LOVELY
PROFESSIONAL
UNIVERSITY

An Enhanced approach to SSL Server Load Management based on Server Health

In

TLS
(Transport Layer security)

A Dissertation-II
Submitted

By Dipesh Gupta

To

Department of Computer Science and Eng.
In partial fulfilment of the Requirement for the

Award of the Degree of

Master of Technology in Computer Science

Under the guidance of

Mr. Hardeep Singh
May 2015

LOVELY
PROFESSIONAL
UNIVERSITY

Transforming Education, Transforming India

School of: SCE.

DISSERTATION TOPIC APPROVAL PERFORMA

Name of the Student: Dipesh Gupta	Registration No: 11100574
Batch: 2011	Roll No. A-03
Session: 2014-2015	Parent Section: K2005
Details of Supervisor:	Designation: A.P
Name: Hardeep Singh	Qualification: M.Tech
U.ID: 16869	Research Experience: 2

SPECIALIZATION AREA: (pick from list of provided specialization areas by DAA)

PROPOSED TOPICS:

1. Transport layer Security Protocol
2. Virtual Private Network (VPN)
3. Distributed Denial of Service

Signature of Supervisor 16819

PAC Remarks: quite positive that student will publish paper in good journal.

APPROVAL OF PAC CHAIRPERSON: Signature: 1101 11/10/14 Date:

*Supervisor should finally encircle one topic out of three proposed topics and put up for approval before Project Approval Committee (PAC)

*Original copy of this format after PAC approval will be retained by the student and must be attached in the Project/Dissertation final report.

*One copy to be submitted to Supervisor.

CERTIFICATE

This is to certify that Dipesh Gupta has completed M.Tech dissertation proposal titled An Enhanced approach to SSL Server Load Management based on Server Health under my guidance and supervision. To the best of my knowledge, the present work is the result of his original investigation and study. No part of the dissertation proposal has ever been submitted for any other degree or diploma.

The dissertation proposal is fit for the submission and partial fulfilment of the conditions for the award of M.Tech Computer science & Eng.

Date: ____________________

Signature of Advisor

Name:

UID:

ABSTRACT

In this Dissertation, the work has been done on “An Enhanced approach to SSL Server Load Management based on Server Health”. In this the system is more advanced than existing one by using advanced server algorithms which handles more requests and serves efficiently. In this different configuration, the servers are used to distribute the load among them so they can serve the requests or assign the task on the basis of the nature of the load and provides the security to the connection with maximum performance, so that the clients should access the system smoothly. Our system enhances the performance of the server in case of excess load on the server. By monitoring the individual health parameters of the resources, the system load weights are calculated so that system requests handling capacity is calculated every 1 second. Servers can handle no. of concurrent requests on the basis of the load and weight with different parameters as described in this methodology which can lead a server in better performance and handling the requests more efficiently. Hence results in increasing the throughput of system.

ACKNOWLEDGEMENT

It is true that people and institutions do help in research work, it has been more so in my cases. The words cannot really convey my gratitude to Mr. Hardeep Singh whose guidance and invaluable suggestions has inspired me to work on this research. His scholarly wisdom has inspired us to go into fine details of every aspect of this research.

I will be failing in my duty if I do not express my gratitude to Mr. Hardeep Singh whose precious advices and timely help has enabled me to complete this research proposal in time.

Last but not the least; I owe my gratitude to parents, who have always inspired me during this course of the research.

Date: May 5, 2015

DECLARATION

I hereby declare that the dissertation proposal entitled An Enhanced approach to SSL Server Load Management based on Server Health submitted for the M.Tech Degree is entirely my original work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree or diploma.

Date: May 5, 2015

CONTENTS

TABLES

FIGURES

Chapter 1
INTRODUCTION

In today's world internet has widely made websites popular. The webservers have the huge traffic. The secured web servers are working on the SSL/TLS (Secured socket Layer/Transport Layer Security) which use the encryption algorithms to secure the communication over the internet. The TLS provides the confidentiality, integrity to the data transmitted over the internet. On the web servers, the large traffic over the servers leads to instability or crash of the servers.

On the webserver when the number of concurrent users accessing server simultaneously the load increases, then after a certain limit of requests the performance of the server is affected. When the server's performance is degraded then the throughput of the server is also affected which results in low response time of the server, slow execution of the upcoming request, low response time in handling and executing the requests. Load Management of the web server is required at that instance, which is a crucial requirement in these days. As there are already different techniques has been developed but all of them are not able to handle the server traffic efficiently and maximizing the throughput of the server during high requests rate. The main concept of the Load Management of the server is to provide the improved experience of the working of the server so that client can't realise the slow working of the server. Generally in the classical secured servers which are using the SSL or TLS to encrypt the server that are preconfigured on the basis of statistical information provided in the system.

1.1 Introduction to Transport Layer Security (TLS)

TLS is designed so that it should provide encrypted communications. SSL is the Secured Socket Layer and TLS is Transport Layer Security protocol which is used for the secure communication over internet between client and server. With the TLS users have gain faith in the secure communication channel to transmit the data on Internet. The latest version of SSL is TLS. The TLS works in between the application and transport layer of network. Generally SSL is issued by CA; Certificate Authority. There are number of the CA's that issue the certificate to website or Browsers. The certificates which are issued by the CA are treated as the authentic certificates. Generally the different browsers support the different number of the CA Certificates. So with this these certificates are allowed without any doubt or question. TLS works on the transport layer of network. The SSL works with handshaking with the Server and client.

TLS/SSL (Transport Layer Security and Secure Socket Layer) encrypts the data and transmit the data in secured way. The SSL/TLS have different versions SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1, and TLS 1.2. The TLS is based on the asymmetric cryptography in which public keys are used. The TLS consists of two sub phases; Handshake phase and application phase. The TLS supports both the symmetric and asymmetric cryptography. The symmetric cryptography uses same key for encrypting and decrypting the data. The various symmetric ciphers support by the TLS is AES, 3DES, RC4, DES with different key exchange and Hash algorithms. The asymmetric cryptography includes the different public and private keys for the encryption with complex calculations. These are provided by the CA's (Certificate Authorities)

The key exchange method determines how the key will be exchanges during the session. The cipher suit determines how the encryption and method is used on the application or bulk of data and lastly the MAC is use for to prove data integrity. [10]

1.2 Working of TLS:

The TLS (Transport Layer Security) establishes the secure connection in between the client and the server. The TLS consist of the following phases

- Negotiation Phase
- Data transmission

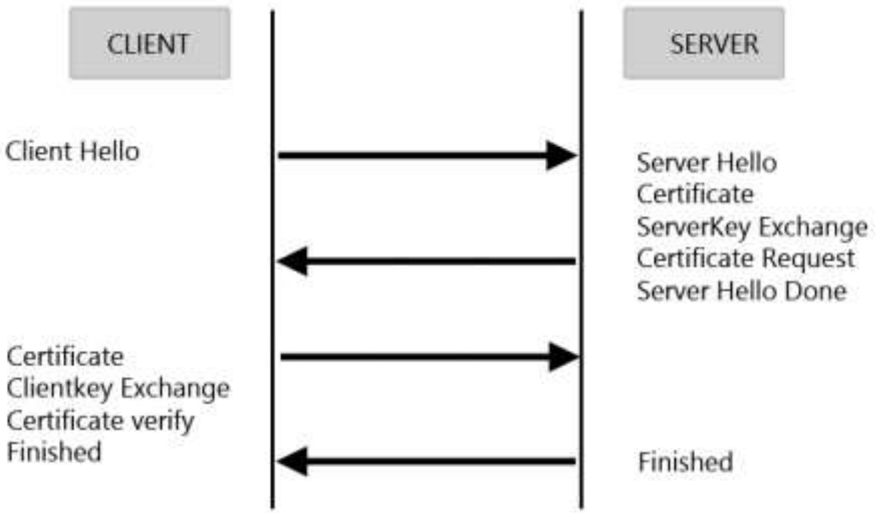


Figure 1 TLS Handshake [1]

In the process for the TLS handshake, the Client sends the *Hello* packet to the server. After that the server responds to that hello message having the protocol information selected and other parameters sent by the client to server. Then corresponding to this, the server sends the session id for that message to resume the handshake process. [1], [2]

The certificate message send by the server depend upon cipher suite. Upon sending the certificate, server sends the *Hello Done Message* which refers that handshake negotiation is completed.

Now the client responds to the *ClientKeyExchange* message which may have the *PreMasterSecret* public key. It is only based on the type of the cipher selected.

