International Journal of Computing and Corporate Research
ISSN (Online) : 2249-054X
Volume 6 Issue 3 May 2016
International Manuscript ID : 2249054XV6I3052016-090

PERFORMANCE EVALUATION OF DYNAMIC ENCRYPTION ON REAL CLOUDS OF IBM BLUEMIX AND REDHAT OPENSHIFT

Gurpreet Singh
M.Tech. Student
Computer Science and Engineering
Desh Bhagat University
Mandi Gobindgarh, Punjab, India

Amandeep Kaur
Assistant Professor
Computer Science and Engineering
Desh Bhagat University
Mandi Gobindgarh, Punjab, India

ABSTRACT

Cloud computing is one of the key domain of research and in use for delivery for assorted computing services including telecommunications, storage, platforms, software, virtual infrastructure and many others. Cloud computing offers the delivery of virtual resources without need to worry on establishing and maintaining the actual infrastructure. The end user of cloud services access the remote computing resources using web based client without having very complex configuration in their own device. A person having 1 GB RAM in their own device can work on the 128 GB RAM installed at cloud server without any issue. That's where the performance of cloud comes to the imagination. In this case, the system or device of user acts as the terminal to access and work on the remote cloud infrastructure. There are number of cloud service providers which can be used for assorted computing infrastructure and applications. This

International Journal of Computing and Corporate Research
ISSN (Online) : 2249-054X
Volume 6 Issue 3 May 2016
International Manuscript ID : 2249054XV6I3052016-090

research work focus on the effective evaluation of assorted cloud service providers and cloudsim based on various factors. In this work, the effective implementation of assorted algorithms and sentiment opinion mining is done on the real as well as simulator based clouds. Sentiment Data Analysis is one of the prominent and widely used domains by the research scholars and practitioners. In this approach, there are number of tools and technologies available for fetching the live datasets, tweets, emotional attributes. Using these tools, the real time tweets and messages can be extracted from Twitter, Facebook, WhatAapp and many other social medial portals.

Keywords – Cloud Computing, Sentiment Mining, Real Cloud

INTRODUCTION

Cloud computing is a unique and effective way to use the computing infrastructure for effective communication and less overhead. "The major technology behind grid as well as cloud computing is the virtualization technology. Using virtualization technology, the number or arbitrary and remote processors works in parallel for the execution of tasks in such a way that there is less complexity and higher integrity of the processes.

In cloud and grid architecture, there are number of remote computing devices and processors which work and execute the processes in parallel so that there is minimum load on the single node or machine. By this way, the overall load is balanced.

There are number of research areas in the domain of grid computing and cloud computing, still few areas are still in the preferences of the research community as these areas needs updations and enhancements in regular basis.

The major areas of research in grid and cloud architecture are -

- Load Balancing
- Power Optimization
- Energy Optimization
- Secured Routing
- Task Scheduling
- Confidentiality and Integrity Management
- Public and Private Key Cryptography ….. and many others

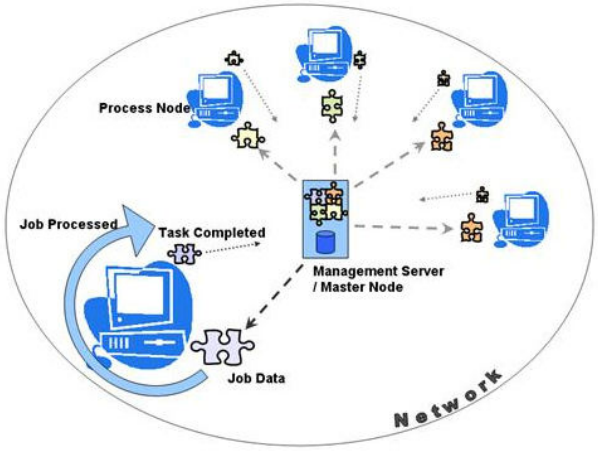


Figure 1.1 - Grid Computing Architecture and Working Model [1]

International Journal of Computing and Corporate Research
ISSN (Online) : 2249-054X
Volume 6 Issue 3 May 2016
International Manuscript ID : 2249054XV6I3052016-090

VIRTUALIZATION TECHNOLOGY

Virtualization technology is back bone in the grid and cloud platforms for effective management of the systems on demand for the jobs under execution. Using virtualization, there is no need of deployment of the dedicated systems or no0064es, rather the virtual machines take charge of all the operating system, platform and the resources to be used.

