Let’s learn CAP (Consistency, Availability, Partition tolerance) theorem and its importance.

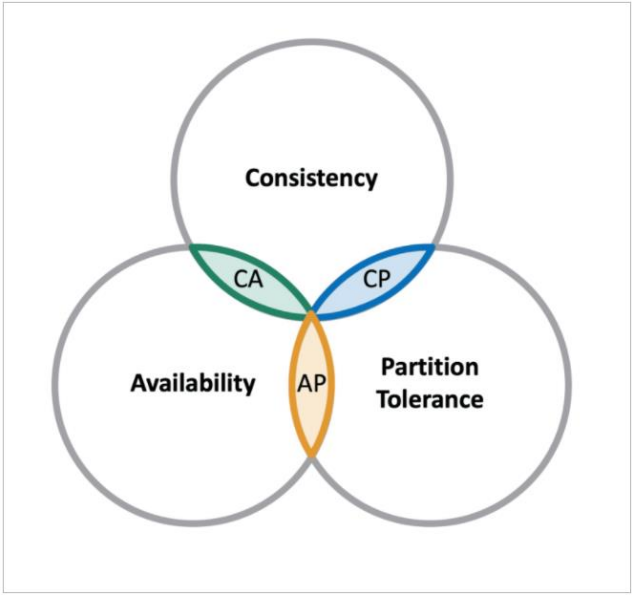
Consistency: Every read receives the latest writes from any server/node in the distributed system. [read my previous post **System Design-002** on consistency]

Availability: Every client request to a server must result in a response. High availability can be achieved by scaling the system to meet customer demands and make them resilient to failure.

Partition tolerance: The term ‘partition’ is confusing here, that actually means that the servers fail to communicate internally. The examples of partition are: loss of messages sent from one server to another, a crashed server (network partition) etc.

To achieve partition tolerance, the system should be able to handle the loss of messages sent internally or a random server crash. *A Partition tolerant system must function despite the communication breakdown between the servers.*

CAP Theorem: *a shared-data system can have at most two of the three following properties: **C**onsistency, **A**vailability, and tolerance to network **P**artitions.* [Dr. Eric Brewer]



In practice, a distributed system **cannot** avoid server crash or network failures. So, they are always designed by considering that such failures can happen. That means, practically you can only build CP (Favor Consistency) or AP (favor Availability) but not Consistency and Availability both.

Does that mean a 100% available and strictly consistent system doesn't exist? (add a comment if you know the answer)

Want to Read more in-depth?

1. Amazon's Highly Available Key-value Store: http://s3.amazonaws.com/AllThingsDistributed/sosp/amazon-dynamo-sosp2007.pdf

2. S3-strong consistency: https://www.allthingsdistributed.com/2021/04/s3-strong-consistency.html