# An Effective Way to Handle Failover in Client Server Architecture

**Jay S. Raval[1] Ravi Raval[2]**
[1]Department of Information Technology [2]Department of Computer Engineering
[1,2]Silver Oak College of Engineering and Technoglogy

*Abstract*— Failover is the operational procedure of exchanging amongst essential and optional frameworks or framework (a server, processor, system, or database) in case of downtime. Such downtime could be brought by either planned support, or unpredicted framework or segment disappointment. In either case, the question is to make adaption to internal failure – to guarantee that mission-basic applications or frameworks are always accessible, paying little heed to the sort or degree of the blame. Because of the different reasons for Client Server Architecture failures, taking care of failover must be vital. So, various types of Failover solutions are implemented to handle failover in client-server architecture. In this, we analyse causes of failures and various solutions of failover and based on that try to handle failover with pacemaker and High Availability (HA) in centos Linux with database replication techniques.

***Key words:*** Failover, Switch over Time, PaceMaker, High Availability, CentOS, Replication of database.

## I. Introduction

In computing, failover is switching to a redundant or standby computer server, system, hardware component or network upon the failure or abnormal termination of the previously active application, server, system, hardware component, or network. Failover is automatic and usually operates without warning. Systems designers usually provide failover capability in servers, systems or networks requiring continuous availability – the used term is high availability – and a high degree of reliability. At the server level, failover automation sometimes uses a "heartbeat" system that connects two servers, either through employing a separate cable or a network affiliation. As long as a daily "pulse" or "heartbeat" continues between the most server and also the second server, the second server won't bring its systems on-line.[10] There may additionally be a third "spare parts" server that has running spare elements for "hot" shift to stop period. The second server takes over the work of the primary as presently because it detects associate alteration within the "heartbeat" of the primary machine. Some systems have the flexibility to send a notification of failover.

## II. Domain Introduction

Failover is the operational process of switching between primary and secondary systems or system components (a server, processor, network, or database) in the event of downtime. Such period of time may well be caused by either care, or unexpected system or part failure. In either case, the item is to make fault tolerance – to make sure that mission-critical applications or systems area unit perpetually offered, notwithstanding the sort or extent of the fault.[6][7]

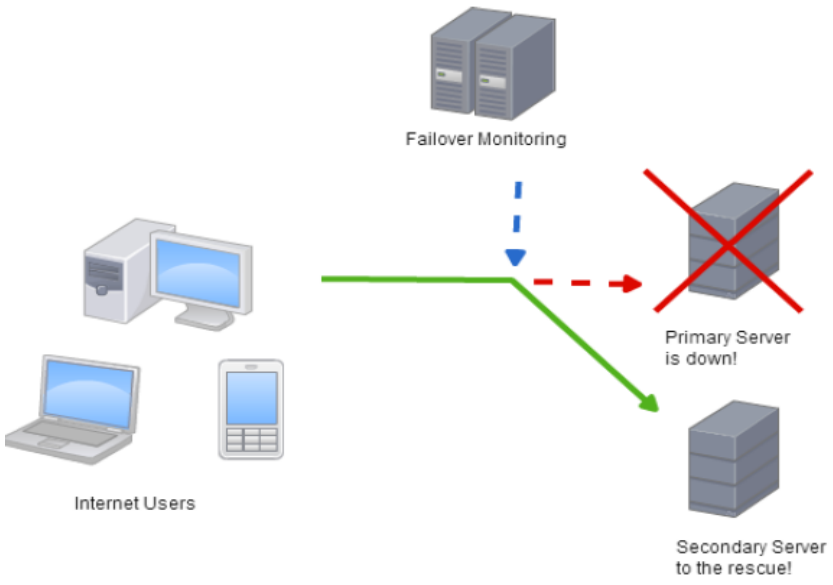


Fig. 1: Failover Overview [7]

### A. *Causes of Failures in Client-Server Architecture:*

Websites experience permanent, intermittent or transient failures, which may affect the entire site or be restricted to parts of the site. Permanent failures persist till the system the system is repaired, e.g., a disconnected network cable. [8][10] Transient failures eventually disappear without any apparent intervention. Intermittent failures area unit transient failures that recur often, e.g. once the system is full. In this section, we explore the common causes of failures in Web applications. We classified the causes of failures in Client-Server Architecture into below categories namely: a. Software Failures b. Human / Operator Error c. Hardware Failures

