 ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

Autonomic Fault Tolerant Framework for Web Applications

[1]Dhananjaya Gupta, [2]Anju Bala
[1,2]Dept. of CSE, Thapar University, Patiala, India

Abstract
Cloud computing is an emerging practice that offers more flexibility in infrastructure and reduces cost than our traditional computing models. Reliability, availability in Cloud computing are critical requirements to ensure correct and continuous system operation also in the presence of failures. Fault tolerance mechanism needs to be developed to deal with faults that can affect the normal operations in cloud environment. Web application because of growing demand is an area of concern in the IT security community. In this paper, a framework for providing a fault tolerance system for web based applications in cloud environment has been designed. Incoming requests for the web applications are balanced between the web servers and data generated is kept in multiple clusters that maintains data in replicated fashion. Storage clusters are continuously monitored for any kind of fault which could halt normal functioning of the system because of data loss. The experimental results depicts that the proposed framework monitors and handles various faults effectively.

Keywords
Cloud Computing, Fault Tolerance, Replication, Load Balancing, Monitoring

I. Introduction

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. Cloud computing model offers several benefits that include an improved peak-load handling and dynamic resource provisioning without the need for setting up a new software or hardware infrastructure in every location. Cloud computing improve services by optimizing the service location and throughput according to users' QoS needs. It provides higher reliability as providers can transparently migrate their services to other domains, thus enabling a highly resilient computing environment [3][4]. Cloud providers offer everything from access to raw storage capacity resources to complete application services in many areas such as payroll and customer relationship management etc. These web applications are a critical part of business. Employees, customers and partners prefer to do business online, and expect to be able to access a variety of information and transactions through Web sites and Web services [23]. Data flows through the network from one location to another while using the services provided by the cloud. However, providers of such environments must solve the challenge of proving fault free operating environment.

To handle faults, a fault tolerant system is required that could identify the fault at early stage before it becomes failure. The end users of a cloud computing network usually have no idea where the servers are physically located—they just spin up their application and start working. Web application because of growing demand is an area of concern in the IT security community. These applications run in a distributed environment. Servers handling requests may fail which could lead to service unavailability. Data in web applications needs to be taken care off because it is transferred between different locations at very high frequency. To handle these issues, new tools and approaches are needed to build reliable and fault tolerant systems [10].

The rest of the paper is organized as follows: section II describes some work related to our research and challenges in building autonomic fault tolerant system for web applications. Section III presents the System Design and is divided into two parts: A part shows the proposed Framework for autonomic fault tolerant System and B part shows the Layered Architecture. Section IV includes the implementation of a prototype using this Framework and the experimental results. Section V concludes the paper.

II. Related Work

cloud computing is a model for providing on-demand access to a wide variety of software applications, and to configurable computing resources that can be deployed to support whatever applications a user chooses to run. On- demand access and self-provisioning are key characteristics of cloud computing, and are one of the principal features that distinguish cloud computing from other types of hosted service [12].With the development of distributed systems, cloud computing provides a flexible and scalable approach of managing thousands of distributed heterogeneous nodes to make up integrate and high performance computing environment for users. However, when the size of cloud scales up, cloud computing is required to handle massive data accessing requests, such as distributed data mining [11]. For handling this massive data and data requests, Hadoop HDFS can be used which enables data to be stored on multiple locations. Data is broken into fixed size blocks and stored on DataNodes [14]. These blocks are replicated for fault tolerance and fast access. Replication is widely used mechanism to mitigate risk of data loss. Replication of data becomes a challenging task in case of Cloud computing as data is to be processed at multiple locations simultaneously. Hadoop computing model operates on data stored in HDFS using Execution Layer i.e. MapReduce [13]. But Hadoop fails to provide data if NameNode goes down. Because of NameNode failure, complete cluster becomes unavailable. Kaushal* et al [9] presents highly reliable system that is developed using HAProxy and can deal with various software faults for server applications in a cloud virtualized environment. Data is replicated from one server to another when there is some fault in one server. However if data becomes unavailable then this system fails to tolerate fault and provide the requested service.

There are many fault tolerance techniques available that deals with Virtual Machine (VM) migration, process migration, application migration to overcome the impact of fault [19-20]. Time series based pre-copy technique migrate VM from one source host to target host [21]. Data in form of pages is transferred in this approach. Proactive Live Process migration mechanism migrate process before the fault occurs [22]. These software faults may or may not manifest themselves during systems operations, but when they do, software fault tolerant techniques should provide the necessary mechanisms of the software system to prevent system failure occurrences [7]. Fault tolerance can be Re-active or Pro-active, depending on the nature of the fault. Reactive fault tolerance policies reduce the effect of failures on application execution when the failure effectively occurs. Various techniques are available to implement Re-active and pro-active fault tolerance

techniques [17]. By combination of these techniques, more efficient fault tolerant model can be prepared. Various tools are available through which these techniques can be implemented. In this paper, a framework for developing such model has been proposed. Implementation has been done using various tools and there experimental results are shown.

