Enforcing Policy and Data Consistency of Cloud Transactions: A Simulation

Tucker Trainor
tmt33@pitt.edu

April 14, 2012

Environment Variables

Network Latency Variables

Simulating interaction among servers in the Cloud requires careful attention to characteristics common to communication networks. We are particularly concerned with the effects of latency on the simulated performance of our algorithms and heuristics. As a typical scenario for Cloud transactions would take place over a wide area network (WAN), we should simulate delays between clients and servers to mimic real-world latency. A simulation done completely on a single machine (i.e., `localhost`), where latency is virtually negligible, inserting artificial delays between messages is necessary. A simulation done over a local area network (LAN) would also require some artificial delay between messages, though probably not as much of a time penalty as there would be for a `localhost`. The following are recognizable instances in our model where latency can be simulated via thread sleeping:

- Delay before Policy Server updates are received by Cloud Servers
- Delay between a request for a Policy Version and receipt of current version
- Delay before Cloud Server receives queries from Robot
- Delay between messages between Robot Threads and Cloud Server Worker Threads
- Delay between messages between two Cloud Server Worker Threads

Client/Server Specific Variables

The client and server models offer scenarios where the frequency of certain operations, limits to performance, and hardware-based delays can be varied.

Client: Robot, Robot Threads

- Frequency of transaction transmissions to Cloud Server
- Maximum number of concurrent threads at a given instance (multi-programming, parallelism)
- Whether or not a series of a specific query types (i.e., consecutive READs or WRITEs) are handled as a group or as discrete operations
- Whether or not significant delays between READs and WRITEs from the Robot side will be implemented to simulate actual usage by a human

Server: Cloud Server, Worker Threads

- Number of Cloud Servers utilized during a simulation
- Length of time for verifying integrity (during commit procedures, may not be implemented in this version of the simulation)
- Length of time for accessing and checking local data (e.g., disk reads and writes)
- Length of time for Policy Version authorization checks (e.g., local checking or calls from Worker Thread to Cloud Server for current Policy version)

Server: Policy Server, Policy Updater, Policy Threads, Policy Request Threads

- Delay between Policy Version updates – varying the frequency of Policy versions directly affects view consistency and global consistency

Errata & Observations

Any artificial pause or delay must be subject to a random amount of variation. Though there may not be much variance in the latency between two real-world nodes under certain conditions, there is always likely to be at least a few milliseconds on either side of its average latency, a variance that must be accounted for in our simulation. Similar variances should be accounted for in the client and server models as well.

It is worth noting that the seeding of pseudorandom number generators itself is a variable. Simulations involving the variation of seed values only may be worth pursuing.