1.什么是拜占庭将军问题，什么是共识?

拜占庭将军问题也被称为“拜占庭容错”，是 Leslie Lamport（2013 年的图灵讲得主）用来为描述分布式系统一致性问题（Distributed Consensus）在论文中抽象出来一个著名的例子。

例子的大意是这样的：

拜占庭帝国想要进攻一个强大的敌人，为此派出了 10 支军队去包围这个敌人。这个敌人虽不比拜占庭帝国，但也足以抵御 5 支常规拜占庭军队的同时袭击。这 10 支军队在分开的包围状态下同时攻击。他们任一支军队单独进攻都毫无胜算，除非有至少 6 支军队（一半以上）同时袭击才能攻下敌国。他们分散在敌国的四周，依靠通信兵骑马相互通信来协商进攻意向及进攻时间。困扰这些将军的问题是，他们不确定他们中是否有叛徒，叛徒可能擅自变更进攻意向或者进攻时间。在这种状态下，拜占庭将军们才能保证有多于 6 支军队在同一时间一起发起进攻，从而赢取战斗?

拜占庭将军问题中并不去考虑通信兵是否会被截获或无法传达信息等问题，即消息传递的信道绝无问题。Lamport 已经证明了在消息可能丢失的不可靠信道上试图通过消息传递的方式达到一致性是不可能的。所以，在研究拜占庭将军问题的时候，已经假定了信道是没有问题的。

这里有两个问题。

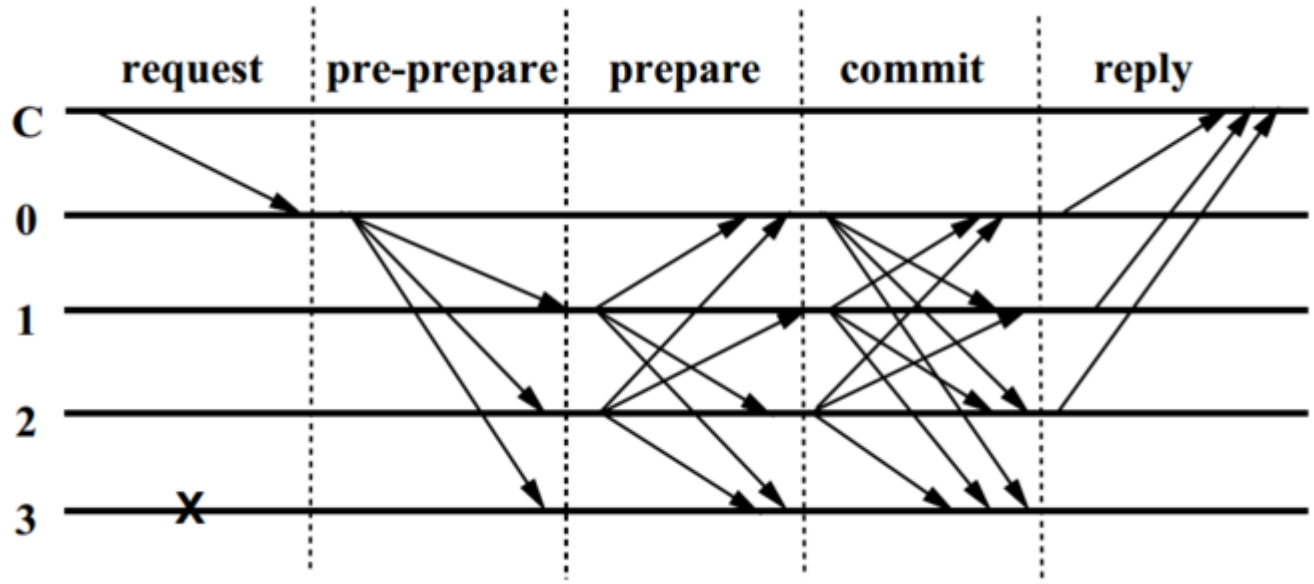
一是将军中可能出现叛徒，叛徒不仅可能给糟糕的策略投票，还可能选择性地发送投票信息。假设有 9 位将军投票，其中 1 名叛徒。8 名忠诚的将军中出现了 4 人投进攻，4 人投撤离的情况。这时候叛徒可能故意给 4 名投进攻的将领送信表示投票进攻，而给 4 名投撤离的将领送信表示投撤离。这样一来在 4 名投进攻的将领看来，投票结果是 5 人投进攻，从而发起进攻；而在 4 名投撤离的将军看来则是 5 人投撤离。最后的结果是军队步调不一，可能会输掉这场战役。

