CMPSCI 630 Systems **Spring 2018**

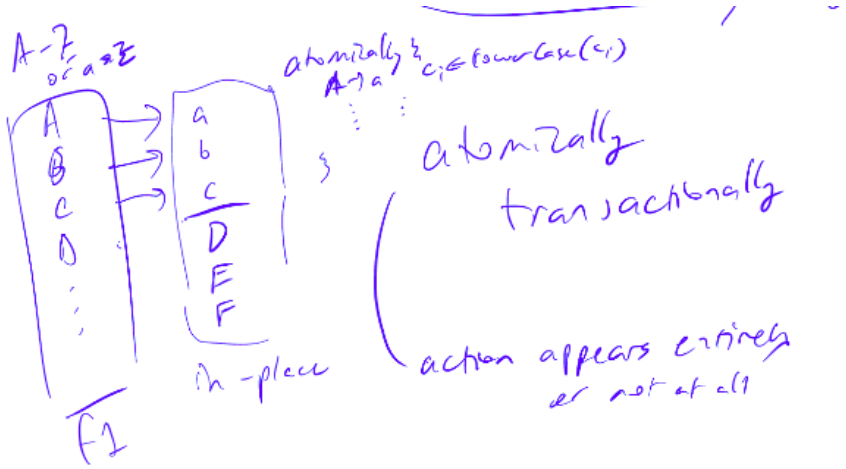
Lecture 18

Lecturer: Emery Berger *Scribe: Prakhar Sharma*

18.1 Fault tolerance

Transaction processing is divided into individual, indivisible operations called transactions. Each transaction must succeed or fail as a complete unit; it can never be only partially complete. Transactions ensure traversal from consistent state to consistent state.

In place file editing is bad. e.g. suppose we have a file with upper case alphabets and we want to convert it to lower case. While editing F1, an error during the write could corrupt the file. The solution is to write to a different file (F2) and then swap their roles.



18.2 Fault tolerance in systems

RAMs can have errors due to cosmic rays - they can flip a bit. Shielding is done as a preliminary measure but RAM error correcting codes are used to prevent errors.

Hard drives are prone to faults because:
1) Manufacturing defects
2)Dust particles can get between the disks
3)Seals in HDD can fail
4)when dropped they get corrupt

The UDP protocol is error prone and a buffer overflow can cause packet loss. UDP works best in absence of faults.

TCP/IP checks for faults. The error correction is built in the protocol via sending of an ack and checksum as a response to the packet. If checksum fails, the sender sends original message again. For big files, big checksums can be used (128k, MD5 sum). This also helps security.

18.3 Byzantine fault tolerance

Preservation of reliability and security is the key while designing systems. Byzantine fault tolerance (Lamport et.al) solves the Byzantine generals problem and asserts that (modulo crypto), a system with 3n+1 nodes can maintain reliability and security (and reach consensus) with at most n malicious generals. Using crypto, the result is that a system with 2n+1 nodes can reach consensus with n malicious nodes.

FLP (fischer lynch patterson) result: Distributed asynchronous consensus is impossible to achieve. This is because a delay and a failure is ough to distinguish. As a solution timeouts can be used but they aren't effective.

18.4 Paxos and PAXOS made simple

Paxos is a family of protocols for solving consensus in a network of unreliable processors. Consensus is the process of agreeing on one result among a group of participants. This problem becomes difficult when the participants or their communication medium may experience failures. It is also widely considered a tough algorithm to understand, hence RAFT was developed. RAFT can be broken down into three main operations:

1)Leader election
2)log replication
3)Log safety

RAFT is better than 2-Phase and 3-Phase commit which fail when the leader dies. Aliveness of the leader can be ascertained using heartbeat messages.