CMPSCI 677 Operating Systems **Spring 2015**

Lecture 12: March 2

Lecturer: Prashant Shenoy *Scribe: Seema Guggari*

12.1 Naming System

As the system gets larger, the naming system itself gets distributed. The design of the naming system includes organization of objects and resources in heirarchial form. This is mainly used to resolve to objects that your application is trying to access. Examples of naming system includes

- Distributed Naming
- DNS
- LDAP

12.1.1 X.500 Directory Service

- X.500 is a general naming service also referred to as the Directory Service. It generalizes the type of functions that DNS provides
- Directory Service requires us to do other kinds of services. For example: Yellow Pages - look for a plumber. The query might be show me listings of name of plumber. here we do not know the name of the plumber. We are looking for the name of the person with the attribute plumber
- Thus, X.500 provides special kind of naming service, it provides general lookup as well as lookups in a variety of domains.

12.1.2 Lightweight Directory Access Protocol

- X.500 is too complex for many applications. Hence we use a simplified variant of X.500 which is LDAP. For example, People finder on umass web page is running on LDAP service. LDAP can also be used for authentication of a user and to query the resources present in a system etc.
- LDAP is a service which is built on top of a database but specifies a schema.
- Practically, every commercial Operating System supports some kind of directory service which is a form of LDAP.
- Examples of LDAP implementations include
  - Active Directory for Windows OS
  - Novell Directory Services
  - iPlanet Directory Services
  - OpenLDAP for Unix as well as MAC OS
  - Typical uses of these services include storing user profiles, access privileges, resources present in a system etc

12.1.3 The LDAP Name Space

- The schema which LDAP uses may be used to store people's records or machine's records
- Attributes for the schema include Country, Locality, Organization etc. LDAP also keeps track of local machines like mail server, print server etc
- LDAP keeps track of the namespace in a heirarchial form

DNS is used for a specific mechanism on the web to do names to IP address translation. LDAP is useful for other applications within an organization such as to keep track of user profiles, their privileges, their credentials etc

12.2 Canonical Problems in Distributed Systems

These are the common set of problems that arise when building a distributed system:

- Time Ordering and Clock Synchronization
- Leader Election
- Mutual Exclusion
- Distributed Transactions
- Deadlock Detection

12.2.1 Clock Synchronization

- Assume that a machine has a system clock that tells you the current time. Every application can request the current time from the system clock. Time is unambiguous in centralized systems, so there is no synchronization problem in centralized systems.
- However in a distributed system each machine has its own clock. Note that clocks in machines are usually crystal oscillators. They are not very accurate. Thus, the asynchronization among the clocks will cause problems.
- Let's use "make" as an example. Make is an utility that allows you to incrementally compile every source file that was modified since the previous time it was run. The way it works is to compare the time stamps on the source files and the object files. If these files are located in different machines, the clocks on these machines may not be synchronized. Then if you save a source file on a machine with slower clock, its time stamp may be earlier than the object file, and make won't recompile that source file. Many problems like this occur because of lack of synchronization. There is no global clock that all time stamps can derive from. Today we will discuss algorithms that synchronize clocks. Note that all these algorithms have errors, but as long as the errors are small or negligible to the application, we can still use these algorithm.

12.2.2 Physical Clocks: A Primer

There are different kinds of clocks.

- The most accurate clocks are atomic oscillators. They have the accuracy of one part in 10^13. Most of the time the way you keep real time is to synchronize with a atomic clock. Most countries have a atomic clock, and broadcasts its time with radios or satellites.
- Most of the clocks are less accurate, example mechanical watches
  - Typically in machines, the clocks are crystal oscillators. There are crystals in these clocks.The clocks count the number of oscillations and ticks the clock after a certain number.
  - Thus, clocks might drift because crystal oscillations are not exact.
- What time really means depends on how astronomers tell time. They can use the rotation of earth, the moon, or the sun to tell time.
- There are multiple time zones on earth, and the zones coordinate with universal time(UTC), which is a international standard for time. So the best way to synchronize time is to synchronize your clock with an atomic clock, or some authoritative clock.

12.2.3 Abstract Properties of Clock

These observations are based on the figure on page 5 of lecture 12.

- The middle line with slope 1 is the perfectly synchronized clock.
- Slow clock has a slope smaller than 1, and fast clock has a slope greater than one.
- The difference of the slope between a clock and a perfect clock is called a drift, which indicates that how long does it take for a clock to drift by a second.
- If a clock has a drift rate ρ , $1 - \rho <= dC/dt <= 1 + \rho$. For two clocks both with drift rate ρ , they may drift by $2\rho\triangle t$ in time $\triangle t$.
- In order to limit the drift to δ , you have to resynchronize every $\delta/2\rho$ seconds. Every clock synchronization protocol makes this assumption.