III. System Design

A. Framework for the Fault tolerant System

This section presents the Framework of the fault tolerant system for web based applications. The proposed framework for the fault tolerant web based system is shown in fig. 1. This framework provides a better Quality of service and fault tolerance for the web applications running in cloud environment. With the combination of load balancer, multiple data clusters and by monitoring these clusters, higher availability, better Quality of service and fault tolerance is ensured.

Various components of the framework are explained below:

1. Load Balancer

Load Balancer receives incoming requests for the web application and forwards those requests to the specific web interfaces. When a web interface is down or serving some other request then the Load Balancer forward this request to other interface which could handle it.

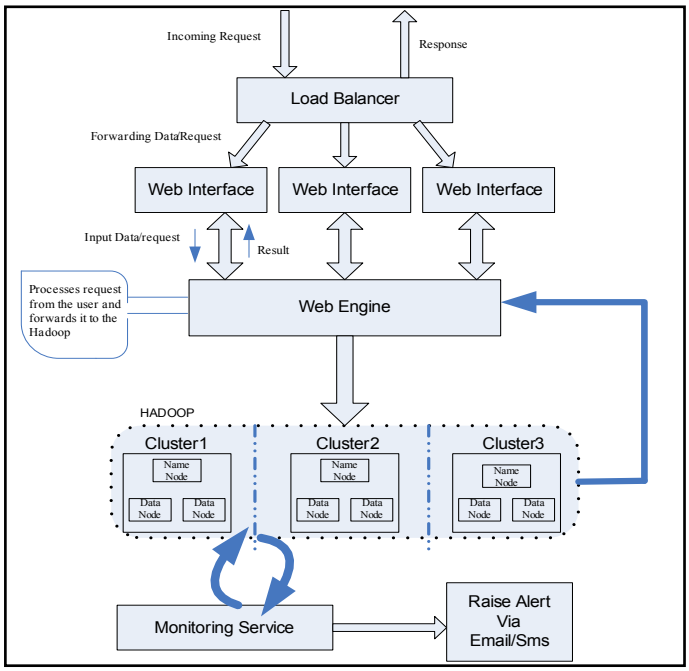

Fig. 1: Proposed Framework

2. Web Engine

Web engine is developed in java. It handles all the requests coming from various interfaces and provides the appropriate service needed to fulfill request. After processing request it forwards data to the storage cluster.

3. Hadoop Clusters

Processing of data is done at various locations and at times data moves from one location to another. Because of this maintaining data becomes one of the critical tasks. These clusters are used to store data that comes from the web Engine. Clusters maintain multiple copies of data. This increases the availability. Also when required, data is fetched from the nearest DataNode which ensures higher performance. Location of data becomes an important factor when we are dealing with distributed computing environment. In case, if one node goes down then data can be easily retrieved from another node.

4. Monitor

Monitoring tool sends alerts when critical infrastructure components fail and recover, providing administrators with notice of important events. It collects information regarding the number of clusters and DataNodes per cluster. If a DataNode or a cluster fails then it raises alert and asks for the appropriate action. Alerts can be delivered via email, SMS.

B. Layered Architecture

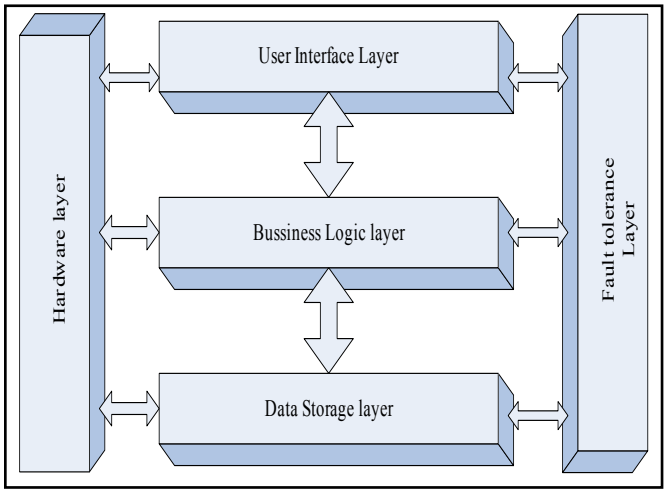

Fig. 2: Layered Architecture

Layered Architecture shown in fig. 2, distributes the overall functionality into five layers. Hardware layer provides the required hardware services such as Memory, CPU, Network card etc in shared environment. For the experimental purpose, virtualized shared environment has been created by using Virtual Machine Manager (VMM). Data Storage layer stores data that is generated at the Business logic layer. Business logic layer contains the implementation of Web engine. User interface layer interacts with the End user and receives requests and provides responses back to the user. Fault tolerance layer provides all three layers i.e. User Interface, Business Logic and Data Storage layer the ability to tolerate fault using different tools. Various tools used to provide such fault tolerance are as follows:

