15640 project3 Design Documentation

Qiyang He

1. VM roles:
    a. Generally, there will only be one master server instance managing a central queue, a cache tier and executing some scaling policies. It will also act like a normal front-end server after first application server finishes booting.
    b. There will only be one master application server which will never be shut down. Besides, it acts like a normal middle tier server.
    c. The frontiers will only be scaled out because we noticed that they will never be the bottleneck in our implementation. All they need to do is to push the request onto the central queue on master server.
    d. The application servers will process the client requests and they will send notifications whenever they process a request. The master server will then decide if the system need to scale in or scale out.

2. System architecture
    a. We implemented a central queue on master server. All the front-end servers (including master server) will pass the requests onto the central queue. The application servers will retrieve requests from queue whenever they finished what they are working on.
    b. A cache instance will be running on master server.
    c. There will be a master server, middle tier master server that will never be shut down. All the other middle-tier/ front-end servers will be scaled in/out according to our scaling policy.

3. Scaling policies
    a. Initially, there will only be one master server (act like a front end), another front-end server and a middle tier application server. A cache will be initialized from master server.
    b. Scale out
        i. Scale out when booting the vms
            1. Before middle tier booting finish, all the requests will probably be dropped. The master server will test the interval of incoming requests. When each 5 requests dropped there will be one additional application server initialized.
        ii. Scale out front end

1. We will compare the number of the front-end servers and the master server's request queue length. If the queue length is larger than 3 times of the front-end server numbers, we need to scale out a new front-end server

iii. Scale out middle tier

1. We will compare the central queue length and the application server numbers. If the central queue length is larger than application server numbers, the application server will notify the master server and also drop this request. If two consecutive drop happens, we assume that the queueing speed of the central queue is larger than the application servers' capacity. We will scale out a new application server

c. Scale in policy

i. Scale in front end

1. As mentioned in first section, the front-end server seldom become bottleneck, so we will not aggressively scale out front-end, nor will we aggressively scale in front-end.

ii. Scale in middle tier

1. The implementation of LinkedBlockingQueue enable us to retrieve requests based on a time-out value. If we wait explicitly longer than this time-out value, the queue will return null. If an application server time out three times consecutively, we will shut down this server. We set time-out value to be 500 ms a time.

4. Database cache

a. The database will be initialized from server and also be managed on server. All the requests will pass through this cache. If it is a get request, we will check the cache and also cache the results if necessary. Other request such as set and transactions will be passed to real database.

b. All the application servers need to look up this cache from registry and get remote instance of this cache class, and pass requests to this cache object.