### *B. Client-Server Architecture Failover Solution:*

There are four main failover solutions

- Website Replication
- Status Monitoring
- Failover Redirection
- Recovery Handlin

### *C. About Failover and High Availability:*

Failover is the mechanism to provide fault-tolerance system to overcome from the problem of failure. Replication is the technique to achieve backup and failover. The techniques of failover used to increase the availability of the network services and not to provide any fault tolerance request at the time of server failure. [1][2]

High Availability (HA) is a capacity of the framework to work ceaselessly in fancied measure of time. Communication framework, for instance, ought to work 99.999%, that implies the framework ought to have just 5.26 greatest downtime for a year. Give high accessible IP based administration, for example, Voice over IP for communication is troublesome since IP is not intended for solid association. Failover component of high accessibility arrangement utilizing server virtualization is essential. High Availability also achieved by placing some middleware which redirects each and every requests to the other site in the failure situation. [3][4]

Failover Time or Switchover time is the time to transfer the current traffic to the other site or backup server. It is much more necessary to focus on the switchover time during failover as it affects the current services and traffic. The main objective should be to achieve and implement failover in such a way that the switching time is negligible. [5]

## III. PROPOSED METHODOLOGY

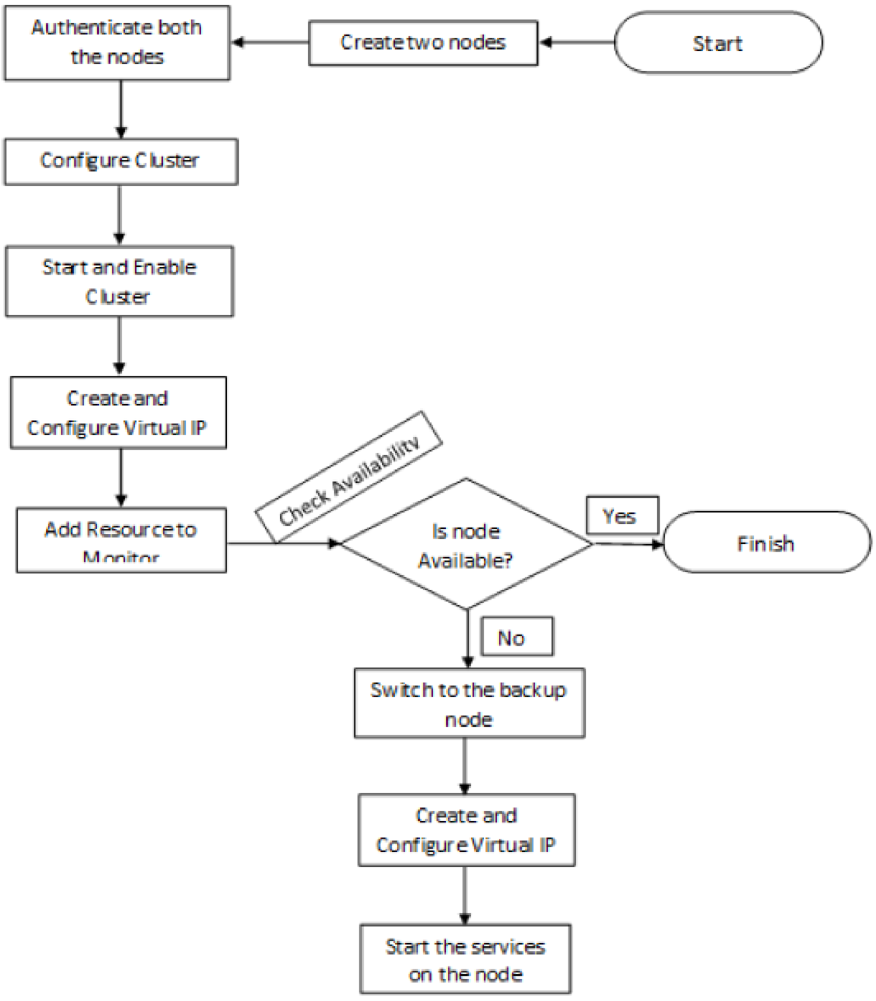

Start
Create two nodes
Authenticate both the nodes
Configure Cluster
Start and Enable Cluster
Create and Configure Virtual IP
Add Resource to Monitor
Check Availability
Is node Available?
Yes
Finish
No
Switch to the backup node
Create and Configure Virtual IP
Start the services on the node

Fig. 2: Proposed flow Diagram

Many researcher work on identifying various reasons for client-server architecture failover and many of them provides various failover mechanisms, strategies and solutions to handle the failover scenario. Here, we are implementing failover solution by using PaceMaker services of CentOS (Linux) [9] [11] with checking HA (High Availability) [12] of the servers, replicating database using master-master replication and analysing effective switchover time.