12.2.4 Cristian's Algorithm

- Christian's algorithm is used to synchronize machines to a time server with a UTC receiver.
- However, we have to take $t_{req} and t_{reply}$ inrto account. One way to estimate t_{req} is to take the round-trip-time and divide it by two. Then add t_{req} to the clock sent back in the reply.
- In reality, network delay are not symmetric. Also, the amount of time a server processes the message is ignored. Cristian's algorithm improves the accuracy by sending a series of request and calculating the mean of the replies. This is absolute time synchronization. The machine is synchronized to real time.

12.2.5 Berkeley Algorithm

- Berkeley's algorithm is useful in systems which has no time server to synchronize to. You can let two machine sync with each other, neglecting their drift from the real time.This algorithm is relative synchronization.
- The first step of Berkeley Algorithm is to elect a master, which is coordinator of the system. It polls to all machines for their time, sets the current time to the average of all machines, and sends it to every other machine. An example is shown on page 8 of lecture 12.
- You don't know the clock difference with respect to the real time, because you don't know which clock is more accurate. The reason that we don't simply send the master's time to every machine is because we still want the time to be more accurate with respect to real time
- In this algorithm, you still have to account for message delays. If the master goes down, you simply elect another master.
- In this scenario, some clocks will be moved back in time. In reality, there are harmful effects to do that. Because if you time stamp a file and them move back the clock, then that file has a time stamp in the future with respect to the new time. This certainly causes confusion. Sophisticated version of clock synchronization algorithms don't move the clock back, instead slows down the rate the clock ticks until real time catches up with this clock.

12.2.6 Distributed Approaches

- Both approaches mentioned above are centralized. The server machine could fail, or it could be the bottle neck. There are distributed algorithms of the Berkeley algorithm as well. Each machine broadcasts its time to every machine, and the average is calculated locally. To get a better average, this algorithm can throw away the highest and lowest values.
- One approach used today is the *rdate* utility on unix. You pass the name of the machine into *rdate*, and it essentially runs the Cristian's algorithm to that machine. The machine periodically runs this utility to sync with a certain machine.
- More recent approach is to use NTP, or network time protocol. It basically uses the same idea, but it has a hierarchy of time servers. The top of the time server synchronizes with an atomic clock, and other machines synchronize with their parent in the hierarchy. The lower the level a machine is in, the higher the drift is .NTP has a accuracy of 1-50ms. For most clocks and applications this is enough. If an application has higher accuracy requirements than 1-50ms, then NTP is not a suitable algorithm. Most machines are preconfigured to use NTP. NTP prevents clocks from going backward. If a clock is faster than real time,it will make the clock to click slower.

12.2.7 Global Positioning System

- GPS system was put in place to find out the coordinates of an arbitrary object
- GPS is a positioning technology that finds the current coordinates of a node. To find the position, you need to know some authoritative positions and do triangluation.
- There are unknown beacons and we have to figure out the co-ordinates of the object wrt these beacons
- The beacons in case of GPS are the satellites. These are the satellites in the sky. They typically broadcast their locations

- A typical receiver has to listen to broadcast and figure out its co-ordinates wrt the satellite.
- For example, if (x_1, y_1, z_1) and (x_2, y_2, z_2) are given satellites with known locations, and we want to calculate the coordinates of the arbitrary object (x, y, z). You somehow calculated the distance r_1, r_2. Then the object has to lie on one of the intersections of the circles with r_1, r_2 as their radii and (x_1, y_1, z_1), (x_2, y_2, z_2) as their centers. Then you use some heuristic to decide the correct location of (x, y, z). For GPS, you look for the coordinates of the intersections that are located on the ground. To calculate r_1, r_2, the beacon S_1 broadcasts a beacon packet including its coordinates. Assume that the object receives the packet after time t. Then the distance between S_1 and the object is $c*t$. But t could be inaccurate if the clocks at S_1 and the object is not synchronized. This implies that to implement GPS, you have to have a clock synchronization built in.
- Now we can figure out how GPS works. The first satellite sends the packet at t ,and the ground device receives at t_1−t−δ. Note that δ is the drift of the clock at that time. The distance d between a satellite and the node on the ground could be calculated by two ways: the Euclidean distance between the two points, and the speed of light multiplied by the time to travel. Equating them gives us an equation. Now there are four unknown variables: x, y, z, and δ,so you will need at least four equations to solve the values of these variables. With extra satellites, you can construct additional equations. Then you can pinpoint the solution that is closest to the ground. Note that when you find the coordinates, you also find the drift and synchronize the clock. Here the drift is the same for all the satellites, since satellites are synchronized to atomic clocks.
- Thus, we are solving clock synchronization problem as a positioning problem.
- GPS synchronization is very accurate. They are submicroseconds accurate.

12.3 Logical Clocks

- In every synchronization mechanism mentioned above, there will be some inevitable drift. You will never get perfect synchronization. If your application depends on comparing time stamps on different machines, the drift might cause you problems.
- If two events have time stamp difference of 10ms, you can not tell which one happened earlier if the machine uses NTP synchronization, whose accuracy is only 1-50ms.
- Logical clocks is a completely different way to figure out the order of two events. Clock synchronization need not be absolute. Processes sometimes need to agree on the order in which events occur rather than the time at which they occurred.