CSE 601: Distributed Systems

Toukir Ahammed

Design Goals of a Distributed Systems

Security

Security

- When a system is distributed, the number of ways that the system may be attacked is significantly increased, compared to centralized systems.
  - attackers may target any of the individual system components or the network itself
- If a part of the system is successfully attacked then the attacker may be able to use this as a 'back door' into other parts of the system.
- Difficulties in a distributed system arise because different organizations may own parts of the system. These organizations may have mutually incompatible security policies and security mechanisms.

Types of Security Attack

The types of attack that a distributed system must defend itself against are:

- Interception, where communications between parts of the system are intercepted by an attacker so that there is a loss of confidentiality.
- Interruption, where system services are attacked and cannot be delivered as expected.
  - Denial of service attacks involve bombarding a node with illegitimate service requests so that it cannot deal with valid requests.
- Modification, where data or services in the system are changed by an attacker.
- Fabrication, where an attacker generates information that should not exist and then uses this to gain some privileges.

Design Goals of a Distributed Systems

Quality of Service (QoS)

Quality of Service (QoS)

- The quality of service (QoS) offered by a distributed system reflects the system's ability to deliver its services dependably and with a response time and throughput that is acceptable to its users.
- Quality of service is particularly critical when the system is dealing with time-critical data such as sound or video streams.
  - In these circumstances, if the quality of service falls below a threshold value then the sound or video may become so degraded that it is impossible to understand.

Quality of Service (QoS)

This is not always practicable for two reasons:

- It may not be cost-effective to design and configure the system to deliver high quality of service under peak load.
    - The peak demands may mean that you need many extra servers than normal. This problem has been lessened by the advent of cloud computing where cloud servers may be rented from a cloud provider for as long as they are required.
- The quality-of-service parameters may be mutually contradictory.
    - For example, increased reliability may mean reduced throughput, as checking procedures are introduced to ensure that all system inputs are valid.

Design Goals of a Distributed Systems

Failure Management

Failure Management

- In a distributed system, it is inevitable that failures will occur, so the system has to be designed to be resilient to these failures.
  - "You know that you have a distributed system when the crash of a system that you've never heard of stops you getting any work done."
- Distributed systems should include mechanisms for discovering if a component of the system has failed, should continue to deliver as many services as possible in spite of that failure and, as far as possible, automatically recover from the failure.

Degree of distribution transparency

Degree of distribution transparency

- Distribution transparency is generally considered preferable for any distributed system
- But there are situations in which blindly attempting to hide all distribution aspects from users is not a good idea.
- There is also a trade-off between a high degree of transparency and the performance of a system.

Degree of distribution transparency

For example,

- Many Internet applications repeatedly try to contact a server before finally giving up
- Attempting to mask a server failure before trying another one may slow down the system
- In such a case, it may have been better to give up earlier, or at least let the user cancel the attempts to make contact.

Degree of distribution transparency

Another example,

- Several replicas, located on different continents, must be consistent all the time
- If one copy is changed, that change should be propagated to all copies before allowing any other operation.
- A single update operation may now even take seconds to complete, something that cannot be hidden from users.

Degree of distribution transparency

- Full distribution transparency is simply impossible
- We should ask ourselves whether it is even wise to pretend that we can achieve it.
- It may be much better to make distribution explicit so that the user and application developer are never tricked into believing that there is such a thing as transparency.
- The result will be that users will much better understand the (sometimes unexpected) behavior of a distributed system, and are thus much better prepared to deal with this behavior.

Degree of distribution transparency

- Aiming for distribution transparency may be a nice goal when designing and implementing distributed systems
- Full distribution transparency is simply impossible and the price for achieving full transparency may be surprisingly high
- It should be considered together with other issues such as performance and comprehensibility.
- Sometimes it may be much better to make distribution explicit so that the user and application developer will much better understand the behavior of a distributed system, and are thus much better prepared to deal with this behavior.