Virtual machines are classified into the two major taxonomies which depend on their usage and the degree of communication -

- **System Virtual Machine** – This system provides the full system environment to support the processes and execution of operating system in use.
- **Process Virtual Machine.** It is also called as Language Virtual Machine**.** Such machine is deployed and designed to process a single task or job or simply program. It means that it is dedicated for a single process.

Key Virtualization Technology Based Software Suite in the Corporate World

- **Windows as Host OS**
  - VMWare Workstation (Any Guest OS)
  - Virtual Box (Any Guest OS)
  - Hyper-V (Any Guest OS)
- **Linux as Host OS**
  - VMWare Workstation
  - Microsoft Virtual PC
  - VMLite Workstation
  - VirtualBox
  - Xen

Hypervisor Type 1 Software Suite

- VMWare ESXi
- Citrix Xen
- KVM (Kernal Virtual Machine)
- Hyper-V

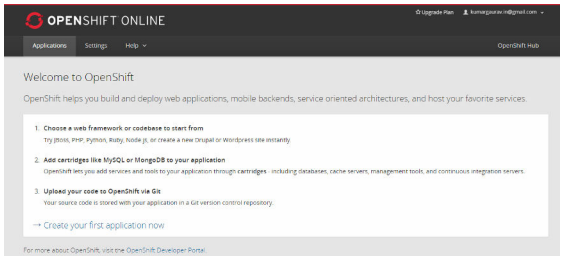
Hypervisor Type 2 Software Suite

- VMware Workstation
- VirtualBox

CREATING PHP APPLICATION PAGE IN RED HAT OPENSHIFT

OpenShift (http://www.openshift.com) is gained a lot of popularity with developers as it is having the enormous speed by which the quick spinning up a new development stack is possible.

To start using the OpenShift platform, the very first step a developer requires is an account that provides access to create and deploy applications.



After creating the new account, there is the option to create a new application. In the online platform, the complete code can also be uploaded using Git so that

International Journal of Computing and Corporate Research
ISSN (Online) : 2249-054X
Volume 6 Issue 3 May 2016
International Manuscript ID : 2249054XV6I3052016-090

the pre-built application can be launched on OpenShift Platform.

To work on a new application, the OpenShift platform presents the number of options which are in the interest of developer to use. If a Python application is to be developed, the OpenShift gives the platform to program and execute Python Code online.

After basic settings and selection of the public URL, a new application is created on which the code can be deployed and executed.

The changes in the existing code can be done using Git. For this Git tool is required to be installed so that that the modifications and uploading can be done.

In OpenShift, the creation and deletion of application is very easy. The earlier created application can be deleted anytime after use.

Methodology Used

- Development of a Novel and Effective Algorithmic Approach
- Implementation of Algorithm in Advance Java
- Implementation of Approach on Real Clouds
    - IBM Bluemix
    - RedHat OpenShift
- Fetch Results
- Comparative Analysis of the Results on multiple parameters
    - Cost Factor
    - Complexity
    - Execution Time
    - Performance
- Data Interpretation

Table 1 – Comparative Analysis on Assorted Parameters

	Cost	Complexity	Execution Time (ms)	Performance
IBM Bluemix	79	58	0.2324674	89
RedHat OpenShift	68	48	0.13838398	96