Both Server & Client use *PreMasterSecret* & random number to calculate the common secret known as *master secret.* Any key required for the connection is extracted from the master secret which is moved thoroughly *pseudorandom function*.

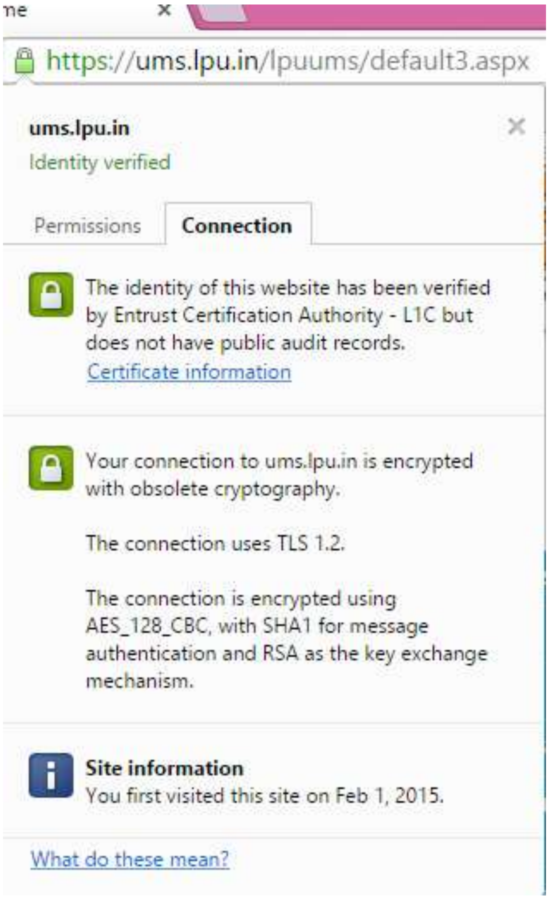
The client will send the *ChangeCipherSpec* that will say that from now onwards everything will be encrypted. Lastly the client sends a message which is encrypted and having the MAC and hash code over the earlier handshake messages.

Now the server will work out to unravel (decrypt) the client’s *FinishedMessage.* If the authentication for the handshake is not done or says its failed then connection is broken down.

Lastly the sever sends the *ChangeCipherSpec* with encryption done (finished or completed) message and the client starts performing the same encryption and verification for the same. [1]

1.3 Trust

Trust is on which users it satisfies on the server to which it starts communicating with any webserver or website. Whenever any website have certificate in which all the encryption parameters and algorithm for encryption and decryption is specified is issued by the CA. The CA ensures user that to which it is going to connect is genuine and there is no security compromise with it.


https://ums.lpu.in/lpuums/default3.aspx

ums.lpu.in

Identity verified

The identity of this website has been verified by Entrust Certification Authority - L1C but does not have public audit records.

Certificate information

Your connection to ums.lpu.in is encrypted with obsolete cryptography.

The connection uses TLS 1.2.

The connection is encrypted using AES_128_CBC, with SHA1 for message authentication and RSA as the key exchange mechanism.

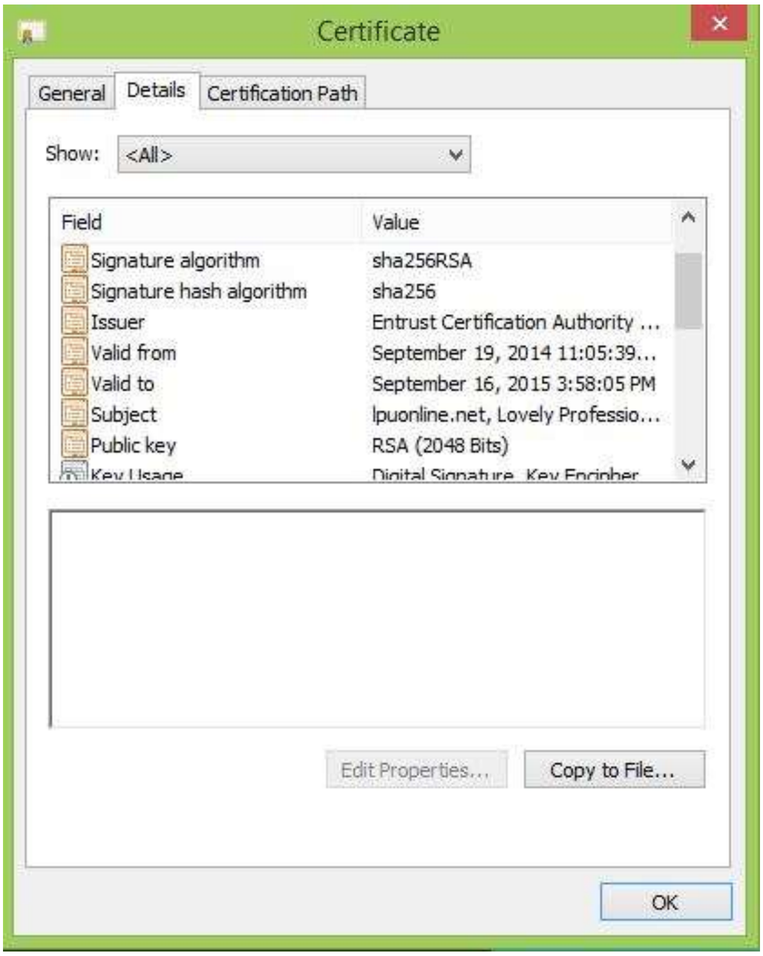
Site information

You first visited this site on Feb 1, 2015.

What do these mean?

Figure 2 Trust Certificate

As in the Figure 2 the snapshot taken from the website https://ums.lpu.in, the university has SSL deployed on website. They have certificate which was issued Entrust Certificate Authority (CA).The CA role is very important for checking the certificate authenticity. If the certificate for any website is issued by any trusted CA then user can access website without any hasslement. In the figure the certificate details are showing that connection is using TLS 1.2 which is successor of SSL. Also it includes the details for the encryption and decryption method for which it is going to start. The encryption and decryption specified in the connection details shows AES_128_CBC which is secured and complex algorithm which results in to negligible chances of MITM. SHA algorithm in the certificate is used for the message integrity. SHA algorithm generates the Hash code for message. The hash message is compared at both sides. When hash comparison is successive then it results that message is not compromised.



Show: <All>

Field	Value
Signature algorithm	sha256RSA
Signature hash algorithm	sha256
Issuer	Entrust Certification Authority ...
Valid from	September 19, 2014 11:05:39...
Valid to	September 16, 2015 3:58:05 PM
Subject	lpuonline.net, Lovely Professio...
Public key	RSA (2048 Bits)
Key Usage	Digital Signature, Key Encipher

Figure 3 Certificate Details

1.4 Typical Load Balance Environment:

In the general load balancer environment there are generally 2-3 servers which are deployed and are connected in network. The servers have some content over it like some websites. These Servers are connected to load balancer through an Internal Network. The LB has two Ethernets Cards E0 and E1.The NIC is responsible for receiving the traffic from outside world or also known as Public IP.

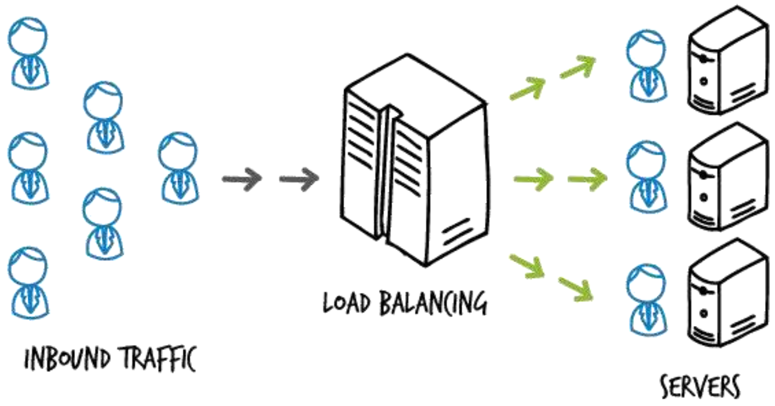


Figure 4 Server Load Balancing

The LB or Load Balancer has algorithms working on it through which it forward the client requests to the server. Assume the E0 is internal Network IP having address 192.168.10.101 and servers are connected in private network with private IP 192.168.10.102 and 192.168.10.103. E1 is having public IP 115.103.15.21 on which clients will communicate.

Whenever a client hits to External IP, the request is received over LB. The LB has request forwarding concept which will move forward. The server on which request is proceeded that will response to the client as per nature of request [3].Authors of [4], [5] & [6] have explained various techniques for server load balancing.

