In all of our experiments, we will be using m1.medium instances on Amazon EC2.

## **Features**:

In this section we plan to evaluate the features of the modified CloudKon(CloudKon-M)

* **Basic Functionality and Decoupled model:**In this test Case we Run Multiple Clients Submitting Heterogeneous Sleep Tasks to the system which has multiple workers. To test the completely decoupled model we will run this experiment with first starting Clients alone and then the workers and vice versa.

**Status: Completed.**

* **Advanced Tasks support:**Original CloudKon supported only sleep tasks. CloudKon-M supports any task which can run independently as a thread. We demonstrate this by running a Client that submits tasks (thread that creates a File of size 1MB ). We expect the Files to be created at the worker nodes.

**Status: Completed.**

* **Task Consistency:**One of the major challenges that CloudKon faced was with respect to the duplicate tasks being executed by workers and its avoidance. CloudKon-M avoids this completely by replacing SQS with Hazelcast. Tests would show that tasks were executed only once in the system.

**Status: Completed.**

* **Advanced monitoring:**CloudKon provided only basic monitoring of the system. CloudKon-M provides a detailed monitoring of all nodes and their respective CPU utilization/minute. We are also using the same module to measure the Q length of individual clients. We expect the monitoring details to be recorded and reported in a user friendly way.

**Status: Completed.**

* **Dynamic Provisioning:**We will demonstrate the systems capability to create new worker nodes as the system detects a rise in Queue Length.

**Status: Work in progress .**

## availability

* **Distributed Queue:**CloudKon had the inbuilt support of SQS which was a highly available queuing service which we have replaced with Hazel cast. We will test the availably of this service under a load of 12000 tasks and 8 nodes. We will kill the nodes forming the hazel cast cluster and watch the consistency of execution.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Node Loss** | | |
| 10% | 30% | 50% |
| **Node Count** | 8 |  |  |  |

* **Distributed Database:** CloudKon had the inbuilt support of DynamoDB which was a highly available service which we have replaced with Cassandra. We will test the availably of this service under a load of 12000 tasks and 8 nodes. We will kill the nodes forming the Cassandra cluster and watch the execution of system for any anomalies.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Node Loss** | | |
| 10% | 30% | 50% |
| **Node Count** | 8 |  |  |  |

## Performance

* **Throughput: No File IO:**We will run 2 Clients and 4 worker (10 threads per worker) per node. Each Client will submit 8000 Sleep 0 tasks. We will run this on different scales and scale it up to 64 nodes. (256 executors)

|  |  |
| --- | --- |
| **Node Count** | **Through Put** |
| 2 | 1380.142247 |
| 8 |  |
| 16 |  |
| 32 | 2909 |
| 48 |  |
| 64 |  |

* **Throughput: File IO:**We will run 2 Clients and 4 worker (10 threads per worker) per node. Each Client will submit 8000 File Write tasks. Tasks will write a file to local disk of a specified size. We will run this on different scales and scale it up to 64 nodes. (256 executors)

|  |  |
| --- | --- |
| Throughput | |
| 1KiloByte | 1MB | 10MB |
| **Node Count** | 2 | 1084.302549 | 730.0010658 | 540.4037623 |
| 8 |  |  |  |
| 16 |  |  |  |
| 32 |  |  |  |
| 48 |  |  |  |
| 64 |  |  |  |

* **Efficiency:**We will run 2 Clients and 4 worker (10 threads per worker) per node. Each Client will submit 8000 Sleep tasks of varying length. We will run this on different scales and scale it up to 64 nodes. (256 executors) .We calculate Speedup (Sp) as Speedup = (Time that would have been taken for sequential evaluation of all tasks submitted) / (Time taken by system which has multiple nodes executing the same set of tasks).

Efficiency is calculated as Ep = Sp/(total number of nodes for that run).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Efficiency | | **Task sleep Time** | | | | |
| Sleep 1 | Sleep 16 | Sleep 32 | Sleep 64 | Sleep1024 |
| **Node Count** | 8 |  |  |  |  |  |
| 16 |  |  |  |  |  |
| 32 | \* | \* | \* | \* |  |
| 48 |  |  |  |  |  |
| 64 |  |  |  |  |  |

* **Latency:**We will run 2 Clients and 4 worker (10 threads per worker) per node. Each Client will submit 8000 Sleep 1024ms tasks. We will run this on different scales and scale it up to 64 nodes. (256 executors)

|  |  |
| --- | --- |
| **Node Count** | **Latency** |
| 8 |  |
| 16 |  |
| 32 |  |
| 48 |  |
| 64 |  |