二是由于将军之间需要通过信使通讯，信使成了关键的一环。叛变将军可能通过伪造信件来以其他将军的身份发送假投票。即便所有将军都保持忠诚，也不能排除信使被敌人截杀，甚至被敌人间谍替换等情况。

因此很难通过保证人员可靠性（叛徒）及通讯可靠性（信使）来解决问题。

共识: 所有参与者的一致意见，意味着每个人都接受并支持这些决定。

2.讲述拜占庭容错算法的共识流程



PBFT 算法流程

1. REQUEST：

客户端 c 向主节点 p 发送<REQUEST, o, t, c>请求。o: 请求的具体操作，t: 请求时客户端追加的时间戳，c：客户端标识。REQUEST: 包含消息内容 m，以及消息摘要 d(m)。客户端对请求进行签名。

2. PRE-PREPARE：

主节点收到客户端的请求，需要进行以下交验：

a. 客户端请求消息签名是否正确。

非法请求丢弃。正确请求，分配一个编号 n，编号 n 主要用于对客户端的请求进行排序。然后广播一条<<PRE-PREPARE, v, n, d>，m>消息给其他副本节点。v：视图编号，d 客户端消息摘要，m 消息内容。<PRE-PREPARE, v, n, d>进行主节点签名。n 是要在某一个范围区间内的[h, H]，具体原因参见*垃圾回收*章节。

3. PREPARE：

副本节点 i 收到主节点的 PRE-PREPARE 消息，需要进行以下交验：

a. 主节点 PRE-PREPARE 消息签名是否正确。

b. 当前副本节点是否已经收到了一条在同一 v 下并且编号也是 n，但是签名不同的 PRE-PREPARE 信息。

c. d 与 m 的摘要是否一致。

d. n 是否在区间[h, H]内。

非法请求丢弃。正确请求，副本节点 i 向其他节点包括主节点发送一条<PREPARE, v, n, d, i>消息, v, n, d, m 与上述 PRE-PREPARE 消息内容相同，i 是当前副本节点编号。<PREPARE, v, n, d, i>进行副本节点 i 的签名。记录 PRE-PREPARE 和 PREPARE 消息到 log 中，用于 View Change 过程中恢复未完成的请求操作。

4. COMMIT：

主节点和副本节点收到 PREPARE 消息，需要进行以下交验：

a. 副本节点 PREPARE 消息签名是否正确。

b. 当前副本节点是否已经收到了同一视图 v 下的 n。

c. n 是否在区间[h, H]内。

d. d 是否和当前已收到 PRE-PPREPARE 中的 d 相同

非法请求丢弃。如果副本节点 i 收到了 2f+1 个验证通过的 PREPARE 消息，则向其他节点包括主节点发送一条<COMMIT, v, n, d, i>消息，v, n, d，i 与上述 PREPARE 消息内容相同。<COMMIT, v, n, d, i>进行副本节点 i 的签名。记录 COMMIT 消息到日志中，用于 View Change 过程中恢复未完成的请求操作。记录其他副本节点发送的 PREPARE 消息到 log 中。

5. REPLY：

主节点和副本节点收到 COMMIT 消息，需要进行以下交验：

a. 副本节点 COMMIT 消息签名是否正确。

b. 当前副本节点是否已经收到了同一视图 v 下的 n。

c. d 与 m 的摘要是否一致。

d. n 是否在区间[h, H]内。