1.5 Why Load Balancing?

In server environment the load balancing is important. The load balancing environment gives the better service and high availability of content. As on single server when the no. of requests are arrived and due to some reason the server fails or goes down then it affects the availability of service. But in the distributed computing upon which if any one of server from the farm fails or goes down then other servers can handle the traffic. If any one of these server fails, another available server starts providing content and user doesn’t know what’s happening behind environment. With this technology the user doesn’t know from which server it is accessing the data or content. The author in [7] have discussed about load balancing scheduling algorithm to improve overall performance

At the behind or at internal network if any server is getting over load then it will go down or it starts discarding the requests. The load distribution depends upon the various factors like algorithm to be used in Dispatcher or Load balancer, also the load balancer have different algorithms on which these are working currently.

Round Robin: In this the LB distributes the equal amount of requests to all webservers. Consider we have scenario of 4 webservers, with 4 webservers the requests are distributed in 1,1,1,1 or 5,5,5,5. It depends on static value which can be assigned in the algorithm. But the major flaw with algorithm is it will equally distribute the request to all servers. If any one of them server is occupied with heavy load then it may crash.

Weighted Round Robin: In the Weighted round robin algorithm, it distributes the no. of request on basis of the weight of the server which represents the capacity of the server for which the no. of requests that would be handled by it. In this the all servers' weights are pre-calculated on basis of the hardware configuration by providing the stress load on it. From that experimental value the weights are generated and are provided in the system that server 1 will handle x no. of requests and server 2 will handle y no. of requests. The major drawback of this system is that if the requested request is of small chunk or heavy nature then it will consume systems resources which will result into uncertainty in forwarding the request to particular system.

Least Connection: In the Least connection scheduling algorithm, the requests are distributed to different servers according to simplified round robin. If there are two servers on the web farm the requests will be forwarded to the system on the basis of the server having least no. of connections. But the drawback with the algorithm is that the system can reach in overload situation whenever there are two similar configuration servers. When the farm has same configuration of servers at that time first server will be preferred for distribution and when the first server is preferred then it will automatically fails one of the server during peak load .

Weighted Least Connection: In this algorithm when there are 2-3 servers of different resource capability it is considered but with this algorithm then no. of active connections and weight of the servers , the ratio is calculated and from that value the data is provided to distribute the requests from the clients to different servers.

Least Connection with Slow Start Time: In the Least connection slow start time scheduling algorithm when an real server is bring online and starts handling the requests.

With this the server having least connections and with slow time are preferred for serving the requests so that the server can get rid of overload situation.

Fixed Weight: In this scheduling algorithm there are multiple servers in the farm. Out of those 1 server is considered as the priority server whose weight is more then all of them. Rest of all servers are of low weight. During the load the requests are forwarded to high weight machine and when the weight of the high weighted machine starts degrading then the low weight servers are initialised .The low weight server's starts serving to clients. The weight for every server is provided based on real servers.

Sticky Session: In this scheduling algorithm the requests having sessions are stickled on the LB; whenever LB forwards the requests to the server the requests with session are stayed with the server and it stay with that server as long as the server doesn't terminate the request or session is not completed/expired. The thing is that user session doesn't need to be created i.e. there is no need of renegotiation until the previous session is not expired. If the session is not restricted with that particular server through which the session was originally created then in that case the server creates another session on another system that would be difficult to handle all the sessions on the different servers for the same client.

Weighted Response: In the weighted response scheduling algorithm is based on weighted round robin. In this health check parameters are used to check the response time of the server .The server which has low response time is considered. Also it is assumed in the health check is that how much time is taken by the machine. Generally the weights of the machine are calculated over every 1 second. The total weight for the servers are calculated on the basis of real severs weight and virtual server. By combing them all the actual weight is calculated. [3], [4]

1.6 Why load balancing is considered?

Load balancer provides a number of features. Some of them are:

- **Priority Activation**: When the load gets increased and number of available servers gest decrease, then standby servers can be brought into picture.
- **HTTP Compression**: It reduces the file content to be transferred by using gzip compression available in browsers. Thus improves the response times.

- **TCP Offload**: This feature uses HTTP/1.1 unite together multiple TCP HTTP requests into one TCP socket to backend servers.
- **TCP Buffering**: Load balancer buffer responses from servers and provides to clients allowing the web servers to free the thread for other tasks.
- **HTTP Caching**: Load balancer stores the static content so that backend servers need not to be contacted.
- **Content Filtering**: Some balancers modify the data on the way through to provide more filtered data to the backend servers.
- **HTTP Security**: Some balancers remove the server identification headers from http requests and encrypt it so that end users cannot change them.
- **Priority Queuing**: This feature gives the priority to different incoming requests to be served by backend servers.
- **Content Aware Switching**: If the incoming requests are encrypted via https, then these requests are decrypted by the load balancers
- **Client Authentication**: It verifies the user against number of authentication sources before giving them access of the website.
- **Firewall**: Direct connection with the backend servers are prohibited with use of firewalls which make decision whether traffic may pass through interface or not.

Chapter 2
TERMINOLOGY

LB (Load Balancer): LB is Load Balancer, which is an intermediate machine placed in between client and server. Load Balancer is the machine or system in which users will request to it on External IP and also it have another Internal Network (NIC 2) through which the servers are connected to it. The LB will receive the requests from the clients and forward the request to the server in the queue. The LB has deployed Windows 2012 Data Center Edition; The LB is having algorithm upon decision taken by the algorithm that the LB will forward the External IP request to Internal Network.

Web Farm: Web Farm is the cluster of servers. There is a cluster of 3 servers with different hardware configuration. The configurations are:

Server 1 is having Configuration Core i7 CPU 2.7 GHz single core, 512 MB RAM, 30 GB HDD and Gigabit Ethernet.

Server 2 is having Configuration Core i7 CPU 2.7 GHz 2 CPU, 2 GB RAM, 30 GB HDD and Gigabit Ethernet.

Server 3 is having Configuration Core i7 CPU 2.7 GHz 4 CPU, 4 GB RAM, 30 GB HDD and Gigabit Ethernet.

Health Parameters: Health Parameters includes the CPU, RAM Usage, Disk Usage, and No. of Active Connections on Ethernet.

CPU Usage: In the CPU usage we'll monitor last CPU usage of every server which will be according to system load. Each server CPU Usage will be monitored and send to LB.

RAM Usage: In the RAM usage last server RAM usage which will be dynamic usage of every server according to system load. The each server RAM Usage will be monitored and send to LB.

Disk Usage: In the Disk usage we'll monitor last disk usage of every server which will be dynamic according to system load. Each server disk usage will be monitored and send to LB.

Active Connections: In the Active connections the monitored last no. of active connections of every server which would vary to system load. Each server active connection value will be monitored and send to LB.

Jmeter: Jmeter is an open source tool for load and stress measure. In this by providing thread value (no. of concurrent users) and Ramp up period and iterations we can generate the virtual users for our testing environment. In this tool we can also generate the table for no. of successive hits and no. of failed hits out of total requests generated by the Jmeter. With this we can also setup different environment like https request or login user by providing credentials. Also Jmeter provides the details for generated request to URL and response generated by the server.

Chapter 3

REVIEW OF LITERATURE

Chunmei Zhang, Junhao Xia, Qun Wang and Xiangwang Chu has proposed a design and implementation of the sever load management which can handle more number of requests .When the large no. of requests are arrived on the system then they are managed by the real time monitoring and status of the dispatcher in the web clusters. As in the general for load balancing, there are two types of algorithm static load balancing and dynamic load balancing. The static load balancing algorithm is for the simple static web pages in which there is no need to check the status of the machine. On the other hand, the dynamic load balancing algorithm like which is based on live load situation of the system. In this system, servers are analysed In this concept, the authors has perform the load balancing strategy with first they use the multiple priority ranking and queue .In this the system task are further split it up into processing task and dynamic real time task. From those the system will setup the priority task and the system will generate the time for the task for the requested task that may exceed the system's ability.

On the other hand the authors also proposed for the dynamic load balancing algorithm based on real load parameters. In this the system load parameters are calculated from the different servers like CPU or RAM etc., these are calculated from ach and are analysed from different servers which helps to reduce the load on each server when the no. of requests arrive in the system. Authors have done the test on the proposed system with different specifications on single node and double node. In this no. of tasks and speed are compared when the no. of total task is increased and parallel assignments are decreased which leads to more system stability and provides the better effectiveness of the system. Lastly based on comparison of data generated by the system and LB approach, the processes or program can achieve high efficiency for better system load calculations. [3]

Zhenbin Yan and Wenzheng Li have proposed for load balancing in the large scale distribution system. The method subnet mask algorithms are implemented by authors to perform the communication in the proxy. When the large no of users send the data the bottleneck problem is controlled by the intermediator.

