Time and Coordination

March 13, 2001

What is time? :-)

- Issue: How do you coordinate distributed computers if there is no global time?
- Problem: There is no Global Time that everyone can synchronize on
  - Issues: Clock drift, clock skew
- Approach - Bound the amount of skew
- Why do we want to coordinate?
  - e.g., certificates, time stamps on files, ...

Synchronization

- Two systems agree to what time it is
  - External - Use an external clock
  - Internal - Use local clocks (subset of external)
- One approach to synchronizing two systems:
  - Send a message with the time, t, receiver sets time to t+transmission time
  - Problem: network is asynchronous, not isynchronous...
  - e.g., Bank Vault Problem

NTP - Network Time Protocol

- Designed to:
  - Synchronize computers on a large network to UTC
  - Reliability - Can survive lengthy losses of connectivity
  - Provide significantly less drift than motherboard clocks
  - Security against denial of service and spoofing

NTP Implementation

- UDP-based
- Hierarchy of primary and secondary servers
  - Root nodes are Primary servers
    - Connected directly to master clocks
  - Next level are secondary servers
    - Synchronized to primary servers
  - Hierarchy continues, e.g., with your workstation at a leaf

NTP Implementation, cont.

- As servers become disconnected from higher tiers, they can synchronize with their peers (or lower)
- Synchronization occurs via three modes:
  - Multicast (on high-speed LANs)
  - Procedure-call, based on Christian's algorithm (probabilistic)
  - Symmetric - Exchange time data regularly, use data to more accurately predict round trip times

Distributed Mutual Exclusion

- Need to restrict access to distributed resource
  - e.g., file on a distributed file system
- Need safety and fairness (liveness, maybe ordering)
- Key objectives - Low bandwidth, low latency, low impact on peak rate at which resource can be accessed
- Simple: Central server algorithm
  - Performance: entering critical section - 2 messages + 1 RT time delay, release is 1 message + no delay

Distributed Mutual Exclusion, cont.

- Ricart and Agrawala - DME based on multicast and clocks
- A peer process multicasts a request to enter a critical section
  - Messages is <Ti, pi> (time stamp + unique process id)
  - Messages are totally ordered
- The process can enter once it hears back from all its peers

Consensus

- 1 process suggests a value, N processes agree to it
  - e.g., all systems must agree to launch or to abort
- Classes of consensus problems:
  - Byzantine generals (attack or retreat)
  - Interactive consistency (agree on a vector of values)

Byzantine Generals

- 3 or more generals must agree to attack
  - Only one issues the order
  - The others must decide to do “the right thing”
  - There can be treachery!
  - No authentication

Byzantine Generals, cont.

- In a synchronous system, impossible with N<= 3f
  - Example solution - for N>= 4, f=1, majority function suffices
- In an asynchronous system, impossible
  - Cannot distinguish between treachery and slowness