**Compiling storm :**

Unzip shared apache-0.9.2-incubating.zip. Run :

cd directory

mvn clean install -DskipTests=true

cd /storm-dist/binary

mvn package -DskipTests=true

**Setting up Storm in the cluster :**

* The required files for running storm are “apache-storm-0.9.2-incubating.zip”, move\_storm\_files.sh, config\_storm.sh, Fault\_injector\_iterate.py, storm.yaml, timeout\_compute.py
* apache-storm-0.9.2-incubating.zip will be be located inside the unzipped folder (shared) under /apache-storm-0.9.2-incubating/storm-dist/binary/target/apache-storm-0.9.2-incubating.zip
* All the other files are zipped and attached (storm-starter-0.9.2-incubating-jar-with-dependencies.jar is also required but it can be obtained by compiling storm starter after unzipping “apache-0.9.2-incubating.zip”)
* Change the paths of some of the files accordingly in move\_storm\_files.sh
* Use the script ./move\_storm\_files.sh after all VMs have been started

**move\_storm\_files.sh :**

Transfers all files to all running cluster VM's.

* apache-storm-0.9.2-incubating.zip is stored in /tmp/ directory in each VM
* config\_storm.sh, storm.yaml, timeout\_compute.py are also transferred to /tmp/ directory in each VM
* storm-starter-0.9.2-incubating-jar-with-dependencies.jar is transferred to /home/vignesh in nimbus. This can be modified accordingly in move\_storm\_files.sh
* Fault\_injector\_iterate.py is also transferred to /home/vignesh in nimbus. The destination can be modified accordingly in move\_storm\_files.sh

**config\_storm.sh :**

Log in to each cluster VM. At each VM do the following to setup storm :

* /tmp/config\_storm.sh - would setup the new version of storm and a link to the executable would be created n /home/vignesh in the VM

**timeout\_compute.py :**

The would remain in /tmp/. It can be relocated to /app/home/storm (Storm home directory) if required. This step is optional. All slaves currently already contain the latest version of timeout\_compute.py

**Fault\_injector\_iterate.py :**

It is a multi purpose script. Word count topology currently takes in 4 arguments. Before starting the python script, the path to “storm” executable and “storm-starter-0.9.2-incubating-jar-with-dependencies.jar” must be set inside the subprocess statement in the script. If all previous steps are followed, both these files will be located in /home/vignesh in nimbus.

Word count topology takes four arguments :

1st argument : Topology name ( which is “WordCount” currently)

2nd argument : Current rate

3rd argument : str(1) or str(0) to enable or disable adaptive timeout respectively

4th argument : sample\_size which is related to the message drop probability as msg drop prob = 1/sample\_size. If sample\_size is ignored or set to 0, then no messages are dropped.

**Setting the adaptive timeout mode :**

The adaptive timeout mode can be set by setting the string Adaptive\_timeout\_mode in Word\_count\_topology.java accordingly :

Current supported modes :

**NORMAL :** Constant timeout value of 30 sec

**END\_TO\_END :** End\_to\_end mode, “End\_to\_End.java” is activated.

**QUEUEING MODEL <Submodelname>** : This represents a class of Queuing model based timeouts. The name can be arbitrary but is must have a prefix “QUEUEING MODEL” to active Queueing\_model.java. For example mode can be “QUEUEING MODEL MM1” or “QUEUEING MODEL G/G/1” and so on. The mode value is passed to the Queueing\_model object and set to the attribute called mode in Queuing\_model.java.

**Current Supported Timeout Modes :**

**NORMAL**

**END\_TO\_END**

**QUEUEING MODEL MM1**

**QUEUEING MODEL Heavy Traffic**

**Output files :**

Currently files would be output only for the NORMAL and END\_TO\_END modes. NORMAL mode would output the total latency and statistics. END\_TO\_END mode would output total latency, statistics, tick tuple files as well. The Word\_Count\_topology.java defines a string called topology\_name which defines the output\_folder name. Currently it is set to “Word\_count\_topology”. So the output files are located inside /app/home/storm/Word\_count\_topology. The folder would contain sub folders defined by the spout rate, sample\_size, tweet keyword and whether the adaptive timeout is enabled or not. These folders can be named as desired by setting the topology-info string in Word\_Count\_topology.java.