Summary

Siao-Ting Wang

Problems

- Original designs of both papers use a centralized way to manage resources.
  - Development of the Domain Name System
    - A host machine provides the host.txt for lookuping the mappings between IP and DN.
  - The Maintenance of Duplicate Database
    - Use a single machine to store one database.
- Why this is a problem?
  - Centralized resources are unreliable and can not scale well.
  - With only one host machine or database, users cannot access resource while it crashes.
  - Development of the Domain Name System
    - When the network becomes larger, the size, changing rate and access counts of host.txt increase dramastically. Therefore, it's costly to distribute when more and more users access the same file.
  - The Maintenance of Duplicate Database
    - The access is slow and inefficient if users are not close to the single database.

Solutions (1/2)

- Both papers break the centralized control through distributing and duplicating resources over the network.
  - Development of the Domain Name System
    - Design a hierarchical domain naming system while the mappings are maintained in a distributed manner. Data distribution is provided by zones and caching.
    - Zones are duplicated to multiple servers to increase reliability.
    - With caching, the mappings are put close to resolvers.
  - The Maintenance of Duplicate Database
    - Maintain duplicate copies of database in a distributed network environment.
- Why build a distributed system? (from the reference website)
  - To connect remote users with remote resources in an open and scalable way.
  - With distributed systems, we can share resources, speed-up computation and increase reliability.

Solutions (2/2)

- But, what is the definition of distributed systems? (from the reference website)
  - An application that executes a collection of protocols to coordinate the actions of multiple processes on a network, such that all components cooperate together to perform a single or small set of related tasks.
  - The characteristics a distributed system must have if it is reliable
    - Fault-tolerant/highly available/recoverable/consistent/scalable/predictable performance/secure.

Challenges (1/2)

- However, it is difficult to design a reliable distributed system over an unreliable communication.
- Both papers encounter the communication delay and server failures
- Development of the Domain Name System
  - Unexpected low performance due to network glitches, such as low-speed links and network congestions. Caching can improve the performance, but a stale cache creates inconsistency and makes the system less reliable.
  - Applications may encounter temporary DNS failures. Applications or OS should handle this problem.
- The Maintenance of Duplicate Database
  - The new design doesn't guarantee that the databases are identical all the time because of communication delays and failures but guarantees that the copies are eventually consistent.

Challenges (2/2)

- Challenges while building a distributed system (from the reference website)
  - It’s not easy to achieve reliability over an unreliable communication network.
    - Caches and replication strategies are methods for dealing with states across components. We try to minimize stateful components in distributed systems, but it's challenging. There are a set of tradeoffs in deciding how and where to maintain state, and when to use caches and replication.
  - Design for failures since failure happens all the time.

More Design Guidelines from The Reference

- How to design a reliable system over an unreliable communication network?
  - Start by limiting the scope and making assumptions. Focus on a particular type of distributed system design, such as some kind of standard protocols. Those protocols can provide considerable help with the low-level details of reliable network communications, which makes our job easier.
    - For example, in “The Maintenance and Duplicate Database”, the authors assumes that the underly communication protocol is something that TCP, the messages from one database process to another are delivered in the same order that they are sent.

More Design Guidelines from The Reference

- How to design a reliable system over an unreliable communication network?
  - Well-known 8 false assumptions when developing a distributed system for the first time
    - The network is reliable.
    - Latency is zero.
    - Bandwidth is infinite.
    - The network is secure.
    - Topology doesn't change.
    - There is one administrator.
    - Transport cost is zero.
    - The network is homogeneous.

More Design Guidelines from The Reference

- Distributed System Design Principle
  - Design with the expectation of failure. Define failure scenarios and identify how likely it may occur.
  - Both clients and servers must be able to deal with unresponsive senders/receivers.
  - Minimize latency and the data you send over the network.
  - Don't assume that data sent across a network (or even sent from disk to disk in a rack) is the same data when it arrives.
  - There are a set of tradeoffs in deciding how and where to maintain state, and when to use caches and replication. Try to minimize stateful components in distributed system
  - Be sensitive to speed and performance.
  - Retransmission is costly.

Conclusion

- These papers and the website first show why a centralized implementation is not always practical.
- They also demonstrate possible challenges when converting a centralized design to a distributed one.
- Finally, some assumptions and solutions are proposed to solve the challenges.

Thank You