When the users send request to the server from the web the job is created through the http.in the real environment when the large no. of the jobs are started concurrently and every job starts communicating in between the client and server (control center) and it becomes heavy in load . So according to the system the load must be balanced to avoid the problem, the authors has done work on consistent hashing as the load balancing algorithm to use at layer proxy.

In this hash algorithm authors have used the mathematical function .The hash function will use the different M$, MD5, SHA-1 .These are some encryption algorithms which all users are generally familiar. According to the ideas provided in the paper by authors to set up the map of hash function for proxy and the user; MD-5 algorithm is used to forward the message on the proxy. To use the system, authors use the open source libketama to observe the hashing which is used in traditional C language. The nodes with system are ranging up to average of 160 virtual nodes. The different test is performed on the system with setting up different testing environment. In this they use x86-x64 Special Server of dual core of 2.2 GHz with 16 GB DDR2 RAM,SCSI HDD at 10000RPM,Gigabit Ethernet and tools are used Cent OS, Python as compiler. From the experiment it is observed that set of programs for the tasks and LB/dispatcher in web servers can have a high efficient and effective response. This gives better calculation of resource from the load balancing in the system. [8]

In this they use 3 RS as real servers, 1 LB as load balancer and there is shared storage behind the storage. In these authors has perform the work on the weighted round robin and least weight connection algorithm. In this author has combined the static and dynamic load balancing algorithm by providing the weights to the LB. The weights are frequently arrived on the LB to provide the status of the servers. The servers are categorized into low load and high load category. When there are no. of requests arrived on the server, the LB will subcategorize servers in to 50 % as mentioned above. If the servers have high threshold then they will pause the request otherwise the request is served by the low load servers. The RS will serve the request and mean time they will continuously to send their load situation to the LB. Overall the servers are handling the HTTP requests which will provide better request serving during different peak load of the servers. [8]

Deepak C. Shadrach, Kiran S. Balagani, and Vir V. Phoha have introduced an algorithm that manages the web requests based on all servers’ status. This structure constitutes of two elements, (1) Prober, gathers the load status information of all web servers and (2) Allocator, calculates the weighted matric (CPU load, server response rate and number of requests served by a server). The special aspect of this technique is to choose the server having least value of this metric and to pass the IP of that server to client to further communicate with it. Putting forward an idea to describe Real Time Server statistics based Load Balancing Algorithm (RTSLB) that uses four aspects to redirect the request of an entity to a web server. These are (1) Weighted metric of cache hits on different web servers. (2) CPU Load of web servers. (3) Server Response Rate. (4) Number of requests handled by server.

Our setup is built of 3 Web Servers, 1 PC running a Prober and Allocator (Redirector) and 1 PCs operating as Client which is generating 700 requests on the average using 2 - 10 simultaneous connections. In this, the prober collects the load status information of all servers and sends it as input to allocator. The client sends the http request to allocator and allocator then sends the IP address as output of the server having minimum value of the weighted metric to that client.

The client requests an http request for each web page from a pool of 50 web pages and the result gives the turnaround times for client generated requests ranging from 10 to 700 with the client making 2, 5 and 10 simultaneous connections.

The Random and Round Robin algorithm shows the acceptable level of performance but it lacks the ability to match the variations in traffic on web servers. The CNN algorithm calculates the load distribution based on cache contents of web server, but diminishes after a peak point. So the RTSLB algorithm make over these algorithms because it varies the distribution based on cache contents as well as examining the load involved by the individual servers.[9]

Carlos Oliveira has presented a virtualized architecture consisting of technique that can be used to allocate resources to multiplier applications which comprise of applications and database tiers which are implemented by multiple configurable virtualized server machines. The architecture accommodate a layer of proxies is placed between client and server to grant the load balancing in different tiers. He also performed an experiment in which he showed CPU resources in a batch of physical systems being allocated and DE allocated as per the client requests. He used RuBiS benchmark tool to simulate the workload on multi-tier application i.e. eBay site by adjusting its client emulator.

Their multi-tier architecture uses 2 technologies: (1) Apache Proxy and (2) MySQL Proxy. In this, each tier may comprise the usage of multiple servers. The requests generated by clients are processed by application proxy running in front end machine and redirected to application server which runs on top of a VM consisting a replica of a RuBiS PHP version. These requests are then sent to database servers through query languages. The application communicates with database server through MySQL proxy. Each database server comprises of a replica of MySQL database that contains the data of website. The front end also runs 2 modules (1) monitor that continuously manages the CPU utilization of physical and virtual machines. (2) Manager that lists the information of CPU utilization of both physical and virtual machines. The manager also uses weights applicable at the Apache proxy. The requests are managed by master and slave databases.

The objective of this experiment was to maintain the CPU utilization under a given edge. An essential report is presented in this experiment. When CPU utilization is low, the response time is low. When CPU utilization is high, response time goes to 400%. Thus to maintain the response time, the VM replication was performed to bring the CPU utilization to 300%. This leaves the 100% CPU utilization available to be used by VM management domain. Thus in this experiment, CPU utilization was considered as the main metric for VM replications.

The main objective of the techniques used in this experiment was to prevent the attempt of modifying application's code and required no knowledge of the previous work. Also, the MySQL proxy was also used for load balancing between replicated databases. [10]

Jorg Jung, Simon Kiertscher, Sebastian Menski and Bettina Schnor proposed an architecture for credit based server load balancing (SLB) for DNS. The advantage of this technique is that the load balancer can simply accommodate dissimilar load requests and back end server capacities. It presented a way to define the credits for UDP based services. Dispatcher based server load balancing (SLB) is an adequate method to give flexible and reliable services. In SLB, the backend servers report a metric called *credit* to LB. It represents a ratio of current metric values to number of requests. The Self Adapting

Load Balancing Network (salbnet) is based on Linux Kernel and comprises of three elements (1) salbd, (2) LVS Scheduler and (3) Libnethook.

A metric is needed to follow the current load on the backend servers for addressing the credits. A filled UDP queue receive queue signifies that the server will not reply to requests anymore. UDP receive queue hold the packets of distinct sizes. The disadvantage of UDP receive queue is that packet size is not known for the backend server application. Therefore, they used old values to conclude the expected packet size. They used the median *pmedian* to determine the packet sizes. They also used the number of dropped packets *pdrop*. If pdop>0, the server is overloaded. The credit based SLB algorithm uses 2 types of credits (1) Hard Credits and (2) Soft Credits. They can be calculated as following

A dispatcher based SLB scenario is built up to calculate the Self Adapting Load Balancing Network DNS extension. This set is two armed, NAT based and uses route path. The LVS is used as load balancer with the salbnet reporting algorithm dynamic pressure relive (DPR) and the salbnet scheduling algorithm.

The current credit calculation for UDP is based on the expected estimated UDP packet size of client requests and the current approach was executed in within the load balancer LVS. Within the calculations of salbnet and DNS scenario servload processed more than 36 million requests and up to 45 thousand requests per second in one pass. This defines the scalability of both, the servload benchmark which produced the request on one single machine and the salbnet module which executed the credit based SLA. Their mock up implementation is a confirmation of concept execution and based strongly on software versions. In future, newer kernel versions may provide distinct and quick options to access the UDP receive queue and its mapping determination. [11]

Jorg Zinke, Bettina Schnor have performed the experiments on the website that used the data statistical from internet. They performed the experiment to analyze the two algorithms Weighted Least Connection and Weighted Round Robin to check the impact and response of the server during high and low load. The web servers are provided 96 values at high extent to check the status. Upon performing the different test, results were generated from the data. It is observed that when there is least load on server the weighted least connection algorithm performs better than the weighted round robin as the load on the server gets increased. During high load the weighted round robin algorithm performs the better and provides the efficient result. [12]

Rajesh Kumar Pateriya the author has described the load management of server. Generally when there are numbers of SSL requests arrived on server, then server gets unstable and leads to lower the performance of the encryption of the data and loose in the throughput of the system. In this with Adaptive SSL the load balancing is performed with using the adaptive SSL algorithm. .In this the Algorithm there are different phases that are specified as the Request Counter, Max, and Cipher.

Request_Counter: The Request Counter in the Algorithm phase is which stores the no. of the concurrent requests arrived on the server.

Max: Max is the value which has the specific limit before over loading.

