Twitter Storm

Saumil Srivastava [sxs129830@utdallas.edu](mailto:sxs129830@utdallas.edu)

Rajas Upadhye [rxu120330@utdallas.edu](mailto:rxu120330@utdallas.edu) Avinash Agrawal [axa129530@utdallas.edu](mailto:axa129530@utdallas.edu)

Paroksh Sharma

[pxs125130@utdallas.edu](mailto:pxs125130@utdallas.edu)

Ekta Dharia

[exd130130@utdallas.edu](mailto:exd130130@utdallas.edu)

***Abstract* – The proposed project is on retrieving the trending topics in Twitter using Storm and Cassandra. In this inputs are the live tweets from Twitter and with help of Storm we find the topmost tweets using hash-tags and rolling count. These topmost tweets are then stored in the Cassandra column family using Cassandra Bolt.**

***Index terms*: Storm, spout, bolt, rolling count, hash-tags**

1. INTRODUCTION

Motivation: In the current era people need to know the top news going around them. The twitter users would like to know the topmost tweets to browse through quickly. These topmost tweets are the trending topics which the world users are mostly interested in. Since the users want to know the current going on tweets the topmost tweets should be in real-time.

This led us in the building of the project, to get the topmost tweets in twitter. Storm is used to collect the topmost tweets and is stored using the Cassandra. Storm is an open source, distributed, scalable, fast, transactional, fault tolerant, reliable computation system. It is used to process real-time streams of data. It is easy to set up and simple to use. It was built in Clojure language but its applications can be written in any languages. Typical use cases of storm are Processing streams, Continuous computation and Distributed remote procedure call. Apache Cassandra is an open source distributed database management system which is used to manage lots of data. It does not have a single point of failure and it also gives high availability. It also provides high performance value.

1. DETAILS OF APPROACH
2. Architecture

The architecture of storm is shown in figure 1. Storm has two cluster nodes:

1. Master node
2. Worker Node

The master node organizes all the nodes to work it continuously. It runs on the daemon: nimbus. It assigns the work in the cluster by giving work to the worker nodes and monitors if any failures. The worker executes the part of work given to it and it works on daemon: Supervisor. Both the daemons are stateless. The zookeeper is where all the clusters are kept by the storm.