Fig. 2 is the proposed flow diagram which provides the work flow for the research, study and implementation.

The Fig. 2 shows the proposed flow diagram. Here, firstly, we will create two nodes and authenticate them. Once the authentication is done, we will configure cluster and start and enable cluster. Then, we will configure Virtual IP on which both nodes will listen. The monitoring resource will be added. Now, when the availability of the node is broken, the traffic will switch over to the backup node and the Virtual IP will be configured and listened to that node.

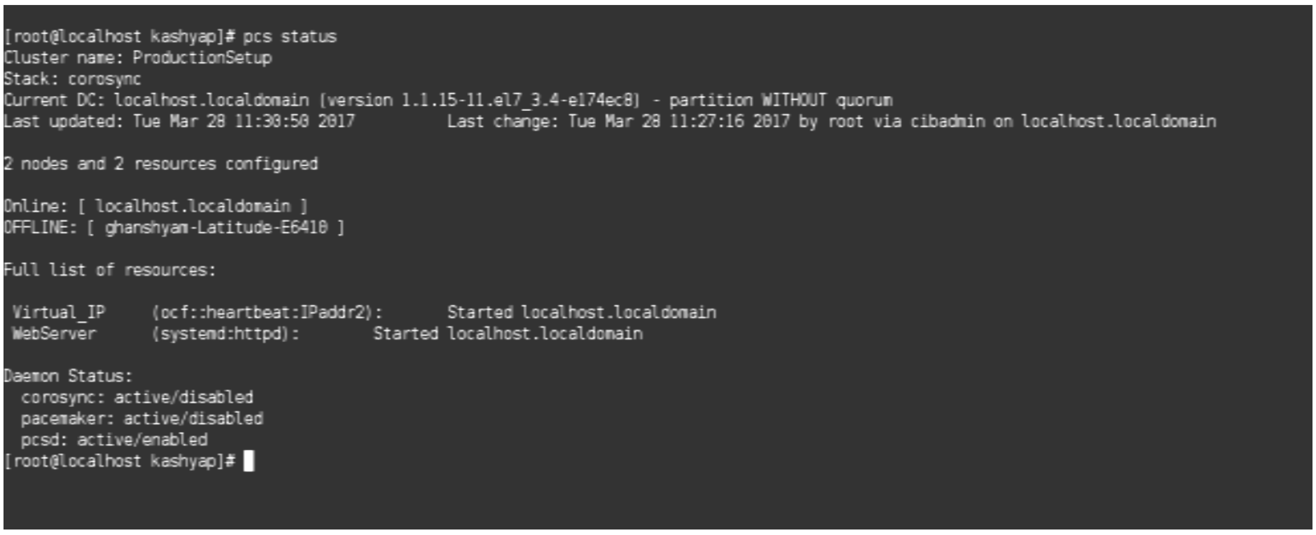
## IV. Implementation and Result

According to the proposed method and work flow, we have implemented setup to handle failover. There are two nodes which is active at a time and pointing to the Virtual IP. Whenever any node is failed due to any reason, using proposed method, the failover will occur.

We have used PaceMaker service to establish the connectivity and provide Availability of the node. For monitoring the services, Coro sync will be used which is the child service of PaceMaker.

The Web Server service apache will be monitored here, as we are implementing failover solution for any Web based application.