Cipher: Cipher is having two encryption algorithm RC4 and 3DES. In this when there are requests are up to 140 then the server is using the 3DES encryption method which is slower but effective strong cryptographic algorithm. And when the requests are up to 240 then it starts using the RC4 which is faster and less secure then 3DES.

The adaptive SSL is implemented in this paper which adapts the encryption algorithm according to the load of the requests on the web server.

In this a web application is deployed on "*tomcat"* a Linux based server, an application has book-store functionality. The multiple concurrent requests are generated in the server by the user thread in the OS and requests are sent to the server. The server is handling the requests up to 140 that is the normal situation before overloading and using the 3DES algorithm for encryption which is better secure algorithm and slower in processing.

But as the requests keeps on increasing or the system is in the over load then the algorithm will switch to RC4. RC4 encryption provides lesser security but it has a feature of high performance. RC4 encryption is 16 times faster than 3DES algorithm. Upon experiment the results show that at overload condition, RC4 maintains the throughput of the system. [13]

Christaiaan J. Lamprecht & P.A van Moorsel has discussed about the self-adapting security in the server. In this paper the performance cost of the security adaptation under different load conditions are analyzed so that better adaptation can be adapt. The adaptation is based in the ASSL (Adaptive Secure socket Layer). In this first the SSL negotiation phase is analyzed that when a client creates a session with the server then every time when it connects, it uses that existing session if it's not expired so the request parameters are re-used. The author has compared some cryptographic algorithms on the basis of their throughput in bytes /second.

Author has performed the experiment on the system with Pentium III CPU with 512 MB RAM and develops its own tool to create connection with server from client at different rates which allow the customization to multiple requests from a client at specific time.

The 24 clients start connecting with the server for 10 seconds resulting into the 240 clients/second to serve by the server. The algorithm use at that instant is 3DES that can serve up to 290 request/second before reaching the overload .The request are send it by different file size with different client rate in different time aspects. The different size available with that particular tool is 1024, 8192 and 12288 bytes.

Table 1: Average No. of Bytes Processed by Different Algorithms [14]

Cryptographic Algorithm	Throughput in kB/Sec
RC4	112000
AES 128	36000
AES 192	30600
AES 256	26500
DES	19300
3DES	6900

The DES, 3DES and RC4 server throughputs are compared and the cryptographic algorithms are used in the server for the Adaptive SSL (ASSL).

In this the server is overloaded when the request number is 230 with 3DES and when the no. is 296 with RC4 and DES.

The Adaptive SSL Mechanism shows the server's maximum throughput using this policy and with available resources. Therefore the ASSL can be best suitable when there are no. of clients who need to download the files over the secure server and viewing the content on the server like email application attachment etc. [14]

OP Verma, Ritu Aggarwal, Dhiraj Dafouti, Shobha Tyagi these authors has compared the different algorithms with different key size and compute the performance and speed of the algorithms with different factors. The different algorithms are DES, 3DES, Blowfish, AES. All algorithm are provided with different key size and different data size to analyse the impact on performance with different data input and key size.

The overall comparison is done over the block cipher (block of group of bits) and stream cipher (plaintext digit in encrypted). In the block cipher the encryption is slow than the stream cipher. Because in the Block Cipher as the Block size will increase, the encryption process will be slower and as the block size is decreased, the encryption process will be fast and size of the input data will be more encrypted very efficiently.

Also the same way the larger the key size in the stream cipher as like the block cipher it will slows down the encryption , because all bits will be involved in the process. As the key size is small then the no. of execution cycles will be reduced. The author has performed on the experiments on the system Pentium 4 2.4 GHz Machine with different data input size ranging from 20527 to 232,398 bytes. The experimental results are below in ECB (Electronic code Book) and CBC (Cipher Block Chaining) Mode.

From the experiment performed it is observed that Blowfish takes less average time after that DES, AES and 3DES comes. Blowfish has the problem of the Weak Key problem Also, the blowfish can’t be implemented in the SSL/TLS. The problem with DES is it has the small key size which can be cracked easily. Out of those AES can be consider as the best because with different key size the AES is performing better than others and it is also much secured and has much less flaws than any other encryption algorithm.[15]

C.J. Lamprecht & Aad P.A. van Moorsel has discussed about the adaptive policy on the SSL webserver when there is an overload situation on the server. In this the SSL has the modules enc, req, resp and conf. that is encrypted, request, response and configuration respectively. The adaptive SSL is implemented on the Apache webserver with some integration into OpenSSL. The Hooks, Request filter and Handler are parts of it. In the Hooks ASSL registers with apache. With Request Filter the client renegotiates on the basis of nature requested by the client.

Various experiments are performed with different scenarios i.e. in first with ASSL, the client creates minimum new SSL connections with the server. The client requests the

1000 requests per session and establishes the load on the server. The main scenario developed with this if the apache server is consuming more time in renegotiation instead of negotiation, the apache server experiences maximum load between 30% to 85 %, the maximum requests can be served and supported by SSL, also renegotiation will be tough one and new client request number is decreased.

In another scenario, the SSL performance is measured on the basis of the when the new client creates the number of the connections with the server in new session. In this SSL has to work more instead of ASSL. The new connections are needed to be created. In this graph figure of “Performance Graph for one request per SSL session “ shows the experimental results that when the server reaches the maximum load, the performance starts degrading. The reason behind is that the server apache which acquires the resource during utilization doesn’t release it.

On the overall experimental results the summarized result from the end of author is that the load on the server is dependent on the client behaviour which includes the processing time used to execute the request. [12]

V.M Suresh, D. Karthikeswaran, V.M. Sudha, D.Murali Chandraseker has done work on the web server load balancing methodology that has been implemented by the authors. In this, the web server having the SSL connection the load requests are balanced by using the balancer at the back end of the server. In this balancing occurs at the server end via NIC (Network Interface Card). The SSL with Backend forwarding technique is used in the webserver to improve the load balancing in the clusters of the servers. In this technique authors has overcome the existing problems of latency and throughput of the server by 40 % as results are provided within the papers. The throughput is overall rate of serving requests. In this technique the 3DES and RC4 algorithm is used for the encryption of the message. In this the load balancing is done with the SSL with session and SSL with the Backend. In the SSL with Session is done when the client establishes the connection with the server and the server will check whether the client is the frequent one or not. If the client is frequent one and it is consuming more resources than normal one in that situation the server will require more computations on that request will not be forwarded to the lightly loaded server.

In the SSL with backend forwarding technique the load balancer is deployed at the distributor to get the load information for the server. In this the variables i, Li denotes the number of server and load on the i^{th} server respectively. T1 and T2 are the two threshold values of the server. The Li , T1 and T2 compare the load on the server with threshold values generated by the load balancer. According to the nature of threshold of the two servers the request is forwarded to the server. In this the server has the two modules at the back end. First is Lightly Loaded Application and another is Heavy Loaded Application on server. Mostly the dynamic requests on the server are served on the heavy loaded application part. Both the heavy and lightly loaded server modules are connected with the NIC (Network Interface Card) at the backend.

With this the technique authors observed that 40 % latency is improved and throughput of the server also gets better. [13]

Ryosuke HATSUGA, Takamichi SAITO has introduced a new concept of secure web communication through SSL. Load balancing for SSL web system is famous to apply. He proposed a new load balancing cluster system for SSL web system by considering the existing ones that can dynamically migrate the SSL session from one system to another server so that SSL servers can use an existing SSL session that other SSL servers has started.

In the approach of use of LB StateLess, the load balancing decision is based on executing states of backend SSL servers and HTTP request are distributed among them. Therefore in this approach, it is difficult to keep on using one SSL session established before for a number of HTTP requests and thus make use of more SSL sessions among peers.

His proposed system was constructed of following elements (1) RAgent, (2) LB and (3) SSLAgent. RAgent is a dispatcher between client and SSLAgent. Its main objective is the TCP port forwarding based on SSL type. LB is a TMX3000 load balancer adopted in this system. SSLAgent achieves SSL session migration by exchanging MSG.SSL.OBJ with other SSL agents. This system supports two operations modes (1) CSSL-SF is used to ensure the persistence between peers and (2) CSSL-SL is utilized for a case server side needn’t ensure persistence.