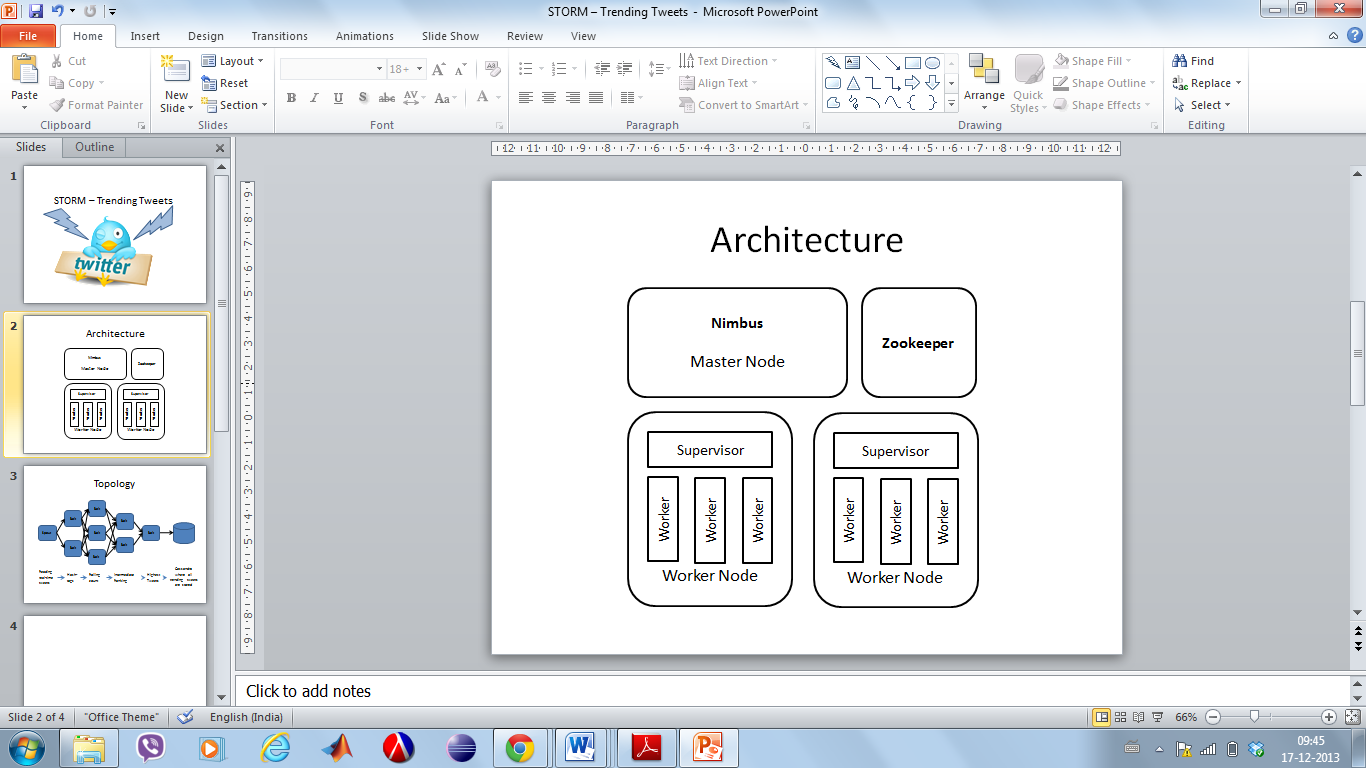


Figure 1: Architecture of Storm

1. Topology

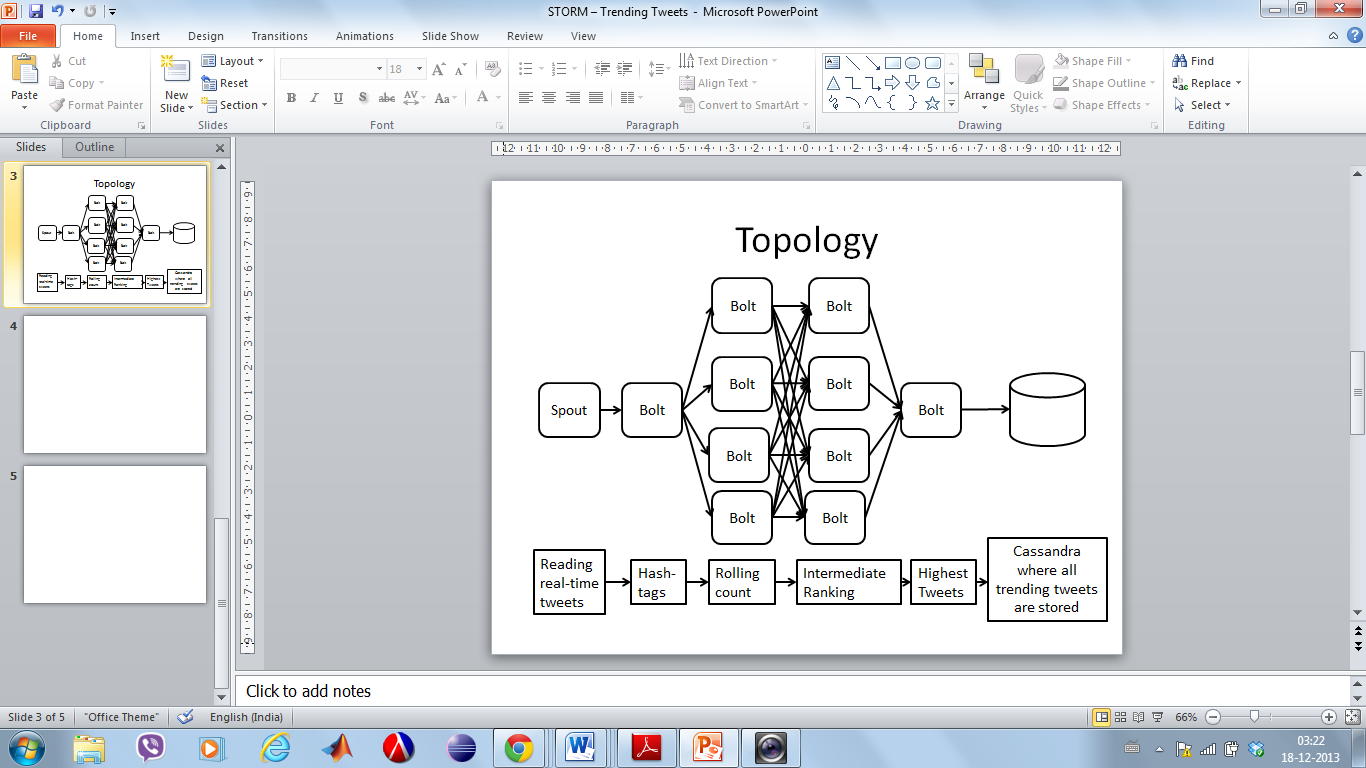


Figure 2: Topology used in Storm

In topology layers: First layer runs a spout to read the real time input of tweets which the users are tweeting. Second layer picks up hash-tags from the tweets. Third layer the rolling count is done of the hash-tags by the four bolts. Fourth layer takes values from third layer and distributes the load to find highest ranks of the trending tweets. Fifth layer actually gets us the highest ranks from merging the inputs from fourth layer and making universal consolidated highest ranking tweets. This is then stored to a database using Cassandra.

1. METHOD
2. Technologies Used:
3. Storm

It is fast, reliable, scalable, distributes, open-source real time processing system. Storm is simple and can be used with any programming language. Spout handles the input stream and passes it to the bolt. The bolt transforms the inputs and passes to other bolts for further processing or sends it to storage.

1. Cassandra

Cassandra is a highly scalable, eventually consistent, distributed, fault tolerant and structured key-value store. It’s a row-oriented and column-structured. Cassandra's support for replicating across multiple datacentres is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive regional outages. Cassandra's data model offers the convenience of column indexes with the performance of log-structured updates, strong support for de-normalization and materialized views, and powerful built-in caching.

1. Hosebird client

Hosebird is Twitter Streaming API's server implementation which is a provision by Twitter to stream the live Tweets in near real time. We used this developer feature with an official twitter account to use the Hosebird client. The Hosebird client is broken down into two modules: hbc-core and hbc-twitter4j. The hbc-core module makes use of a message queue, which the consumer can poll for the raw String messages and hbc-twitter4j module uses the twitter4j listeners and data model on top of the message queue to provide a parsing layer.

1. Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that supplies resizable calculated size in the cloud. It is created to make web-scale processing easier for users. On the Amazon’s processing environment It provides you with complete control of your computing resources and lets you run on Amazon’s proven computing environment.

1. First setting up Technologies
2. Storm

Installation of Storm

Linux VMs with Java version 1.6 and Python version 2.7 were used to install the Storm. The installation of Storm was performed as following;

1. Zookeeper Cluster Installation
2. Storm Client Installation
3. Storm Cluster Installation
4. Install ZeroMQ2.1.7 native dependency
5. Install JZMQ native dependency
6. Start Zookeeper cluster
7. Start Storm Cluster
8. Start master node (Nimbus)
9. Start worker node(Supervisor)
10. Upload topology using Storm Client
11. Amazon EC2 Storm configuration

Amazon EC2 Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. As storm is a distributed real time computation system, leveraging ec2 services to deploy a storm cluster makes this project more robust and scalable. Nathan's storm-deploy project [1] allows deploying a Storm cluster on EC2 based on user's configuration.

Storm-deploy are built on jclouds and pallet. It sets up Zookeeper, sets up Nimbus, launches the Storm UI on port 8080 on Nimbus, launches a DRPC server on port 3772 on Nimbus, sets up the Supervisors, sets configurations appropriately, sets the appropriate permissions for the security groups, and attaches the machine to the cluster.

Ganglia are a scalable, distributed system designed to monitor clusters and grids while minimizing the impact on their performance. It provides the capability of generating reports, viewing the performance of the cluster as a whole, as well as inspects the performance of individual node instances.

The deployment of storm on Amazon EC2 needs the following components:

* + 1. Public and private key paths for setting up ssh on the nodes. It is important that while setting up Linux, you should have a null passphrase on the keys.
    2. AWS user id Identity: Your AWS access key
    3. Credential: Your AWS access key secret
    4. Region where the security groups will be defined.