1. HAProxy (Load Balancer)

HAProxy stands for High Availability Proxy and is used by companies such as RightScale for load balancing and server fail over in the cloud. Companies do not want their website to go down, or worse, for users to notice the site is down. Due to this larger websites will run on multiple servers to provide some level of high availability. In HAProxy there is typically a load balancer to distribute the load among a pool of web servers. It also works as a fault tolerant system. Whenever a server goes down it is taken out of the pool until it is once again ready to handle requests. HAProxy has the ability to perform this task by doing periodic health checks on all the servers in a cluster. Even if one of the application servers is not working, users will still have the availability to the application. HAProxy will properly handle the request from users by redirecting them to the second server, giving the impression that all is well [5].

HAProxy is a free, very fast and reliable solution offering high availability, failover and load balancing mechanism, and proxying for TCP and HTTP based applications.

2. Hadoop

Hadoop [1,13-14] provides a distributed file system and a framework for the analysis and transformation of very large data sets using the MapReduce paradigm. HDFS is the file system component of Hadoop. HDFS stores file system metadata and application data separately. As in other distributed file systems, like PVFS [15] and Lustre [16], HDFS stores metadata on a dedicated server, called the NameNode. Application data are stored on other servers called DataNodes. All servers are fully connected and communicate with each other using TCP-based protocols.

3. Nagios

Nagios monitors entire IT infrastructure to ensure systems, applications, services, and business processes are functioning properly. In the event of a failure, Nagios can alert technical person of the problem, allowing them to begin remediation processes before outages affect business processes, end-users, or customers [6]. Nagios has a modular design with a web interface and a set of plugins to check the different services. Nagios is not specifically an IDS. On the other hand, Nagios possesses a friendly interface, is easy to use, very flexible and it is endowed a system of sending alerts.

IV. Implementation and Experimental Results

We have implemented a prototype using HAProxy as a Load Balancer, Hadoop HDFS for storing data in replicas and Nagios as a monitoring tool. Two web pages have been designed and are deployed on web servers running on different machines. HAProxy distributes the incoming load on these servers. Using difference scheduling algorithms, the behavior of load balancing can be controlled. Also if one server fails then HAProxy can recover from failure in few milliseconds and starts providing service via second server.

In this prototype we have used round robin scheduling. HAProxy provides a web interface for statistics known as HAProxy stats. Interface login requires data configured in HAProxy configuration file. HAProxy stats provide a review of the current availability of cluster system. If server status is up in HAProxy stats then server is live and working properly. Using configured address, web server can be accessed. End user sends a request to access web application. This request goes to the Load balancer. Load balancer has a cluster of web server which handles these requests. It balances the load between these servers and forward requests to these servers. Server takes input data from the end user and sends it to Web engine.

Web engine does the processing job. It provides the requested service and then data generated is passed to Hadoop HDFS. It receives data and looks for the source where it came from. After determining the source, it pumps data into various clusters accordingly. These clusters contains two data nodes each i.e. replication factor is kept to be 2. Maintaining two copies of data ensures higher data availability and fault tolerance is better. If one data node fails then data can be easily retrieved from other node seamlessly.

Keeping replicas of data at multiple nodes takes extra storage but it provides better performance than if data is kept at single location only. When required, data is taken from the nearest location possible. It reduces data transfer time as well as cost. Since data is provided by the nearest location so bisection bandwidth usage is reduced. Also these clusters are being monitored by Nagios. It keeps a check on the status of clusters. If a DataNode or a cluster goes down then Nagios raises alert. Fault tolerance of the system has been evaluated by considering certain cases given below:

Case 1: when storage clusters and web servers are working properly, HAProxy chooses the web server to serve request according to configurations made in the file. HAProxy statistics are shown in fig. 3. Two storage clusters and two web servers are configured and there status shows that all are working properly.

Case 2: Web interface for NameNode of one storage cluster is shown in fig. 4. This interface updates the information after few seconds. Under this NameNode, two DataNodes are configured and both are live. Replication factor is kept to be two.

Case 3: when there is some fault in memory, number of user exceeds, some DataNode goes down then these faults are monitored by Nagios. Fig. 5 shows currently there are seven services being monitored by Nagios and there status shows all are working correctly.

