Automatic Reconfiguration for Large-Scale Reliable Storage Systems

Abstract

Byzantine-fault-tolerant replication enhances the availability and reliability of Internet services that store critical state and preserve it despite attacks or software errors. However, existing Byzantine-fault-tolerant storage systems either assume a static set of replicas, or have limitations in how they handle reconfigurations (e.g., in terms of the scalability of the solutions or the consistency levels they provide). This can be problematic in long-lived, large-scale systems where system membership is likely to change during the system lifetime. In this paper, we present a complete solution for dynamically changing system membership in a large-scale Byzantine-fault-tolerant system. We present a service that tracks system membership and periodically notifies other system nodes of membership changes. The membership service runs mostly automatically, to avoid human configuration errors; is itself Byzantine fault-tolerant and reconfigurable; and provides applications with a sequence of consistent views of the system membership. We demonstrate the utility of this membership service by using it in a novel distributed hash table called dBQS that provides atomic semantics even across changes in replica sets. dBQS is interesting in its own right because its storage algorithms extend existing Byzantine quorum protocols to handle changes in the replica set, and because it differs from previous DHTs by providing Byzantine fault tolerance and offering strong semantics. We implemented the membership service and dBQS. Our results show that the approach works well, in practice: the membership service is able to manage a large system and the cost to change the system membership is low.
Existing System

In Existing System, replication enhanced the reliability of internet services to store the data’s. The preserved data to be secured from software errors. But, existing Byzantine-fault tolerant systems is a static set of replicas. It has no limitations, So, scalability is inconsistency. So, these data’s are not came for long-lived systems.

The existence of the following cryptographic techniques that an adversary cannot subvert: a collision resistant hash function, a public key cryptography scheme, and forward-secure signing key and the existence of a proactive threshold signature protocol.

Proposed System

In Proposed System, has two parts. The first is a membership service (MS) that tracks and responds to membership changes. The MS works mostly automatically, and requires only minimal human intervention; this way we can reduce manual configuration errors, which are a major cause of disruption in computer systems periodically, the MS publishes a new system membership; in this way it provides a globally consistent view of the set of available servers. The choice of strong consistency makes it easier to implement applications, since it allows clients and servers to make consistent local decisions about which servers are currently responsible for which parts of the service.

The second part of our solution addresses the problem of how to reconfigure applications automatically as system membership changes. We present a storage system, dBQS that provides Byzantine-fault-tolerant replicated storage with strong consistency.
Modules

1. Reliable Automatic Reconfiguration
2. Tracking membership Service
3. Byzantine Fault Tolerance
4. Dynamic Replication

Reliable Automatic Reconfiguration

In this Module, it provides the abstraction of a globally consistent view of the system membership. This abstraction simplifies the design of applications that use it, since it allows different nodes to agree on which servers are responsible for which subset of the service. It is designed to work at large scale, e.g., tens or hundreds of thousands of servers. Support for large scale is essential since systems today are already large and we can expect them to scale further.

It is secure against Byzantine (arbitrary) faults. Handling Byzantine faults is important because it captures the kinds of complex failure modes that have been reported for our target deployments.

Tracking membership Service

In this Module, is only part of what is needed for automatic reconfiguration. We assume nodes are connected by an unreliable asynchronous network like the Internet, where messages may be lost, corrupted, delayed, duplicated, or delivered out of order. While we make no synchrony assumptions for the system to meet its safety guarantees, it is necessary to make partial synchrony assumptions for liveness.

The MS describes membership changes by producing a configuration, which identifies the set of servers currently in the system, and sending it to all servers. To allow the configuration to be exchanged among nodes without possibility of forgery, the MS authenticates it using a signature that can be verified with a well-known public key.
Byzantine Fault Tolerance

In this Module, to provide Byzantine fault tolerance for the MS, we implement it with group replicas executing the PBFT state machine replication protocol.

These MS replicas can run on server nodes, but the size of the MS group is small and independent of the system size. So, to implement from tracking service,

1. **Add** – It takes a certificate signed by the trusted authority describing the node adds the node to the set of system members.
2. **Remove** – It also takes a certificate signed by the trusted authority that identifies the node to be removed. And removes this node from the current set of members.
3. **Freshness** – It receives a freshness challenge, the reply contains the nonce and current epoch number signed by the MS.
4. **PROBE** – The MS sends probes to servers periodically. It serves respond with a simple ack, or, when a nonce is sent, by repeating the nonce and signing the response.
5. **New EPOCH** – It informs nodes of a new epoch. Here certificate vouching for the configuration and changes represents the delta in the membership.

Dynamic Replication

In this Module, to prevent attacker from predicting

1. Choose the random number.
2. Sign the configuration using the old shares
3. Carry out a resharing of the MS keys with the new MS members.
4. Discard the old shares
System Configuration

Hardware Requirements

- System : Pentium IV 2.4 GHz.
- Hard Disk : 40 GB.
- Floppy Drive : 1.44 Mb.
- Monitor : 15 VGA Color.
- Mouse : Logitech.
- Ram : 512 Mb

Software Requirements

- Operating system : Windows XP.
- Coding Language : C#.Net
- Database : Sql Server 2005