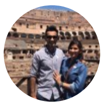
# CAP theorem explained



Shankar Vasudevan
May 10, 2018 · 2 min read

If you have a software application that needs to maintain any form of state over time, chances are you are using a database of some sort. If you want to mitigate the risk of having a single point-of-failure, you probably have a distributed database. These two requirements of *maintaining state* and *being distributed* present an inherent challenge.

One of the central concepts of distributed systems is CAP theorem. CAP theorem is a limitation that is imposed on all distributed databases. CAP stands for the following:

- *Consistency* — Data in different nodes is always in the same state
- *Availability* — Any request always returns a non-error response
- *Partitioning* — The system continues to operate even if communication between a subset of nodes on the network has failed.

Any distributed system must sacrifice one of the above three principles.

## Example

To illustrate, lets set an example where you have a distributed database with 2 nodes (A and B).

1. Both nodes initially have the state 1
2. The database receives a write request (state=2)
3. The database writes state=2 to Node A
4. The database receives a read request

At this time, Node A has state=2 and Node B has state=1, so the system has a trade-off between:

- *Availability*: Respond immediately and deal with Node A and Node B are in different states
- *Consistency*: The system does not respond until Node A and Node B are in the same state.

If a system wants to fulfil consistency and availability, it cannot be distributed. This is a fundamental limitation in all distributed systems.

You can have any two but not all three!

Cap Theorem Database Distributed Systems Software Development Programming