This system is evaluated based on comparing the existing load balancing SSL clusters. (1) Network Environment in which all hosts are equipped with 1000 Mbps NIC and connected to each other by a 1000 Mbps layer 2 switches. A round trip delay of 30 msec is inserted between client and server side. (2) Client in which an HTTPS load generator is running on host equipped with Pentium 4 and 312 MB of RAM. (3) Server-Side- There is 3 types of SSL web systems to compare the performance. (a.) Load Balancer and Apache mode_ssl (b) Pound and Apache (c) C-SSL

2 Scenarios are prepared to evaluate the performance of the proposed system.

Scenario A: It is an experimental system to evaluate the performance of LB-IP, Pound-IP and CSSL-SF that provides the persistence between peers.

Scenario B: It is an experimental scenario without persistence between peers to evaluate the performance of LB-RR, Pound-RR and CSSL-SL.

The performance of this system got improved by migrating the SSL session from one to other servers that can efficiently use the one existing SSL session among them to minimize SSL setups.[14]

Suresh V. Limkar, Rakesh Kumar Jha, Shilpa Pimpalkar, Santosh Darade, authors has proposed a SSL-VPN technique based on graphical encryption named as geo-encryption. Author used graphical location parameters obtained by a GPS Module to provide better security to mainstream SSLVPN. The proposal is based on geo-encryption. The proposed system is a kind of access control system which can allow to access the confidential information only from trusted locations

The system contains two phases: one is registration phase where the mobile device is registered on the server and the trusted location use that device is saved on the authentication server. In the other phase the client use the device by successfully login into the device.

According to the authors this scheme can be used at very highly security area like military to perform crucial operations from only trusted locations e.g. firing a missile only from base camp and etc. This system can be more advanced and it can be implemented or embedded anywhere in specific domain according to the requirement as like Banks, Managers of organizations to monitor their employees in real time. [15].

Chen Fei , Wu Kehe has proposed a method of VPN gateway based on SSL protocol. It is used to establish virtual private network and works on Secure Socket Layer. It uses series of cryptographic techniques, symmetric and asymmetric encryptions, digital signatures and certificates as well as message digest algorithm.

SSL protocol uses TCP protocol to transmit data. SSL VPN is based on layer 7 application. It provides a number of secure solutions like network logic isolation, server isolation protection, strong authentication applications etc.

Implementation steps are as follows

a) SSL VPN imports a trustworthy third party root and server certificates. Client downloads these certificates and stores them in to physical certificate.
b) Set access to back end application server
c) SSL VPN gateway should be open to ensure that client software is running and it establishes a communication with server. Client authenticates SSL VPN gateway based on certificates and server authenticates client through certificates and password.
d) Finally establish a secure encryption tunnel between SSL VPN gateway server and client.

SSL VPN application provides a virtual URL address to the users outside network. The connection started from client is obtained by SSL when users want to access the intranet network. Then it is mapped to different application servers after authentication. SSL VPN must transfer the content from internal FTP server to HTTPS protocol and html format and then finally transfer the content to clients.

SSL VPN is widely used in all aspects of public remote access due to its technical features and simple configuration. It provides a secure remote access platform for its users [16] [21].

Chapter 4

RATIONALE & SCOPE OF STUDY

In this system the servers will be able to handle the number of TLS based requests. The main aim for the study is for the advancement in the servers so that they can avoid existing issues and manage overload situations.

As in the existing system servers can’t be able to handle the TLS requests which are most crucial in these days. In today era the servers which are used by the systems can handle millions of requests by the system in distributed computing using the load balancer. We’ll use the Load Management technique on the servers with which we will use the hybrid and modified weighted round robin algorithm for server load distribution .So that each server can handle the requests and avoid the overload situation. Also it can handle pending requests without discarding most of them.

The pending request in waiting must be in waiting pool for some waiting interval and wait for previous requests to be executed by the server. If in the mean-time the request is not entertained by the server due to non-vacant slot, only in that case the request must be discarded.

Chapter 5
OBJECTIVE OF STUDY

In this work, the Web Farm having multiple servers which are deployed over the network, with our algorithm and concept the of TLS (HTTPS) user requests are distributed to different servers through which load is efficiently distributed with proper encryption and no security is compromised.

- The Web Farm server's health parameters are monitored continuously to check the load situation of every individual server. The health parameters will include CPU, Disk, RAM and No. of active connections usage.
- From the load situation of every server the weights are provided for every individual server so that it can handle that no. of request.
- The health parameters are analysed after 1 second through which the weights will be obtained.
- We are using Improved Weight Distribution Algorithm on the LB to distribute the load on all servers.
- The load is distributed according to live performance and load on server.
- The requests are served on the priority of servers having least load. The system which has least load that will have high weight for request handling.
- In this work we are using the multiple servers which will serves TLS (HTTPS) requests among all servers.
- In our work the TLS is deployed on Web server. With the SSL/TLS many combination algorithms are preferred for better encryption.
- The different configuration servers are used for distributing the load.
- Different servers lead to better availability of service from different services.
- The main idea behind the concept is that our Web Farm is handling '*n*' requests concurrent and when clients are trying to access the system then there will be better security for the user to avoid the eavesdropping of data.
- Discarding of the requests will be less in comparison to other techniques. With enhanced distribution technique load is distributed among servers.
- The upcoming requests will reside in queue and will be served accordingly. If the

request is not served within 30 seconds in that case it will be discarded.

- To avoid overloading situation CPU & RAM threshold is fixed at 95 %. Whenever any server’s CPU or RAM reaches the threshold it will stop accepting further requests.
- If any one of the server gets down failed or overloaded then a log will be created for individual server detail with specific reason.

Chapter 6

RESEARCH METHODOLOGY

6.1 Methodology

In our work the existing system is more advanced, improved and more secured with usage of TLS; TLS is Transport Layer Security which is enhanced, better and secured version for providing the security. Our Web Farm will consist of TLS deployed which will provide secure communication to its clients. The system will have different servers on the Web Farm so that system can handle the overload circumstance and distributes the load among all the servers.

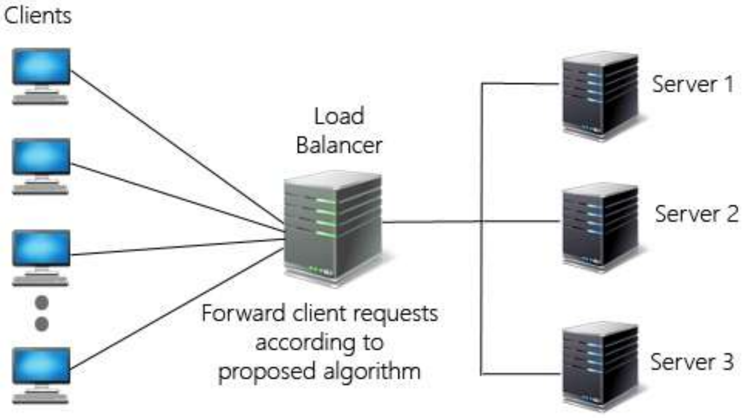


Figure 5: Proposed Model for Management

In our methodology there are multiple web servers in the Web Farm on which various clients are communicating through encrypted channel. We have 3 different servers with different hardware configuration on which the website is deployed. Each web server is configured with Windows 2012 Data Center Edition for hosting website. The Servers are connected to LB which is also as a machine in Internal Network. The LB will have two NIC which is dealing with two networks simultaneously. One NIC is handling the Internal Network (192.168.112.1) which is connected with the servers and another NIC is connected to deal with the clients who are on LAN/WAN (192.168.10.1).

The CPU, RAM and Disk Impact factors are used by observing the roles of each resource on different multiple servers that how these different resources behave during the different load situations like low load, high load or peak load. So by observing the roles of resource the ration values are provided and are used in our methodology.

Our LB is performing all tasks On LB, the health parameters for all the servers are arriving frequently. The Health Parameters include the CPU Usage, RAM Usage, Disk Usage and No. of active connections on the NIC for every server. Based on the usage of these resources of every system, the Machine Load or Weight is calculated for every system with the expression below:

System load= (load of CPU+ load on disk+ load of ram)

∑Wload = Wc+Wd+Wr

Each individual load is calculated by:

Load of resource =last usage x impact factor

Wc = lastUcpu x Ic

Wd = lastUdisk x Id

Wr = lastUram x Ir

Ucpu is the last usage of the CPU, Udisk is the activity of the disk in % and Uram is last usage of the RAM on server.

Ic represents the Impact factor for the CPU, Id for disk, Ir for Ram and lastly Ic for no. of connections active on NIC. For each resource we have provided the impact factor depending upon the importance in using and role during working of the server .Here we have provided the 0.5 for CPU, 0.4 for RAM and 0.3 for Disk.

After Every 1 Seconds the health parameter are observed by LB and for every resource last utilization is used and is multiplied with their impact factor. From that value the each Wcpu, Wdisk, Wram are obtained and that values are simulated. From that the whole load on the server is calculated. In this way the load of every server is calculated.