1. Cassandra

Linux VMs with Java version 1.6 were used to install the Cassandra. The installation of Storm was performed as following;

* 1. Download Cassandra 2.0.3
  2. Configuring the Cassandra
  3. Start Cassandra
  4. Using cqlsh
  5. Test connection to localhost:9160

1. PERFORMANCE EVALUATION

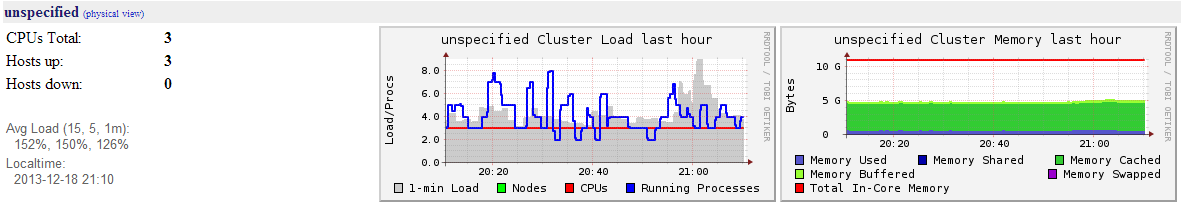
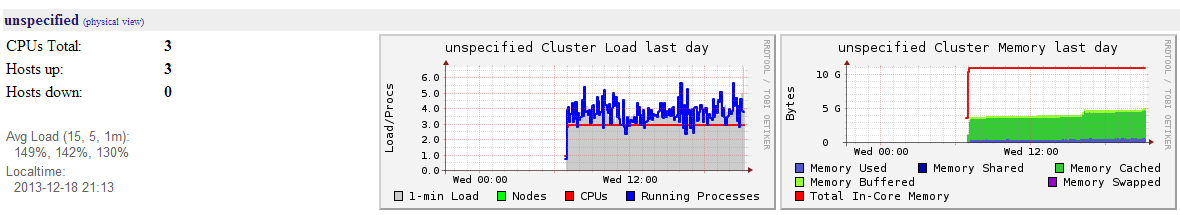
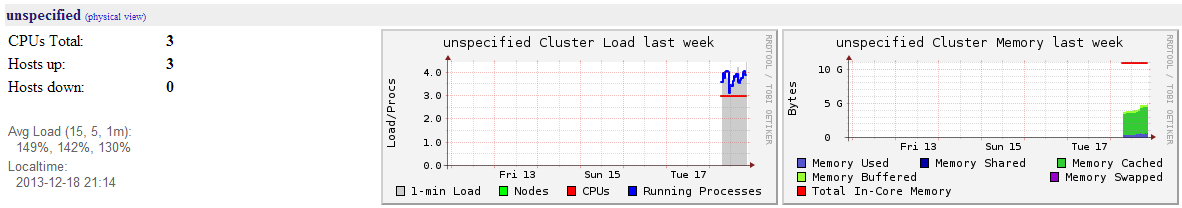
As the project is setup on Amazon EC2 cluster we are utilizing five instances of nimbus, superviser1, superviser2, zookeeper and Cassandra. We can get a detail statistics of performance of instances using Ganglia monitoring system. Ganglia Interface showing the statistics of our setup as shown in figure 3, figure 4 and figure 5.

Figure 3: Hourly statistics

Figure 4: Daily statistics

Figure 5: Weekly statistics

1. RESULTS

Stream of tweets were obtained from near real-time Twitter. The storm took these inputs using the spout and the bolts transformed them to get the top 5 tweets at that instance which with the help of Cassandra Bolt can be added into a Cassandra Column Family.

1. CONTRIBUTION

In this project work we have enhanced the current storm starter project which is made available by Apache. The enhancements were integrating the Hosebird Client with the Storm Starter project to periodically count the Top 5 tweets at a particular timestamp. The Tweets here are near real-time data.

1. CONCLUSION

Thus as a part of implementation of this Twitter Storm project we have extended the rudimentary version of storm starter to a developed version which fetches and streams near real time Twitter data to finally provide with the Top 5 tweets periodically.

REFERENCES

[1]- <https://github.com/nathanmarz/storm-deploy>,

2 -<http://10jumps.com/blog/storm-installation-single-machine> - Single Node Storm Setup

3-<http://wiki.apache.org/cassandra/GettingStarted> - Single node Cassandra cluster

4- <https://dev.twitter.com/blog/the-hosebird-client-streaming-library> - Twitter streaming API for Twitter Developers

5- <http://www.ibm.com/developerworks/opensource/library/os-twitterstorm/>

6 –www.youtube.com for video references