1. **Compare passive replication with active replication approach. Also discuss with a technique that make the distributed system highly available. - 2069 Chaitra**

**Ans:**

|  |  |
| --- | --- |
| **Active Replication** | **Passive Replication** |
| Can be only used for deterministic process | Can be used by non- determinsiric process also |
| Client’s request is processed by all servers | Only one server processes the client’s request known as primary server. |
| No coordination required as all replicas process same request in same order. | Primary coordinates with other replicas by sending update information to backups. |

|  |  |
| --- | --- |
| Response is not delayed in case of failure. | Response is delayed in case of failure. |
| All replicas send back result to client. | Only primary send back answer to client. |
| Requires more processing power. | Requires less processing power |

we need to know how we can improve the availability of a system. The first and straightforward solution is your system should not have a single point of failure. So, if a component in the system is such that if that one fails, the entire system will fall.

Redundancy:

Redundancy is the act of duplicating or multiplying a component of the system.

Now let’s say a system has one server to handle the request of clients. It’s a single point of failure.

So, we need to add more servers to handle the requests. But we will need a load balancer to balance the loads between the servers.Now, if the load balancer is out, the system will not be available.

So, we need multiple load balancers to remove the single point of failure of the system. Now, if one of the servers is down, other servers can handle the client requests. If a load balancer is down, other load balancers can take requests and send them to the server.

Thera are also several factors that need to be considered:

* **Environment:** if all your servers are located in the same geographical area, an environmental condition such as an earthquake or flooding could take your whole system down. Having redundant servers in different datacenters and geographical areas will increase reliability.
* **Hardware:** highly available servers should be resilient to power outages and hardware failures, including hard disks and network interfaces.
* **Software:** the whole software stack, including the operating system and the application itself, must be prepared for handling unexpected failure that could potentially require a system restart, for instance.
* **Data:** data loss and inconsistency can be caused by several factors, and it’s not restricted to hard disk failures. Highly available systems must account for data safety in the event of a failure.
* **Network:** unplanned network outages represent another possible point of failure for highly available systems. It is important that a redundant network strategy is in place for possible failures.

1. **What are the different requirements of replication in distributed system? - 2069 Sample-Question**

Replication is the mechanism of maintenance of multiple copies of data at multiple computers. It helps to make distributed system effective by enhancing performance. Data replication is done with an aim to:

· Increase the availability of data.

· Speed up the query evaluation.

The different requirements of replication in distributed system are

**Transparency:**

Replication transparency ensures that replication of databases are hidden from the users. It enables users to query upon a table as if only a single copy of the table exists.

Replication transparency is associated with concurrency transparency and failure transparency. Whenever a user updates a data item, the update is reflected in all the copies of the table. However, this operation should not be known to the user. This is concurrency transparency. Also, in case of failure of a site, the user can still proceed with his queries using replicated copies without any knowledge of failure. This is failure transparency.

**Consistency:**

Consistency model is the contract between processes and the data store. Whenever a read operation is done, it should return a value showing the last write operation on data. But due tolack of global clock, it is difficult to determine last write operation.To use DSM, one must also implement a distributed synchronization service. This includesthe use of locks, semaphores, and message passing. Most implementations, data is read fromlocal copies of the data but updates to data must be propagated to other copies of the data.Memory consistency models determine when data updates are propagated and what level ofinconsistency is acceptable.

Simply consistency is A contract between the client developer and a provider of the replicated service .The provider guarantees that the data will be updated according to some consistency criteria The application developer will need to devise applications with these criteria in mind “Ideal consistency”: system behavior is indistinguishable from a non-replicated system.

**Performance enhancement**

The copy of data placed in the proximity of the process using that data decreases the time to access the data. This enhances performance of the distributed system.

Example: Web browsers store a copy of previously fetched web page locally as a cached data to reduce latency of fetching resources from the server.

**Increased availability**

Users always want the services to be highly available. Replication helps in data redundancyi.e. availability of data even if the server fails. If each of n servers has independentprobability p of crashing, then the availability of resources at each server is 1-pn.

Example: 5% chance of a server failure within a given period - two independent servers give99.75% availability

**Fault Tolerance**

Even if one server fails, the data on other servers are provided to the users. To allow file access to occur even if one file server is down. A server crash should not bring the entire system down until the server can be rebooted.

**Dependability requirements**

**Availability**

- at least some server somewhere

- wireless connections => a local cache

**reliability (correctness of data)**

- fault tolerance against data corruption

- fault tolerance against faulty operations

**Better performance**

1. Multiple servers offer the same service – parallel processing of client

requests

**Geographical distribution**

1. Creating copies of data/objects closer to the clients leads to smaller

network delay and possibly reduced network traffic

1. **Explain the different replication models with suitable examples. - 2069 Sample-Question**

**Active Replication**

Active replication, also called the state machine approach , is a non-centralised replication technique. Its key concept is that all replicas receive and process the same sequence of client requests. Consistency is guaranteed by assuming that, when provided with the same input in the same order, replicas will produce the same output. This assumption implies that servers process requests in a deterministic way.

The main advantage of active replication is its simplicity (e.g., same code everywhere) and failure transparency.

The following steps are involved in the processing of an update request in the Active Replication, according to our functional model.

1. The client sends the request to the servers using an Atomic Broadcast.

2. Server coordination is given by the total order property of the Atomic Broadcast.

3. All replicas execute the request in the order they are delivered.

4. No coordination is necessary, as all replica process the same request in the same order. Because replica are deterministic, they all produce the same results.

5. All replica send back their result to the client, and the client typically only waits for the first answer (the others are ignored)

**Passive Replication**

The basic principle of passive replication, also called Primary Backup replication, is that clients send their requests to a primary, which executes the requests and sends update messages to the backups (see Figure 3). The backups do not execute the invocation, but apply the changes produced by the invocation execution at the primary (i.e., updates). By doing this, no determinism constraint is necessary on the execution of invocations, the main disadvantage of active replication.

Passive replication can tolerate non-deterministic servers (e.g., multi-threaded servers) and uses little processing power when compared to other replication techniques. However, passive replication suffers from a high reconfiguration cost when the primary fails.

The five steps of our framework are the following:

1. The client sends the request to the primary.

2. There is no initial coordination.

3. The primary executes the request.

4. The primary coordinates with the other replicas by sending the update information to the backups.

5. The primary sends the answer to the client.

**Semi Active Replication**

Semi-active replication is an intermediate solution between active and passive replication. Semi-active replication does not require that replicas process service invocation in a deterministic manner. The protocol was originally proposed in a synchronous model [PCD91]. We present it here in a more general system model.

The main difference between semi-active replication and active replication is that each time replicas have to make a non-deterministic decision, a process, called the leader, makes the choice and sends it to the followers.

The following steps characterise semi-active replication, according to our framework.

1. The client sends the request to the servers using an Atomic Broadcast.

2. The servers coordinate using the order given by this Atomic Broadcast.

3. All replicas execute the request in the order they are delivered.

4. In case of a non deterministic choice, the leader informs the followers using the View Synchronous Broadcast.

5. The servers sends back the response to the client.