronous system, closed model
- This model help us to design an algorithm for the permissioned blockchain environment, system is fully connected, the communication is a reliable communication medium and the system is synchronous. we are trying to develop an algorithm which is synchronous in nature.

Lamport-Shostak-Pease Algorithm

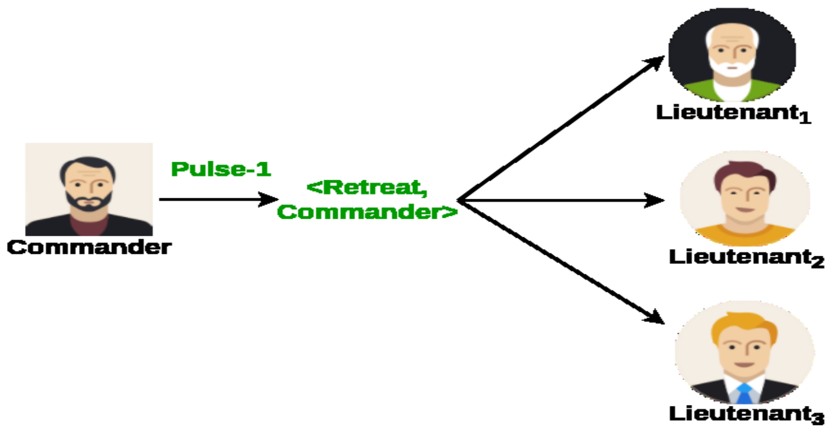

Fig.10

- The commander sends the message at pulse one. So, the pulse one is the initial pulse when the commander sends the message to all the lieutenants.
- This base condition we have 2 parameters here. N is the number of processes that are there in the system. And t is the algorithm parameter. That denotes the individual rounds. t equal to 0 indicates that you are in pulse 0 when the commander is sending a message to all the lieutenants. So, the commander it decides on it is own value, whether to go for a retreat or to go for an attack.
- At pulse 0 commander broadcasts this retreat message. Consider n equal to 3, because we have 3 different process among which we are trying to reach to the consensus. So, we send the 3 messages to 3 different lieutenants. Now that is the base condition for the commander.
- It is a pulse one message means that the message is coming from the commander. If it is no that means, it is not a pulse one message, then you cannot take any decision, because you do not know that from where the message is coming from. If it is a pulse one message; that means, the first message in the system, and the message is coming from the commander, then you accept what the commander is saying. Otherwise the system goes to the undefined state.
- Base Condition:

  Broadcast(N,t=0)

  – N: number of processes

  – t: algorithm parameter
- Commander decides on its own value

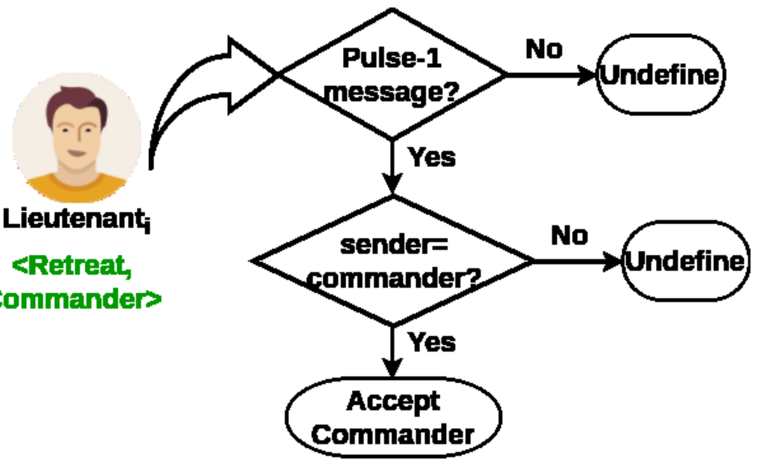

Fig.11

- Base Condition:

  Broadcast(N,t=0)

  – N: number of processes

  – t: algorithm parameter

- Lieutenants decision by sender matching

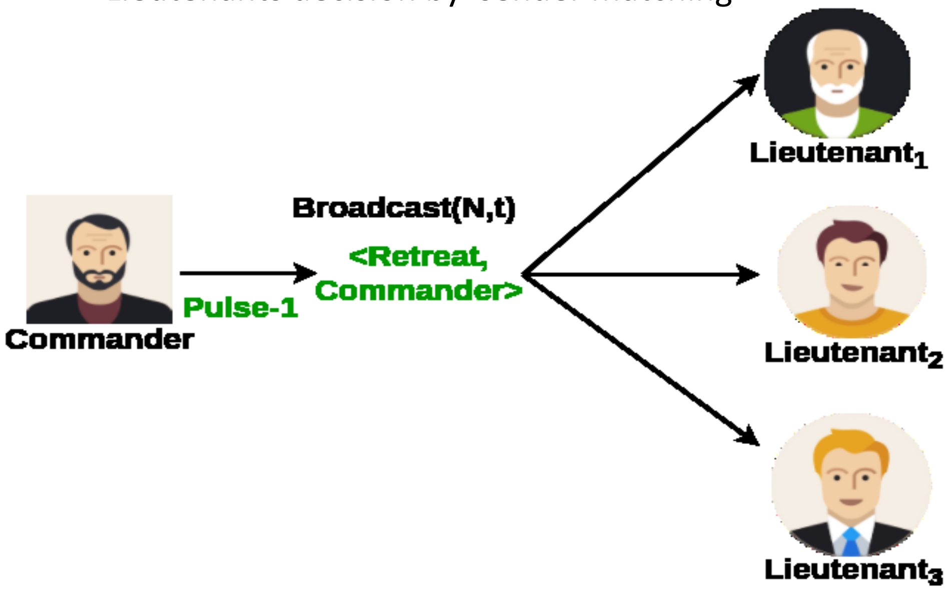

Fig. 12

The commander it sends this message to all the lieutenants in the tth round.

- General Condition:

  Broadcast(N,t)

  - N: number of processes
  - t: algorithm parameter

- Only commander sends to all lieutenants

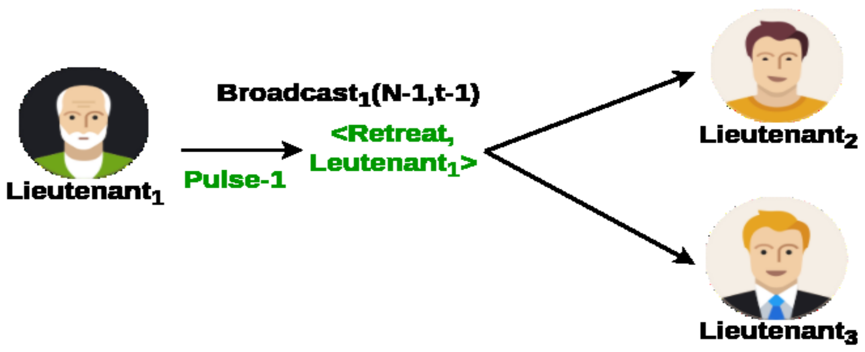

Fig. 13

- General Condition:

  Broadcast(N,t)

  - N: number of processes
  - t: algorithm parameter

- All lieutenants broadcast their values to the other lieutenants except the senders

  you have N number of lieutenants in the system. Then after nth round you are getting the messages from all the individual lieutenants. the system is synchronous, so you are known to get the messages from all other lieutenants who are there in the system after nth round. So, once you receive all this messages, then you can apply the majority voting principle. So, every individual lieutenant they apply the majority voting principle.

- There is no solution for 3m +1 generals, >m traitors