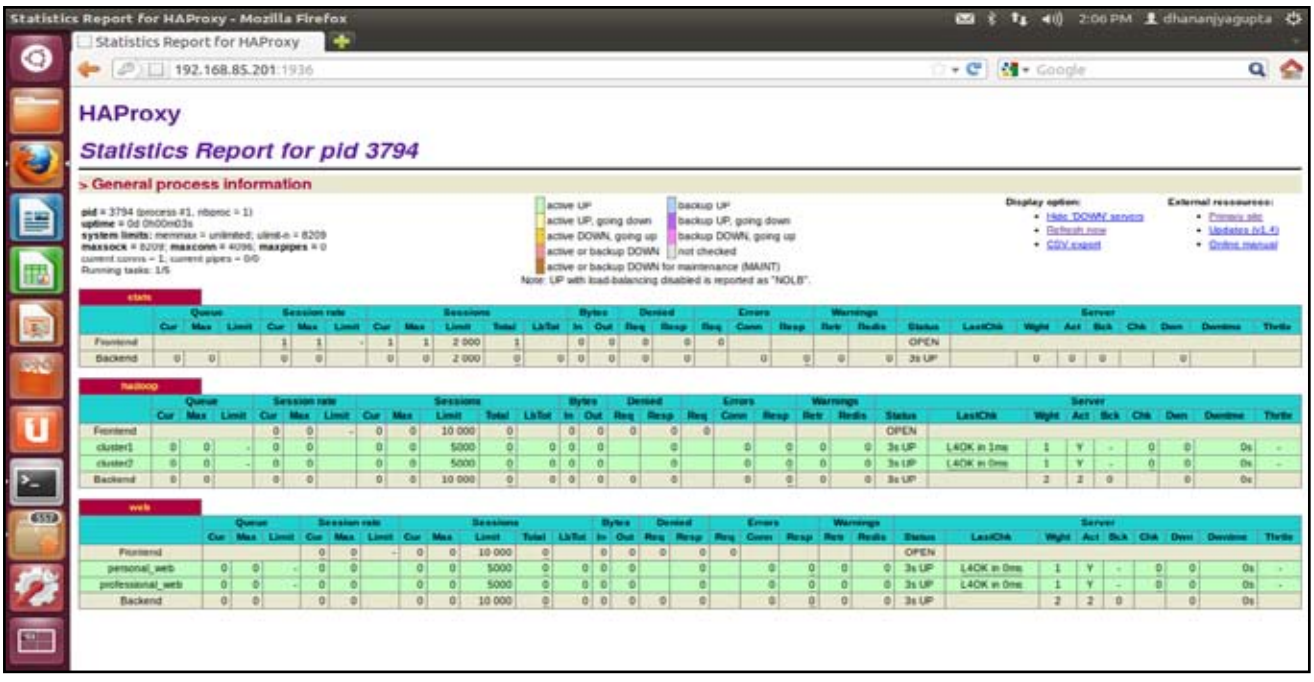

Fig. 3: HAProxy Statistics

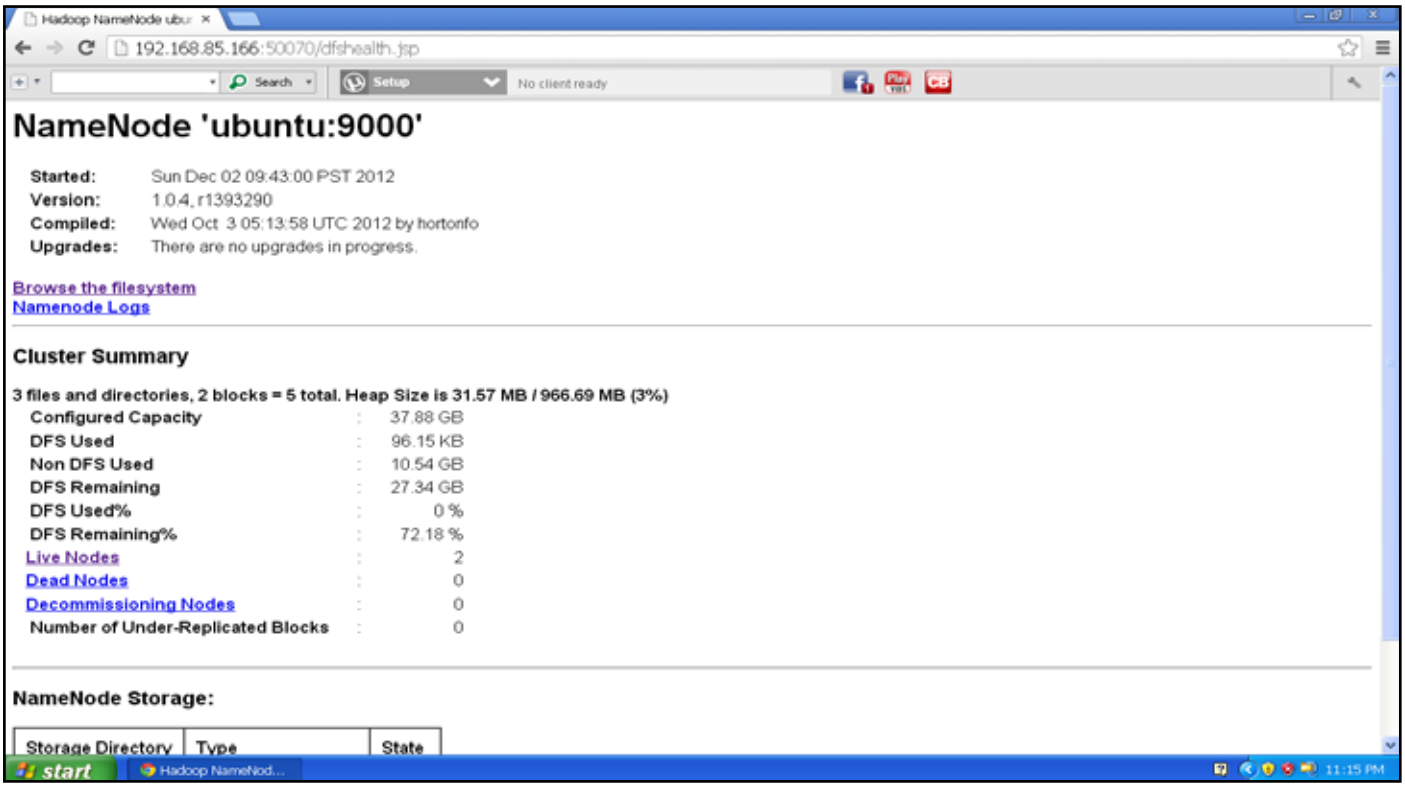

Fig. 4: Hadoop Cluster

A cluster has two live nodes i.e. DataNodes as presented in fig. 4. Similarly another cluster is also configured as shown in HAProxy statistics. Since replication factor is kept to be two. So if one DataNode fails then still data is provided by the second DataNode. Nagios observes the failure of the Data Node and reports this failure to the administrator.