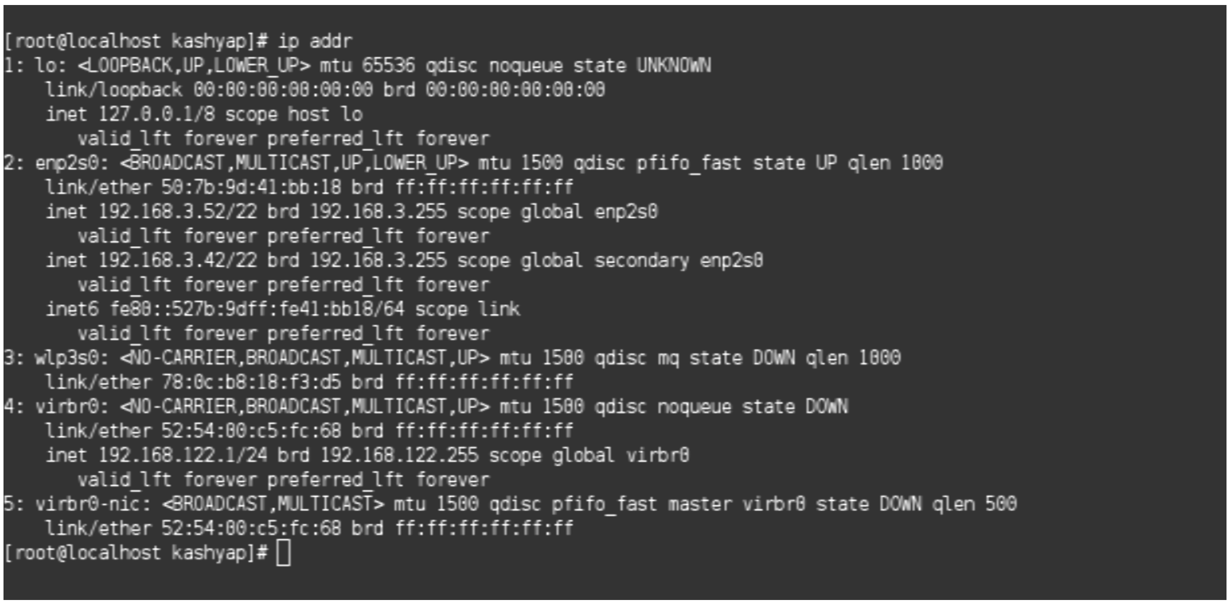
The Fig.3 shows the status of the nodes and resources configured on the server. Also it shows the Active and Inactive node status. Cluster name and other daemon service status can also be seen in the figure.



```
[root@localhost kashyap]# pcs status
Cluster name: ProductionSetup
Stack: corosync
Current DC: localhost.localdomain (version 1.1.15-11.el7_3.4-e174ec8) - partition WITHOUT quorum
Last updated: Tue Mar 28 11:30:50 2017          Last change: Tue Mar 28 11:27:16 2017 by root via cibadmin on localhost.localdomain

2 nodes and 2 resources configured

Online: [ localhost.localdomain ]
OFFLINE: [ ghanshyam-Latitude-E6410 ]

Full list of resources:

 Virtual_IP     (ocf::heartbeat:IPaddr2):       Started localhost.localdomain
 WebServer      (systemd:httpd):        Started localhost.localdomain

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
[root@localhost kashyap]# █
```

Fig. 3: pcs status

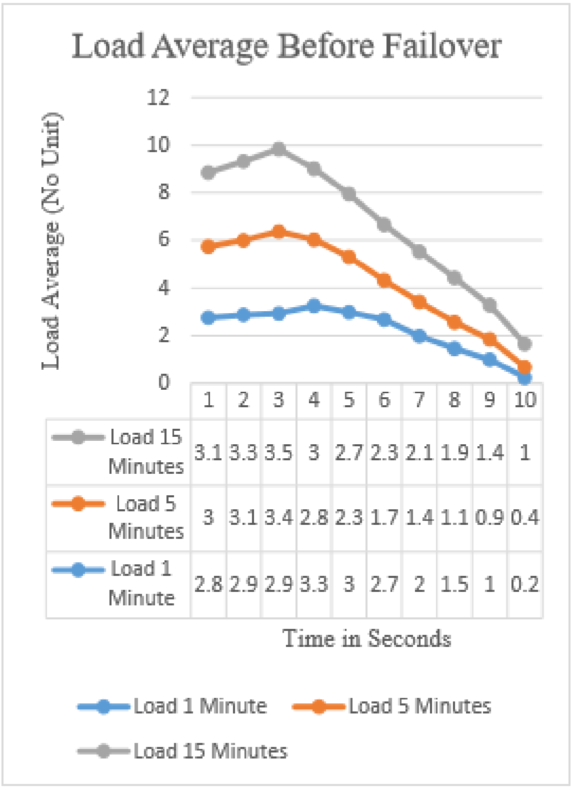
The Fig.3 shows the configured IP address status. It shows the private IP of the two node and a Virtual IP which is listening by any of the node at a particular time.



```
[root@localhost kashyap]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 50:7b:9d:41:bb:18 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.52/22 brd 192.168.3.255 scope global enp2s0
       valid_lft forever preferred_lft forever
    inet 192.168.3.42/22 brd 192.168.3.255 scope global secondary enp2s0
       valid_lft forever preferred_lft forever
    inet6 fe80::527b:9dff:fe41:bb18/64 scope link
       valid_lft forever preferred_lft forever
3: wlp3s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN qlen 1000
    link/ether 78:0c:b8:18:f3:d5 brd ff:ff:ff:ff:ff:ff
4: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
    link/ether 52:54:00:c5:fc:68 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
5: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN qlen 500
    link/ether 52:54:00:c5:fc:68 brd ff:ff:ff:ff:ff:ff
[root@localhost kashyap]# ▯
```

