Storage and Redundancy in Distributed Systems

A storage device is a hardware component used primarily for data storage. It provides a way for computers to save data, either temporarily or permanently. Examples of storage devices are flash drives and hard drives. These devices store various types of data like videos, documents, photos, and raw data.

Persistence: Persistence is the ability of an object or state to remain unchanged even after the process that created it has stopped or the computer has shut down. When data needs to persist, it is saved in a non-volatile storage location like a hard drive, instead of a temporary file or volatile memory like RAM.

Types of Storages

Storage is one of a computer system's core components and can be categorized into many types, but there are two primary types:

Volatile Storage (Memory)

A storage device that requires a continuous supply of electricity for data storage/retention. This storage serves as the primary storage for temporarily storing data and managing the workloads of applications. Cache memory and random access memory are examples of nonvolatile storage (RAM).

Non-Volatile Storage

A storage device that preserves digital data even though it is turned off or electrical power is not supplied. This is often referred to as a secondary storage device and is used for I/O operations involving permanent data storage. A hard disc, USB storage, and optical media are examples of volatile storage.

What is Redundancy?

In distributed systems, Redundancy is the duplication of critical components (web server, database server, etc.) to increase reliability and availability. This concept provides a backup or failsafe, ensuring that the system continues to operate even if few components fail. The term redundancy is used because if everything is working correctly, the duplicate device or competent does nothing and is redundant.

Redundancy comes into play when we need to prevent a single point of failure (a point that can lead to system failure). By duplicating or adding components, the system becomes more resilient and less prone to failure. For example, consider a system that includes two identical web servers that are managed by a load balancer. Traffic from clients is evenly distributed between the two servers, but if one fails, the load balancer redirects all traffic to the remaining functioning server. In this way, redundancy eliminates the single point of failure and enhances the reliability of the system.

Passive Redundancy

In passive redundancy, we duplicate components in a system without actively using them. When the primary component is functioning correctly, duplicated components stay in a standby or passive state. During the passive state, the redundant component monitors the primary component's status and waits for a failure event. Once a failure is detected, the redundant component will take over the task of the failed component.

Note: In some distributed systems, the decision for the standby component to take over from the primary can be made by other components or depend on the system architecture.

- In case of a failure, the system must ensure that the state of the redundant component is sufficiently fresh before resuming services. So it is the responsibility of the primary to continuously inform

standby components to update their state. This is important to prevent data discrepancies between the primary and standby components.
- Periodically forcing switchovers (once a day or once a week) can increase the availability of the system. This practice ensures that the standby components are ready to take over when needed.
- Sometimes, passive redundancy may not provide instant recovery because there can be delays in detecting failures and switching to the redundant component. So it may require other fault-tolerant techniques like active monitoring, regular testing of the redundant components, and fault detection mechanisms.

When to use passive redundancy?

- When scalability is not a primary concern or workload is expected to remain stable.
- When downtime or failure can have significant consequences.
- When system can tolerate a slight delay in failover or recovery.
- When the failover process requires manual intervention.

Some distributed systems requirements directly align well with passive redundancy. For example, when data retrieval speed is not critical (data archiving systems), the passive state of redundant storage devices can be used for backup and recovery purposes.

Active Redundancy

In active redundancy, multiple identical components are simultaneously active in parallel, executing tasks concurrently and sharing the workload. In other words, they all actively contribute to the system's operations at the same time and work in a coordinated manner.

- We can use load balancing among redundant components to distribute the workload. So active redundancy reduces the likelihood of a single point of failure.
- Redundant components can work in parallel. This can improve

performance because the system can handle a larger volume of work in a given time.
- The redundant components are synchronized in the same state. Due to this, the switching time from a failed component to a redundant one is typically minimal (in milliseconds) in the case of a failure. In other words, active redundancy ensures smooth failover by automatically transferring the workload from the failed component to the remaining active components.

There are some critical points to consider:

- Active redundancy requires careful coordination and fault detection mechanisms to ensure that redundant components operate in a synchronized manner.
- When multiple components concurrently access shared resources, active redundancy should maintain consistency and avoid conflicts.
- Active redundancy generally requires more resources and can be more complex to implement compared to passive redundancy. But the increased reliability and continuous operation make it a preferred approach in many critical systems.

When to use active redundancy?

- When systems demand uninterrupted operation (high-availability environments).
- When systems require fast response and minimal latency.
- When significant processing power is required (high traffic situations).
- When seamless scalability is required.
- When rapid recovery is required in the event of a failure.

Redundancy vs. Replication

In distributed systems, the terms "redundancy" and "replication" are often used interchangeably. Both concepts involve having multiple nodes, components, or processes in a system, but they serve different purposes.

- The focus of redundancy is to provide backup components to handle failures, while the focus of replication is to create and maintain consistent copies of data across multiple components.
- The aim of redundancy is to enhance fault tolerance and availability i.e. minimize downtime and ensure continuous operation. But the aim of replication is to ensure data consistency, availability, and performance.
- Replication can be a part of redundancy because redundant components often require synchronized copies of data to operate effectively.

Some critical ideas to think!

- How does the choice of storage device impact system performance, reliability, and scalability? Think about factors like access times, throughput, latency, durability, etc.
- What mechanisms are used to ensure data consistency in active redundancy?
- What are the challenges in implementing fault detection mechanisms?
- How can we measure and quantify the reliability of redundant systems?
- How automated monitoring and alerting systems work to detect faults?
- What are the best practices for system maintenance in redundant systems?

Enjoy learning, Enjoy system design!