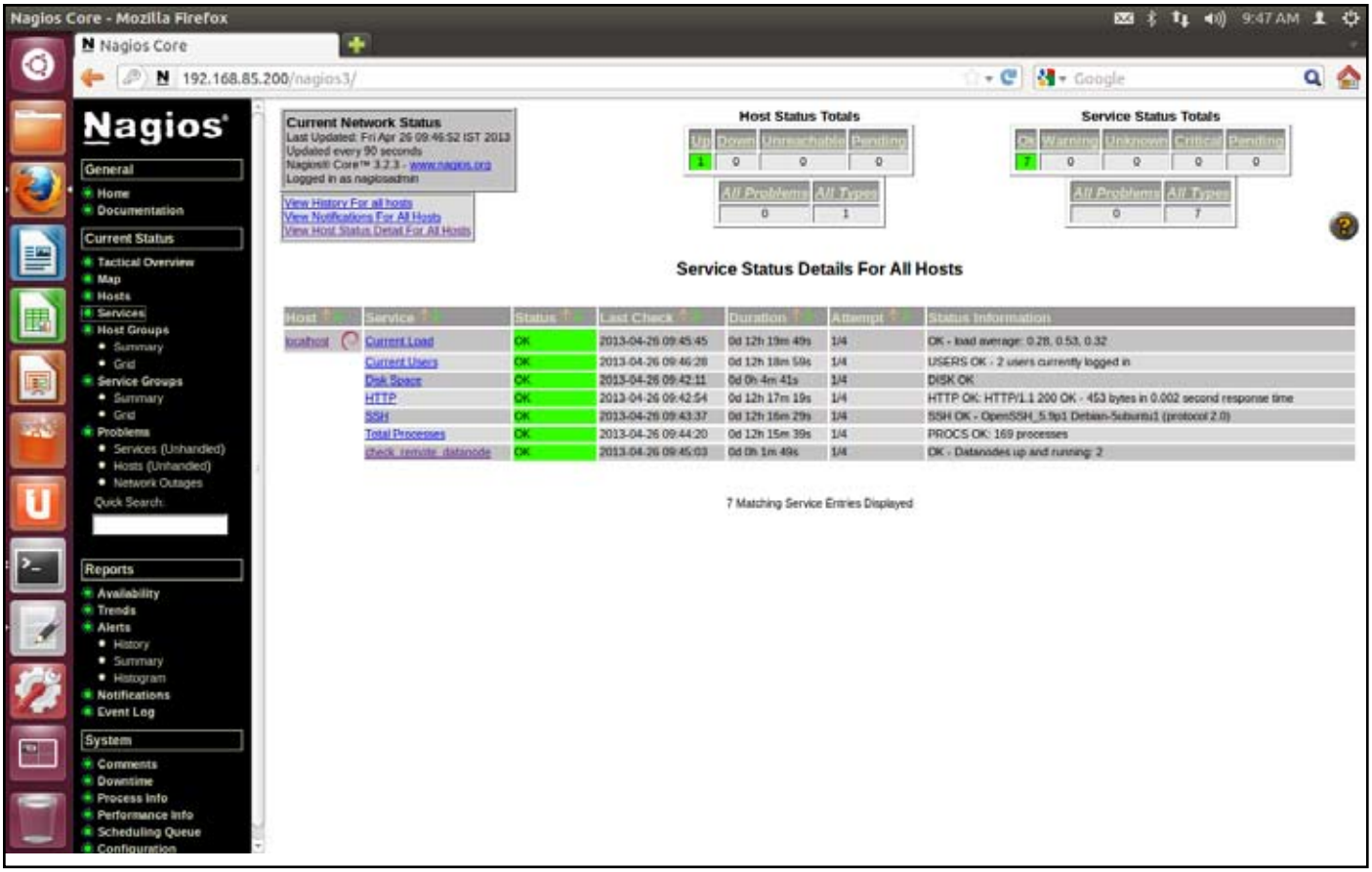

Fig. 5: Services Monitored by Nagios

The services being monitored by the Nagios are presented in fig. 5. Nagios statistics shows that all the services are working properly.

ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

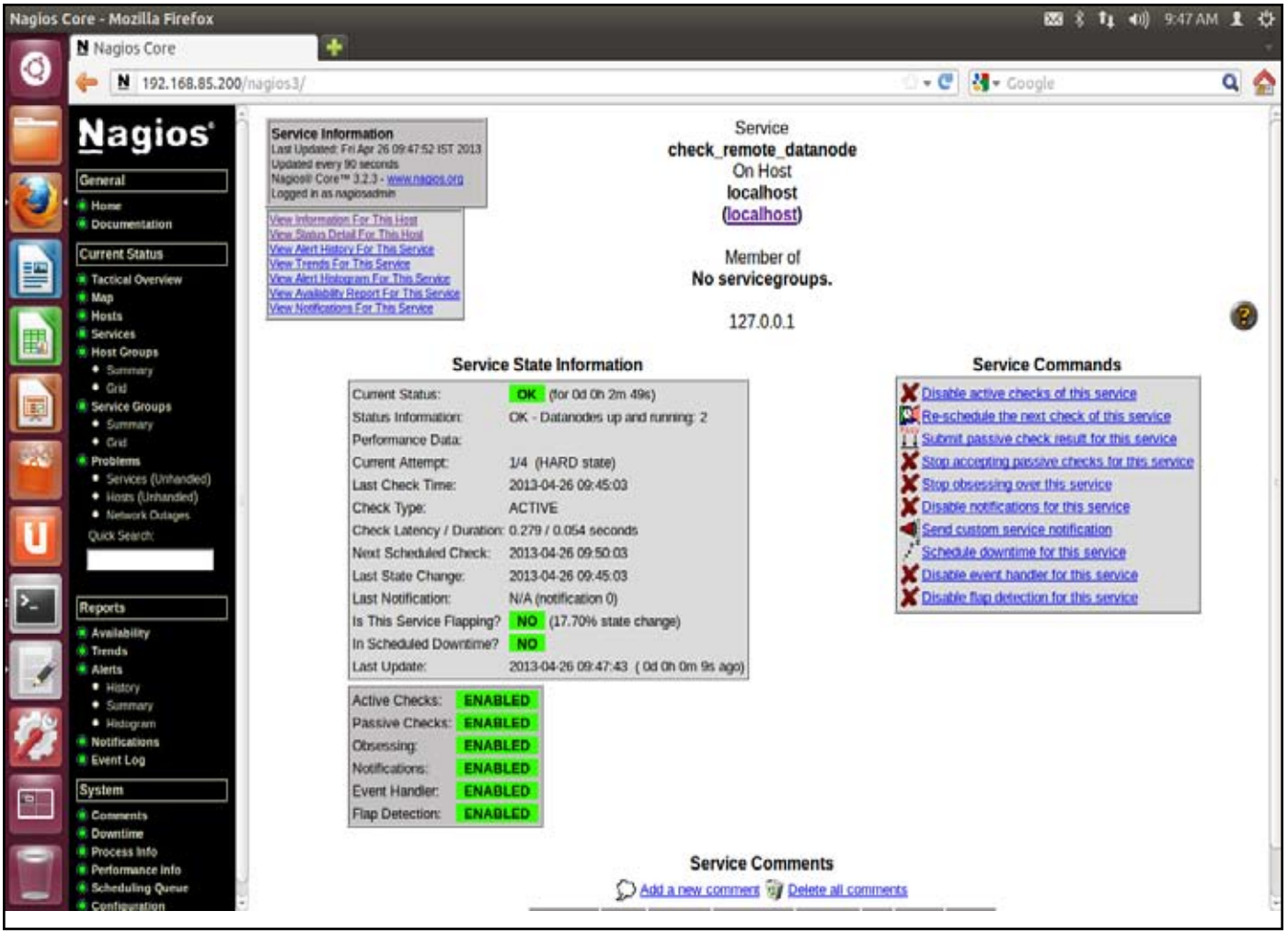

Fig. 6(a): Status of DataNodes

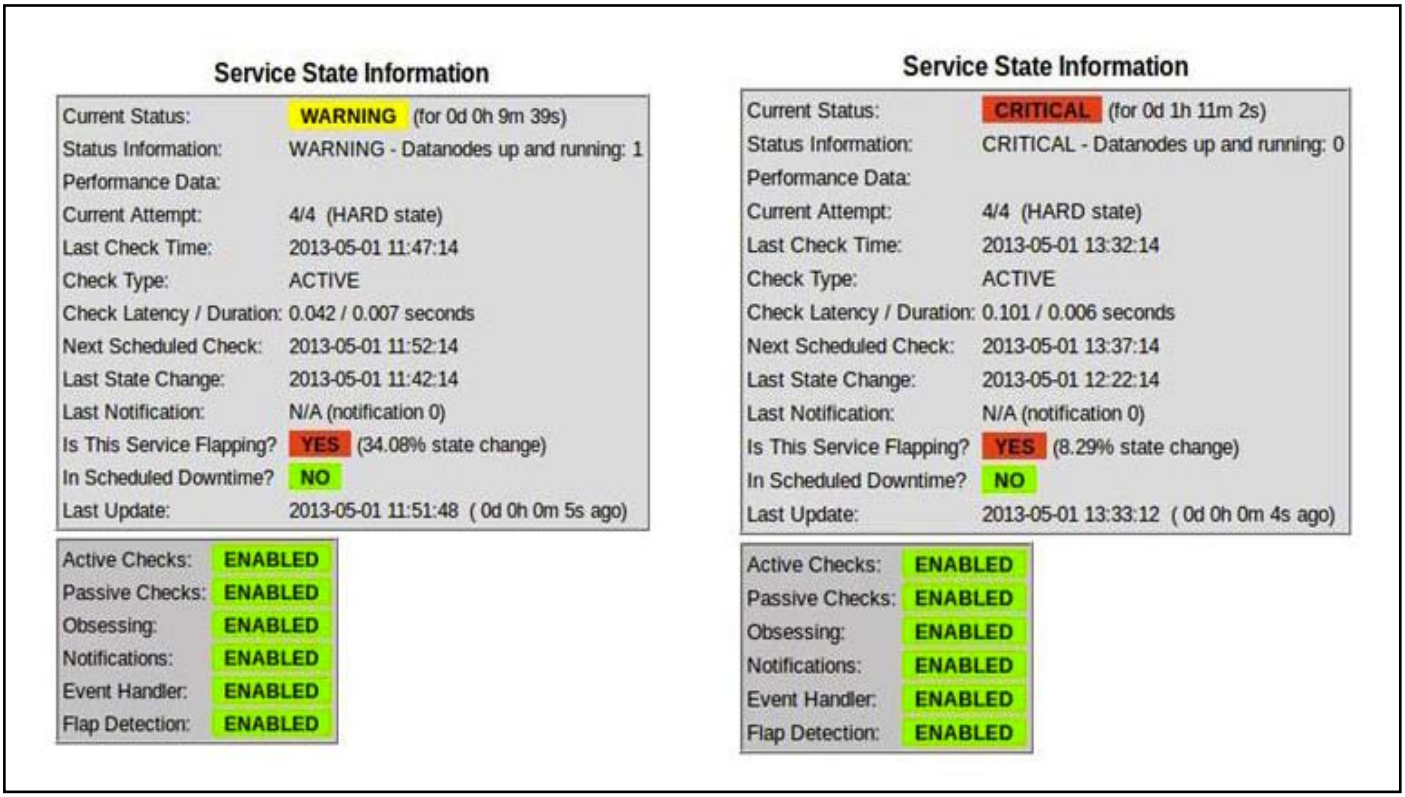

Figure 6(b): alerts raised by Nagios

The active DataNodes are constantly monitored by Nagios as shown in fig. 6(a) to ensure higher availability. If DataNode's count reaches a threshold value then Nagios changes its status to warning and if it decreases than threshold then it changes its status to critical and raises alert as depicted in fig. 6(b).

V. Conclusion and Future Scope

In this paper, we have proposed a framework that provides fault tolerance for web based applications in cloud environment. We have presented highly reliable system that can deal with various software faults for web based applications in a cloud environment. Multiple web applications are deployed on a VMM. These applications are evaluated and experimental results are obtained, that validate the system fault tolerance. The web engine presented in the framework can be a bottleneck if it is overloaded with the incoming requests as requests from all the interfaces are fed to this engine. This may degrade the performance because response time to every request would be increased. More efficient algorithms can be implemented to parallelize the operations in web engine that could increase the performance.

