Gossip协议

Gossip 协议，顾名思义，就像流言蜚语一样，利用一种随机、带有传染性的方式，将信息传播到整个网络中，并在一定时间内，使得系统内的所有节点数据一致

Gossip 协议存在的原因?

- 为了实现 BASE 理论中的“最终一致性原则”
- 两阶段提交协议和 Raft 算法需要满足“大多数服务节点正常运行”原则，如果希望系统在少数服务节点正常运行的情况下，仍能对外提供稳定服务，这时就需要实现最终一致性。
- 最终一致性是指系统中所有的数据副本在经过一段时间的同步后，最终能够达到一个一致的状态。

Gossip 的三板斧

直接邮寄（Direct Mail）

- 就是直接发送更新数据，当数据发送失败时，将数据缓存下来，然后重传
- 直接邮寄虽然实现起来比较容易，数据同步也很及时，但可能会因为缓存队列满了而丢数据
- 只采用直接邮寄是无法实现最终一致性的

反熵（Anti-entropy）

反熵是一种通过异步修复实现最终一致性的方法

- 反熵指的是集群中的节点，每隔段时间就随机选择某个其他节点，然后通过互相交换自己的所有数据来消除两者之间的差异，实现数据的最终一致性：
- 在实现反熵的时候，主要有推、拉和推拉三种方式
- 因为反熵需要节点两两交换和比对自己所有的数据，执行反熵时通讯成本会很高，所以不建议你在实际场景中频繁执行反熵，并且可以通过引入校验和（Checksum）等机制，降低需要对比的数据量和通讯消息等。
- 虽然反熵很实用，但是执行反熵时，相关的节点都是已知的，而且节点数量不能太多，如果是一个动态变化或节点数比较多的分布式环境（比如在 DevOps 环境中检测节点故障，并动态维护集群节点状态），这时反熵就不适用了

谣言传播（Rumor mongering）

- 谣言传播，广泛地散播谣言，它指的是当一个节点有了新数据后，这个节点变成活跃状态，并周期性地联系其他节点向其发送新数据，直到所有的节点都存储了该新数据
- 谣言传播非常具有传染性，它适合动态变化的分布式系统