Chapter 8: The trouble with Distributed Systems

- Unreliable Networks
  - timeout : after some time you give up waiting and assume that the response is not going to arrive
    - there's no correct value for timeouts
    - need to be set experimentally
- Clocks
  1. time-of-day-clocks
     - returns the current date and time according to some calendar
     - usually synchronized with NTP (a timestamp from one node is the same as a timestamp on other nodes)
  2. Monotonic clocks
     - suitable for measuring a duration (time interval) eg. timeout, a server's response time,
     - doesn't assume synchronization between different nodes' clocks and is insensitive to slight inaccuracies of measurement.

- Relying on Synchronized Clocks
  - it's important to monitor the clock offsets between all the machines
  - any node whose clock drifts too far from the others should be declared dead and removed from the cluster
- Logical clocks V.S Physical clocks
  - logical clocks are based on incrementing counters rather than an oscillating quartz crystal
    - a safer option for ordering events
  - time-of-day & monotonic clocks that measure actual elapsed time

- State of any node is determined by majority of votes in distributed systems
  - ↳ quorum
    - decisions require some minimum number of votes from several nodes in order to reduce the dependencies on any one particular node
- fencing token
  - a number that increases every time a lock is granted
  - can be used to prevent any unwanted situation in locking
- Byzanine Faults
  - behaviours such as a node claims to have received a message when in fact it didn't.

System Model and Reality

3) system models for timing issues

an abstraction that describes what things an algo may assume

1. Synchronous model
   - assumes bounded network delay, bounded process pauses, and bounded clock error.
   - not realistic model
2. Partially synchronous model
   - assumes that a system behaves like a synchronous system most of the time, but it sometimes exceeds the bounds for the network delay.
   - realistic model
3. Asynchronous model
   - an algo is not allowed to make any timing assumptions (not having a clock)
   - very restrictive model

3 models for node failures

1. Crash-stop faults
   - can also may assume that a node can fail in only one way.
   - a node may suddenly stop responding and is gone forever.
2. Crash-recovery faults
   - nodes may crash at any moment, and may start responding again after some time
   - nodes are assumed to have stable storage that is preserved across crashes, while in memory state is to be lost.
3. Byzantine (arbitrary) faults
   - nodes may do anything (such as trying tricking and deceiving other nodes)

- safety - nothing bad happens
- liveness - something good eventually happens