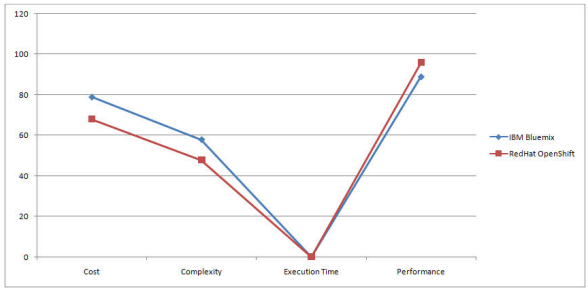


Figure 1 – Comparative Analysis

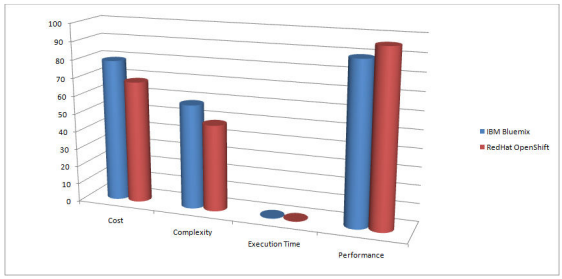


Figure 2 – Bar Graph of Comparative Analysis

Many social networks and apps have their own interface that programmers can work with. "These

interfaces are called APIs (short for Application Programming Interface).

Following is the results of Twitter-Java integration

```
/**
 * This Class is used to get the list of status.
 */
import java.util.*;
import twitter4j.*;
import twitter4j.Status;
import twitter4j.TwitterFactory;
import twitter4j.conf.ConfigurationBuilder;

public class Keyword {
        public static void main(String[] args)
throws TwitterException {

 ConfigurationBuilder cf=new
ConfigurationBuilder();
 cf.setDebugEnabled(true)

.setOAuthConsumerKey("56dgm39RKf2Mk64maypJ4
LzWb")

.setOAuthConsumerSecret("CT9RNduvafznnI5Gbqse
7kVlJSw3e55z1jkkbnMzqiqzjT7d2U")
 .setOAuthAccessToken("96367691-
TEJPMXkf7HqTHVVD7SJ3Xeu8zHikKbCFNGNAgU
tv7")

.setOAuthAccessTokenSecret("QTMDSl5udT8ux2LtZ
wbe849Kg8P8K0MZfd6PNW4bTOWhd");

 TwitterFactory tf=new TwitterFactory(cf.build());
 twitter4j.Twitter twitter = tf.getInstance();

 // The factory instance is re-useable and thread safe.
//   Twitter twitter = TwitterFactory.getSingleton();
   // requesting page 2, number of elements per page
is 40
   Paging paging = new Paging(2, 40);
   List<Status> status =
twitter.getUserTimeline(paging);
   for (Status st : status) {

System.out.println(st.getUser().getScreenName() +
":" + st.getText());
   }

   // requesting page 3, since_id is (long)1000
/*    statuses = twitter.getUserTimeline(new
Paging(3).sinceId(1000l));
   for (Status status : statuses) {

System.out.println(status.getUser().getScreenName()
+ ":" + status.getText());
   }
          */
}
}
/**
 * This Class is used to get the list of status.
 */
import java.util.*;
import twitter4j.*;
import twitter4j.Status;
```

```
import twitter4j.TwitterFactory;
import twitter4j.conf.ConfigurationBuilder;

public class FirstTwitterApp {
        public static void main(String[] args)
throws TwitterException {

 ConfigurationBuilder cf=new
ConfigurationBuilder();
 cf.setDebugEnabled(true)

.setOAuthConsumerKey("56dgm39RKf2Mk64maypJ4
LzWb")

.setOAuthConsumerSecret("CT9RNduvafznnI5Gbqse
7kVlJSw3e55z1jkkbnMzqiqzjT7d2U")
 .setOAuthAccessToken("96367691-
TEJPMXkf7HqTHVVD7SJ3Xeu8zHikKbCFNGNAgU
tv7")

.setOAuthAccessTokenSecret("QTMDSl5udT8ux2LtZ
wbe849Kg8P8K0MZfd6PNW4bTOWhd");

 TwitterFactory tf=new TwitterFactory(cf.build());
 twitter4j.Twitter twitter = tf.getInstance();

 List<Status> status=twitter.getHomeTimeline();
for (Status st : status)
 {

        System.out.println(st.getUser().getName()+"
-----"+st.getText());
 }
        }
}
```

CONFIGURATION OF TWITTER4J

Twitter4J is an unofficial Java library for the Twitter API.