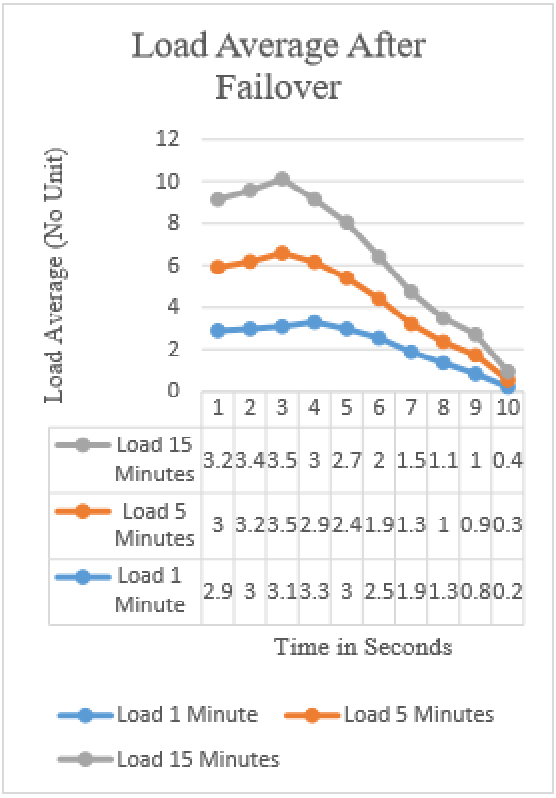
Fig. 4: IP address

The Fig. 5 and 6 shows the Load Average during the failover process. The framework load is an estimation of the computational work the framework is performing. This estimation is shown as a number. A totally sit without moving PC has a heap normal of 0. Each running procedure either utilizing or sitting tight for CPU assets adds 1 to the heap normal. Fig. 5 shows the Load Average process before the failover and Fig. 6 shows the Load Average process after the failover.



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Load 15 Minutes | 3.1 | 3.3 | 3.5 | 3 | 2.7 | 2.3 | 2.1 | 1.9 | 1.4 | 1 |
| Load 5 Minutes | 3 | 3.1 | 3.4 | 2.8 | 2.3 | 1.7 | 1.4 | 1.1 | 0.9 | 0.4 |
| Load 1 Minute | 2.8 | 2.9 | 2.9 | 3.3 | 3 | 2.7 | 2 | 1.5 | 1 | 0.2 |

Fig. 5: Load Average before Failover



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Load 15 Minutes | 3.2 | 3.4 | 3.5 | 3 | 2.7 | 2 | 1.5 | 1.1 | 1 | 0.4 |
| Load 5 Minutes | 3 | 3.2 | 3.5 | 2.9 | 2.4 | 1.9 | 1.3 | 1 | 0.9 | 0.3 |
| Load 1 Minute | 2.9 | 3 | 3.1 | 3.3 | 3 | 2.5 | 1.9 | 1.3 | 0.8 | 0.2 |

Fig. 6: Load Average after Failover

The result can be seen by the comparison that the resource utilization and optimization before and after the failover doesn't change much. Although, we can get some improved result after failover in the process of resource utilization. The Table 1 shows the time difference or delay or switchover time during the failover. The first execution time is concerned with the time when the failover occurred and the second execution time is concerned with the time when the backup node will be active after failover.

| Switchover Time | | | |
|---|---|---|---|
| SrNo. | Execution Time after failover (in ms) | Execution time after node is UP (in ms) | Diff. of delay (Switch over Time) (in ms) |
| 1 | 378 | 14344 | 13966 |
| 2 | 144 | 12605 | 12461 |
| 3 | 79 | 4072 | 3993 |
| 4 | 423 | 2469 | 2046 |

Table 1: Switch over time during failover

The result shows the improvement in delay time or the switchover time during the failover. So, the way we have implemented failover is giving us better switchover time as well.