After getting all the Wload value for all the servers the server which is having the least load and having higher capacity to serve the request is prioritized by using the improved weight distribution algorithm. In improved weight distribution algorithm the algorithm will forward the requests to servers in simultaneous order and the health parameters are updated and monitored every after 1 seconds. So that all servers get the distributed load rather than only some. The algorithm will perform request distribution on different servers' e.g (9, 5, and 2) or (6, 4, 1) or (5, 3, 2).The server will have maximum capacity for handling the requests will handle the maximum requests at that time. The weight distribution algorithm will performs better and provide better prob. to servers for better load handling. In case if any one of them server goes in high request rate then its weight for holding requests will be decreased. On the LB weights of different servers of having different load ae fetched and the load ratio of each server is calculated by the expression. From the load of the each server we'll calculate the capacity of each server by using the dynamic values provided by the system.

Each server capacity will be calculated with ratio like 1:2:4. The smallest part is 1 and capacity is calculated with the value of x: y: z

Consider x=10 then y will have the value 20 and z will have 40. The lowest value is the capacity for the high load server and 2nd is for middle one and 3rd value will correspond to high capacity server.

With our methodology different configuration of servers are also compatible. Because every system will have different configuration and different configuration give rise to different load handling and different resource capacity to provide the output. The algorithm will keep going on to calculate weight of every server after 1 second. After calculation if weights are changed then request tasks are forwarded according to updated weights otherwise the tasks will be forwarded by the previous weights which are obtained by calculation performed in algorithm. So with load distributing the chances of getting a server due to high load are very less i.e negligible.

In this we have a request queue on LB on which the requests will arrive. The requests will start distribution on the basis of the server current load. The queue will hold the all requests that are coming to the LB. In the meantime queue will starts de-queue on basis of server load. If the request which arrived on queue but not entertained by the system in 30 seconds then in that case it will be discarded.

Also in our system there will be log feature on the LB. The automatic log will be created on the LB when there is some activity held like server reaches its threshold value means the CPU or RAM is at maximum usage or whenever any server goes down due to any reason a log will be generated with reason at time stamp. If the server reaches its threshold then will generate status as overload and if server fails or loses its connectivity with the LB then Inactive status is generated in the log to determine the reason.

In our methodology the all the servers are managed with threshold value of 95 % CPU and RAM. Whenever any server’s CPU or RAM reaches its threshold value in that case the server stop accepting the further requests from queue. The server will process only that requests that are present in system or system started serving them. Whenever the CPU or RAM % starts degrades beyond its threshold only in that case server resumes taking further requests; only in that case if algorithm allows system to forward it by calculating the load of the server.

In our methodology when no. of users try to connect with the LB from external network then the requests are monitored and analysed by the weighted round robin and health parameters which calculates the weight of every server. The maximum the load on the server the server will have least weight to server the requests. As any server will have less load value then its load capacity is high that it can handle more no. of requests. The LB will forward the request from External IP Network to Internal IP Network of the server to which request should be served according to decision taken by the algorithm. Every request on the server is using the encrypted tunnel https (TLS) and then they will send request to the server. On the server there is the Content generator that is holding some application where the users will finally access.

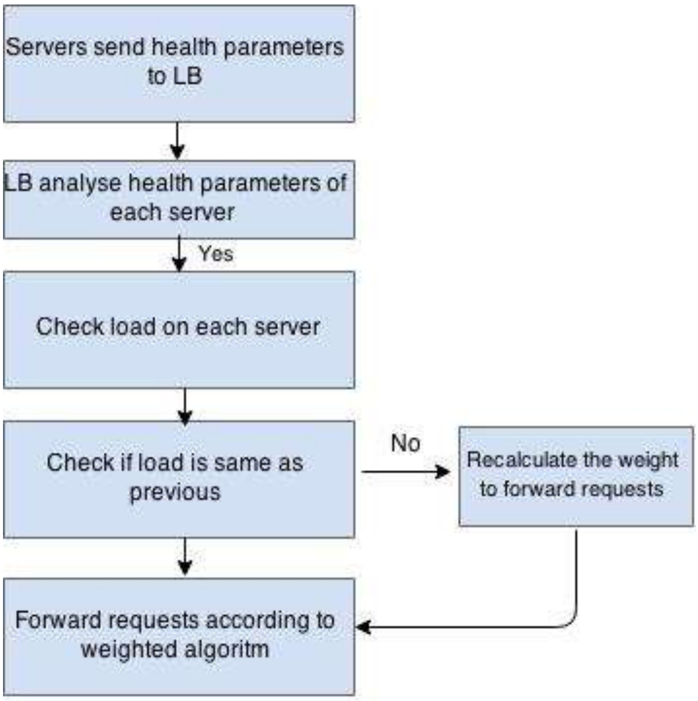

Servers send health parameters to LB
LB analyse health parameters of each server
Yes
Check load on each server
Check if load is same as previous
No
Recalculate the weight to forward requests
Forward requests according to weighted algoritm

Figure 6: Flow chart for working of system

The Basic algorithm for working of the concept is that servers are initialised and they will start their resource usage to the LB to which it is connected. The different servers will have different load status because of different hardware configuration. We use the max CPU & RAM threshold to 95%, 5 % CPU usage is reserved for the other OS activities which will perform some internal activities like processing threads of the Operating system. If we try to utilize the full system usage then in that case the CPU or RAM will gets busy or gets unstable when it usage is 100 % on long duration. So to avoid that such kind of situation, the CPU usage is reserved. In this if the CPU is reached at 100 % then the system may get stable when load of systems gets decreased but in the same way if RAM usage gets exceeded to 100 %, then in that case the system may crash which leads to wastage of time to restore in stable state.

Algorithmic Steps:

1. Set the individual server CPU and RAM threshold max. Usage up to 95 %.
2. Servers will send health parameters to LB continuously.
3. LB will monitor the resource usage and calculate the load weight of every server after 1 second for all servers.
4. The server which have least load weight; have high load capacity to handle the

requests.

5. The LB will forward request to all servers having high capacity to low capacity to distribute using improved weight distribution algorithm.
6. After again 1 second, the server will again check the usage of the servers and calculate the load for all servers to assign the task.

6.2 Pseudo code for Algorithm

```
If (totalRequests > requestsServed)
            then
 while (requestsServed < (requestCapacity < totalRequests ? requestCapacity :
totalRequests))
              {
                ClientRequests request = (ClientRequests)RequestQueue.Dequeue();

                DistributeMessage(request.buffer, request.clientSocket;

                requestsServed++;
              }
              totalRequests = totalRequests - requestsServed;
              requestsServed = 0;
```

6.3 Tools for Implementation

The Load Management with TLS is implemented in Windows 8.1, .Net Platform with using Visual Studio 2012 Ultimate. The C# and ASP.Net is used for programming the server- client application with .Net Framework 4.5. The Hardware used in the development process and for running server application is Intel Core i7 3740 QM 2.7 GHz, 16 GB RAM and with Gigabit Ethernet.

6.4 Interface of Load Balancer

In the Load Balancer which is designed in c# windows form. We need to specify the IP’s for the servers to being connect. We can enter multiple IP’s of the server in LB to map them. After providing the IP to the LB by clicking the Connect button the LB starts operation.LB has two networks one public network to receive the requests from the client/users and another network to which servers are connected and request will be forwarded. When a client hits public IP of the LB it will create connection with the LB and LB will forward the request to server according to our algorithm which is deployed in LB.

On the LB we have a grid which is displaying the CPU, RAM, Disk and Load weight of the individual server. These values are getting updated after every 1 second and these are calculated on the basis on resource usage of each server. The server which has low loaded that will be used for serving request on the priority.

In the Interface of the LB we’ll have column which is showing loaded weight of the server which is generated upon calculation of parameters. The next columns have the Ratio of different servers according to their load. It will provide the value for different servers upon which their capacity for load should be recognised. In the last column of the grid it is showing the total requests allocated to particular server allocated.

Chapter 7
RESULTS & DISCUSSION

After successfully implementing our proposed system on server, servers are able to handle no. of concurrent requests on the basis of the load on the different servers in the Web Farm.

If the LB gets more no. of concurrent requests at that instance, system will distribute the no. of requests to servers in the Web Farm. Secured connection is considered so that the system can entertain concurrent requests without compromising the security. The data is secure and performance of the server will be better. In this situation the system will easily handle all the requests.