With Twitter4J, you can easily integrate your Java application with the Twitter service. Twitter4J is an unofficial library.

Twitter4J is having following features -

- 100% Pure Java - works on any Java Platform version 5 or later
- Android platform and Google App Engine ready
- Zero dependency : No additional jars required
- Built-in OAuth support
- Out-of-the-box gzip support
- 100% Twitter API 1.1 compatible

Generic properties

There are a number of properties available for configuring Twitter4J. You can specify properties via twitter4j.properties file, ConfigurationBuilder class or System Property as follows : -

via twitter4j.properties

Save a standard properties file named "twitter4j.properties". Place it to either the current directory, root of the classpath directory.

debug=true

*oauth.consumerKey=**********
oauth.consumerSecret=**********
****
oauth.accessToken=**********
****
oauth.accessTokenSecret=**********
****
via ConfigurationBuilder
You can use ConfigurationBuilder class to configure Twitter4J programatically as follows:
ConfigurationBuilder cb = new ConfigurationBuilder();
cb.setDebugEnabled(true)

.setOAuthConsumerKey("**********")

.setOAuthConsumerSecret("**********
**********")

.setOAuthAccessToken("**********
**********")

.setOAuthAccessTokenSecret("**********
**********");
TwitterFactory tf = new TwitterFactory(cb.build());
Twitter twitter = tf.getInstance();
via System Properties
You can configure Twitter4J via System properties as well. Note that you need "twitter4j." prefix.
$ java -Dtwitter4j.debug=true
-Dtwitter4j.oauth.consumerKey=**********
****
-Dtwitter4j.oauth.consumerSecret=**********
****
-Dtwitter4j.oauth.accessToken=**********
****
-Dtwitter4j.oauth.accessTokenSecret=**********
****
-cp twitter4j-core-4.0.4.jar:yourApp.jar yourpackage.Main*

via environment variables

You can configure Twitter4J via environment variables as well. Note that you need "twitter4j." prefix. This makes it easier to test, stage and deploy apps running on Heroku.

*$ export twitter4j.debug=true
$ export twitter4j.oauth.consumerKey=**********
***
$ export twitter4j.oauth.consumerSecret=**********
****
$ export twitter4j.oauth.accessToken=**********
****

*$ export twitter4j.oauth.accessTokenSecret=**********
****
$ java -cp twitter4j-core-4.0.4.jar:yourApp.jar yourpackage.Main*

On Heroku:

*$ heroku config:add oauth.consumerKey=**********
$ heroku config:add oauth.consumerSecret=**********
$ heroku config:add oauth.accessToken=**********
****
$ heroku config:add oauth.accessTokenSecret=**********
****
$ git push heroku master*

CONCLUSION AND FUTURE WORK

Now days, most of computing services are available on demand which are hosted by number of cloud service providers. There are number of domains and key areas in cloud computing which are under research from a long time due to escalating need and applications of digital services.

As a result of the advances in the innovation, the issues and further extension in science and building are turning out to be more convoluted than any time in recent memory. To tackle these muddled issues, network figuring turns into a well known apparatus. A framework domain gathers, coordinates, and uses heterogeneous or homogeneous assets scattered the world over by a rapid system. A lattice situation can be ordered into two sorts: processing networks and information frameworks. This exploration work concentrates on employment booking calculations and their execution on numerous parameters in the lattice environment. In processing matrix, occupation booking is a critical undertaking. A decent booking calculation can allocate occupations to assets productively and can adjust the framework load.