References

[1] Apache Hadoop, [Online] Available: http://hadoop.apache.org/

[2] HDFS (hadoop distributed file system) architecture, [Online] Available: http://hadoop.apache.org/common/docs/current/hdfs, design.html, 2009.

[3] Buyya R, Ranjan R.,"Special Section: Federated Resource Management in Grid and Cloud Computing Systems", Future Generation Computer Systems 2010, 26(8), pp. 1189-1191.

[4] Wayne P.,"Top Business and Operational Advantages Cloud Computing Can Deliver to IT Organizations", SunGard Availability Services; 2010. [Online] Available: http://www.dssresources.com/ news/3120.php [2011.11.12].

[5] [Online] Available: http://www.aproxy.1wt.eu/download/1.3/doc/configuration.txt

[6] Nagios. [Online] Available: http://www.nagios.org.

[7] Deepal Jayasinghe, Hyojun Kim, Mohammad M. Hossain, Ali Payani,"EWeb: Highly Scalable Client Transparent Fault Tolerant System for Cloud based Web Applications", ECE6102 Dependable Distribute Systems, fall 2010.

[8] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler,"The Hadoop Distributed File System", IEEE, 2010.

[9] Vishonika Kaushal* et al. / (IJAEST) INTERNATIONAL JOURNAL OF ADVANCED ENGINEERING SCIENCES AND TECHNOLOGIES Vol. 7, Issue 2, pp. 222 – 227.

[10] Amazon EC2,"Amazon Elastic Compute Cloud, [Online] Available: http://www.aws.amazon.com/ec2/.

[11] Ang Xu, Lei Wu, Liying Guo, Zheng Chen Lai Yang, Zhongzhi Shi,"An Intelligent Load Balancing Algorithm Towards Efficient Cloud Computing", AAAI workshop, 2011.

[12] Trevor Lorimer, Roy Sterritt,"Autonomic Management of Cloud Neighborhoods through Pulse Monitoring", Fifth International Conference on Utility and Cloud Computing, IEEE/ACM , 2012.

[13] J.Venner,ProHadoop.Apress, June22, 2009.

[14] T. White,"Hadoop: The Definitive Guide", O'Reilly Media, Yahoo! Press, June 5, 2009.

[15] W. Tantisiriroj, S. Patil, G. Gibson,"Data-intensive file systems for Internet services: A rose by any other name", Technical Report CMU- PDL-08-114, Parallel Data Laboratory, Carnegie Mellon University, Pittsburgh, PA, October 2008.

[16] Lustre File System, [Online] Available: http://www.lustre.org.

[17] Anju Bala, Inderveer Chana,"Fault Tolerance- Challenges, Techniques and Implementation in Cloud Computing", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No. 1, January 2012.

[18] Sushant Goel, Rajkumar Buyya,"data replication strategies in wide area distributed systems".

[19] Chao Wang, Frank Mueller, Christian Engelmann, Stephen L. Scott,"Proactive Process-Level Live Migration in HPC Environments".

[20] Gang Chen, Hai Jin, Deqing Zou, Bing Bing Zhou, Weizhong Qiang, Gang Hu,"SHelp: Automatic Self- healing for Multiple Application Instances in a Virtual Machine Environment", IEEE International Conference on Cluster Computing, 2010.

[21] Bolin Hu, Zhou Lei, Yu Lei, Dong Xu, Jiandun Li,"A Time-Series Based Precopy Approach for Live Migration of Virtual Machines", IEEE 17th International Conference on Parallel and Distributed Systems, 2011.

[22] Chao Wang, Frank Mueller, Christian Engelmann, "Proactive process level live migration and back migration in HPC environments", 2011.

[23] Sachin D Choudhari, Dr. S. K. Shrivastava,"Analysis of Web Application Vulnerability in Cloud Computing", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 2, Issue 7, July 2012.

Dhananjaya Gupta was born in Patiala, Punjab, India in 1990. He received is B.Tech degree in computer science engineering from Maharishi Markandeshwar University, in 2011. Currently he is a postgraduate student of software engineering from Thapar University. His interest is focused on cloud computing.

Anju Bala completed her B.Tech in Computer Science (1999) and M.Tech. in Information Technology from Pbi. University (2004) and pursuing Ph.D. in Cloud Computing from Thapar University, Patiala (2009) and has over eleven years of teaching experience. She is working as Assistant Professor in Computer Science and Engineering Department of Thapar University, Patiala. Her research interests lie in Fault Tolerance in Cloud Computing and Workflow management in Cloud Computing. She has over 16 publications in Conferences and International Journals of repute.