The Fig.7 shows the graphical representation of the switchover time analysis. So, the switching delay can be monitored effectively. As we can observe the data in the chart that after attempting multiple experiments of failover, the switching over time is decreased and improved. And so the execution of failover is done via fault-tolerance in which the traffic on the failed node much affect too much.

Also to sync the data on both the nodes, we have implemented master-master replication in MySQL database. The replication will sync the data from master to slave every time when the alteration is done in master database. So, if the master node is failed, then the backup node will be active and the Web and Data both will be synced.

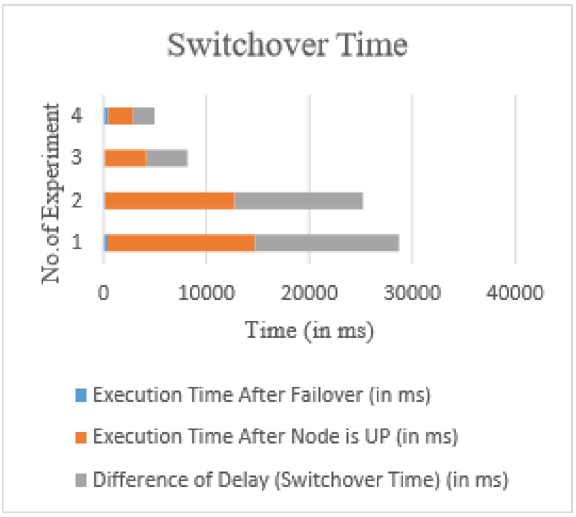


Fig. 7: Switchover Time analysis during failover

## V. Scope and Conclusion

In this paper, we review the causes of Client-Server Architecture Failure, Failover mechanism, Failover Strategies and different solution and also give the literature review of different papers. We have implemented failover with an effective way by monitoring Switchover Time using PaceMaker service of CentOS. Also replication will sync the data on both the nodes. So, we conclude here, one of the effective way to handle failover in client-server architecture. There will be a scope to implement it with more effective way considering other environment and also focusing on other parameters like Memory Usage, CPU Usage.

## References

[1] Tarandeep Singh, Parvinder S. Sandhu, and Harbax Singh Bhatti, “Replication of Data in Database Systems for Backup and Failover – An Overview” International Journal of Computer and Communication Engineering, Vol. 2, No. 4, July 2013, IEEE.
[2] Whai-En Chen, Li-Yao Tseng, Chien-Lung Chu “An Effective Failure Recovery Mechanism for SIP/IMS Services” Heterogeneous Networking for Quality, Reliability, Security and Robustness (QSHINE), 2015 11th International Conference on, 2015, IEEE.
[3] Yi-Chen Chan, Kuochen Wang, Yi-Huai Hsu “Fast Controller Failover for Multi-domain Software Defined Networks” Networks and Communications (EuCNC), 2015 European Conference on, 2015, IEEE.
[4] Hery Dian Septama, Ardian Ulvan, Gigih Forda Nama, Melvi Ulvan, Robert Bestak, “Dynamic tunnel switching using network functions visualization for HA system failover”, Science in Information Technology, 2015, IEEE.
[5] Güner D. Celik, Long B. Le, Eytan Modiano, “Dynamic Server Allocation Over Time Varying Channels with Switchover Delay”, IEEE Transactions on Information Theory (Volume: 58, Issue: 9, Sept. 2012).
[6] https://en.wikipedia.org/wiki/Failover
[7] http://www.howto-expert.com/how-to-createa-server-failover-solution/
[8] http://www.wethinksolutions.com/sitereplication-and-failover
[9] https://wiki.centos.org/
[10] https://www.howtoforge.com/mysql_master_ master_replication [11] http://clusterlabs.org/doc/ [12] http://www.linux-ha.org/wiki