For future scope of the work, following techniques can be used in hybrid approach to better and efficient results –

- Particle Swarm Optimization
- HoneyBee Algorithm
- Simulated Annealing
- Genetic Algorithmic Approaches

REFERENCES / BIBLIOGRAPHY

[1] Kune, R., Konugurthi, P. K., Agarwal, A., Chillarige, R. R., & Buyya, R. (2014, December). Genetic Algorithm based Data-aware Group Scheduling for Big Data Clouds. In Proceedings of the 2014 IEEE/ACM International Symposium on Big Data Computing (pp. 96-104). IEEE Computer Society.

[2] Mathew, T., Sekaran, K. C., & Jose, J. (2014, September). Study and analysis of various task scheduling algorithms in the cloud computing environment. In Advances in Computing, Communications and

Informatics (ICACCI, 2014 International Conference on (pp. 658-664). IEEE.

[3] Singh, S., & Kalra, M. (2014, November). Scheduling of Independent Tasks in Cloud Computing Using Modified Genetic Algorithm. In Computational Intelligence and Communication Networks (CICN), 2014 International Conference on (pp. 565-569). IEEE.

[4] Verma, A., & Kaushal, S. (2014). Deadline constraint heuristic-based genetic algorithm for workflow scheduling in cloud. International Journal of Grid and Utility Computing, 5(2), 96-106.

[5] Javanmardi, S., Shojafar, M., Amendola, D., Cordeschi, N., Liu, H., & Abraham, A. (2014, January). Hybrid job scheduling algorithm for cloud computing environment. In Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014 (pp. 43-52). Springer International Publishing.

[6] Rodriguez, M. A., & Buyya, R. (2014). Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds.Cloud Computing, IEEE Transactions on, 2(2), 222-235.

[7] Singh, L., & Singh, S. (2014). A Genetic Algorithm for Scheduling Workflow Applications in Unreliable Cloud Environment. In Recent Trends in Computer Networks and Distributed Systems Security (pp. 139-150). Springer Berlin Heidelberg.

[8] Shojafar, M., Javanmardi, S., Abolfazli, S., & Cordeschi, N. (2015). FUGE: A joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and a genetic method. Cluster Computing, 18(2), 829-844.

[9] Intisar A.Majied Al-Said et. al. (2015). Evolution of grid computing architecture and grid adoption models. *IBM Systems Journal*, *43*(4), 624-645.

[10] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A. & Warfield, A. (2003). Xen and the art of virtualization. ACM SIGOPS Operating Systems Review, 37(5), 164-177.

[11] Al-Fares, M., Loukissas, A., & Vahdat, A. (2008). A scalable, commodity data center network architecture. ACM SIGCOMM Computer Communication Review, 38(4), 63-74.

[12] Robinson, J., Sinton, S., & Rahmat-Samii, Y. (2002). Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. In Antennas and Propagation Society International Symposium, 2002. IEEE (Vol. 1, pp. 314-317). IEEE.

[13] Sailer, R., Valdez, E., Jaeger, T., Perez, R., Van Doorn, L., Griffin, J. L., ... & Berger, G. S. (2005). sHype: Secure hypervisor

approach to trusted virtualized systems. *Techn. Rep. RC23511*.

[14] Gu, J., Hu, J., Zhao, T., & Sun, G. (2012). A new resource scheduling strategy based on genetic algorithm in cloud computing environment. Journal of Computers, 7(1), 42-52.

[15] Pandey, S., Wu, L., Guru, S. M., & Buyya, R. (2010, April). A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on (pp. 400-407). IEEE.

[16] Kune, R., Konugurthi, P. K., Agarwal, A., Chillarige, R. R., & Buyya, R. (2014, December). Genetic Algorithm based Data-aware Group Scheduling for Big Data Clouds. In Proceedings of the 2014 IEEE/ACM International Symposium on Big Data Computing (pp. 96-104). IEEE Computer Society.

[17] Verma, A., & Kaushal, S. (2014). Deadline constraint heuristic-based genetic algorithm for workflow scheduling in cloud. International Journal of Grid and Utility Computing, 5(2), 96-106.

[18] Javanmardi, S., Shojafar, M., Amendola, D., Cordeschi, N., Liu, H., & Abraham, A. (2014, January). Hybrid job scheduling algorithm for cloud computing environment. In Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014 (pp. 43-52). Springer International Publishing.

[19] Ye, H. (2015). Research on Emergency Resource Scheduling in Smart City based on HPSO Algorithm. city, 5, 6.

[20] Pawar, A., Scholar, M. T., & Kapgate, P. D. (2014). A Review on Virtual Machine Scheduling in Cloud Computing. vol, 3, 928-933.

[21] Shojafar, M., Javanmardi, S., Abolfazli, S., & Cordeschi, N. (2015). FUGE: A joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and a genetic method. Cluster Computing, 18(2), 829-844.

[22] Zhang, F., Cao, J., Li, K., Khan, S. U., & Hwang, K. (2014). Multi-objective scheduling of many tasks in cloud platforms. Future Generation Computer Systems, 37, 309-320.

[23] Quang-Hung, N., Tan, L. T., Phat, C. T., & Thoai, N. (2014). A GPU-Based Enhanced Genetic Algorithm for Power-Aware Task Scheduling Problem in HPC Cloud. In Information and Communication Technology (pp. 159-169). Springer Berlin Heidelberg.

[24] Tsai, C. W., & Rodrigues, J. J. (2014). Metaheuristic scheduling for cloud: A survey. Systems Journal, IEEE, 8(1), 279-291.

[25] Lin, W., Liang, C., Wang, J. Z., & Buyya, R. (2014). Bandwidth-aware divisible task scheduling for cloud computing. Software: Practice and Experience,44(2), 163-174.

[26] Kumar, M., & Doegar, A. (2014). Reliable and Efficient Task Scheduling based on Genetic Algorithm in Cloud Computing Environment.

[27] Frîncu, M. E. (2014). Scheduling highly available applications on cloud environments. Future Generation Computer Systems, 32, 138-153.

[28] Yang, T., & Gerasoulis, A. (2014, June). Author retrospective for PYRROS: static task scheduling and code generation for message passing multiprocessors. In 25th Anniversary International Conference on Supercomputing Anniversary Volume (pp. 18-20). ACM.

[29] Leena, V. A., & Rajasree, M. S. (2016). Genetic Algorithm Based Bi-Objective Task Scheduling in Hybrid Cloud Platform. International Journal of Computer Theory and Engineering, 8(1),

[30] Xu, Y., Li, K., Hu, J., & Li, K. (2014). A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues.Information Sciences, 270, 255-287.

[31] Lin, J., Zhong, Y., Lin, X., Lin, H., & Zeng, Q. (2014). Hybrid Ant Colony Algorithm Clonal Selection in the Application of the Cloud's Resource Scheduling. arXiv preprint arXiv:1411.2528.

[32] Zdenek Konfrst, " Parallel Genetic Algorithms: Advances, Computing Trends, Applications and Perspectives", 18th International Parallel and Distributed Processing, 2004.

[33] Marin Golub, Leo Budin, "An Asynchronous Model of Global Parallel Genetic Algorithms" Unska 3, HR-10000 Zagreb, Croatia

[34] Multiprocessor booking Algorithm Based On Genetic Algorithm (Intisar A.Majied Al-Said, Nedhal Al-Saiyd, Firas Turki Attia)

[35] Multiprocessor Environment Using Genetic Algorithm (Volume 2, Issue 5, May 2012 ISSN: 2277 128X, Sandeep Jain, Shweta Makkar

[36] DMP: Deterministic Shared Memory Multiprocessing (Joseph Devietti Brandon Lucia Luis Ceze Mark Oskin Computer Science & Engineering, University of Washington)