The whole performance of the servers are dependent on various factors like server hardware configuration, CPU Cycles ,RAM Usage ,Disk Usage, no. of requests i.e. either static web or dynamic. In this the values for each parameter will only depend upon the load of servers.

In our work the servers can handle more pending requests in the queue so that there is less discard of pending requests. The request have to wait for time *t* in the queue, in the meantime if they are not served then the system will reject the pending requests. The requests from the queue will be served on the basis of the requests completed or the no. of users leaves webpage. If the users are not entertained in that time then in that case all remaining requests will be in waiting queue for 30 seconds.

7.1 EXPERIMENTAL WORK

The whole environment is hosted on VMware Workstation 11. For experimental environment we have setup the 3 x IIS Web Servers with Windows 2012R2 as OS. The details are as follows:

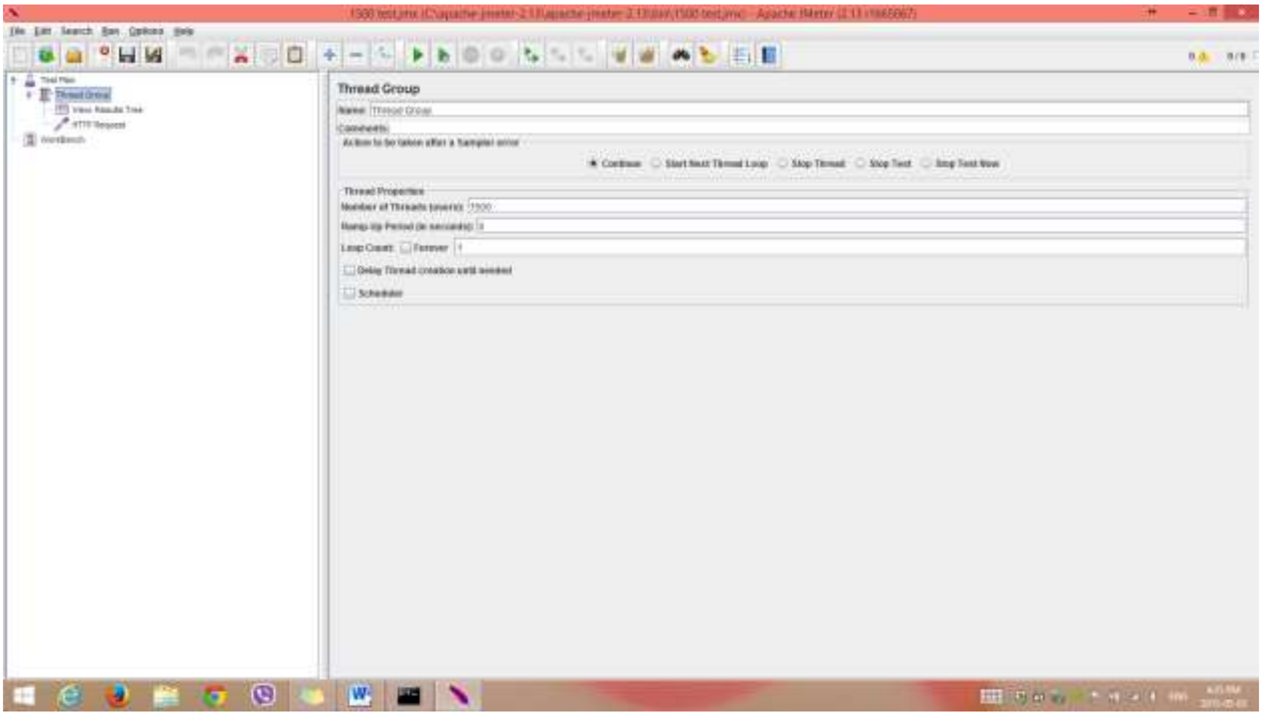
1. Web Server 1
    a. OS: Windows Server 2012 R2 Data Center
    b. CPU – Intel Core i7 2.7 GHz , Single Core
    c. RAM – 512MB
2. Web Server 2
    a. OS: Windows Server 2012 R2 Data Center
    b. CPU – Intel Core i7 2.7 GHz, Dual Core
    c. RAM – 2 GB
3. Web Server 3
    a. OS: Windows Server 2012 R2 Data Center
    b. CPU – Intel Core i7 2.7 GHz, Quad Core
    c. RAM – 4 GB

We have also setup a custom Load Balancer which is running a custom designed load balancing algorithm based on system weighted enhanced Round robin method. The algorithm functionality and how the clients are connecting to the web servers are developed in C#.net platform. The details of the Load Balancer machine are as follows:

1. Custom Load Balancer
    a. OS: Windows Server 2012 R2 Data Center
    b. CPU – Intel Core i7 2.7 GHz , Dual Core
    c. RAM – 2 GB

On the LB we need to enter the IP’s of the Web Servers to which it is required to connect to serve the client requests. We then click on connect button on the application interface. After few seconds the LB starts initialise it and ready for operations of distribution load.

We have simulated the clients on Jmeter Load testing tool. Jmeter simulates concurrent multiple users accessing the site hosted in our environment. We have prepared 3 test scenarios where we have requests coming from 500, 1000 and 2000 concurrent users with different ramp up time and different loops to put load on the site and the web servers hosting that site.



1500 test.jmx (C:\apache-jmeter-2.13\apache-jmeter-2.13\bin\1500 test.jmx) - Apache JMeter (2.13 r1665067)
File Edit Search Run Options Help
Test Plan
Thread Group
View Results Tree
HTTP Request
WorkBench

Thread Group

Name: Thread Group

Comments:

Action to be taken after a Sampler error

● Continue ○ Start Next Thread Loop ○ Stop Thread ○ Stop Test ○ Stop Test Now

Thread Properties

Number of Threads (users): 1500

Ramp-Up Period (in seconds): 3

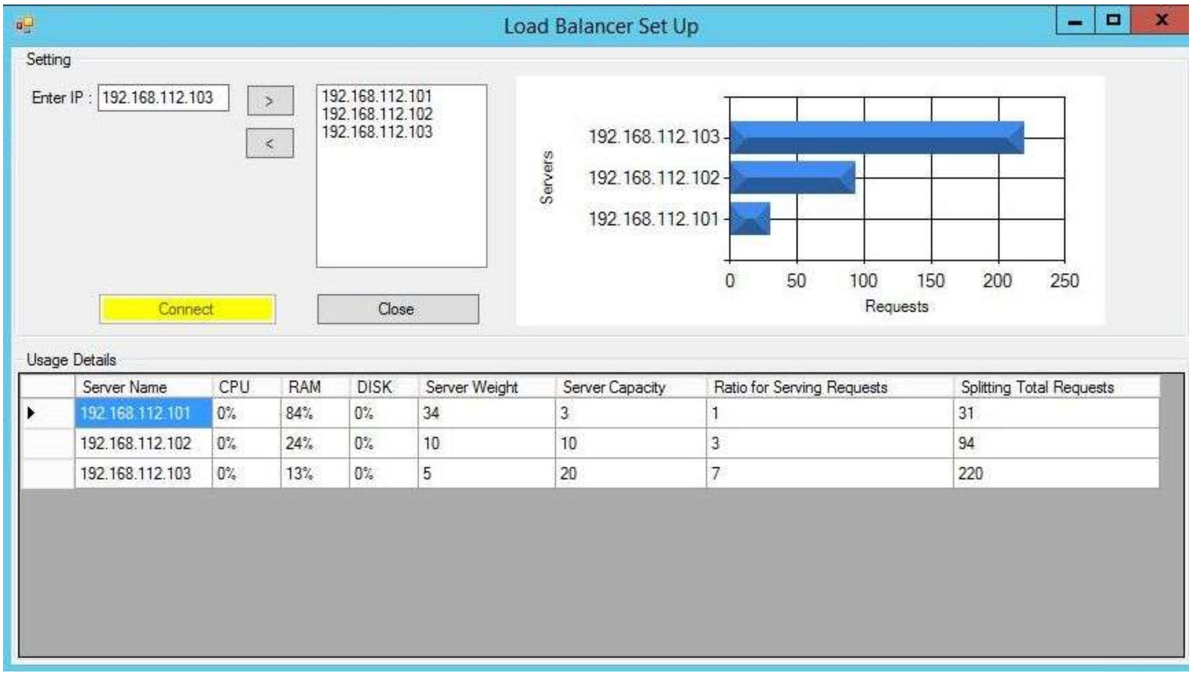
Loop Count: □ Forever 1

□ Delay Thread creation until needed

□ Scheduler

Figure 7 Jmeter Testing Tool

On the LB we have provided the IP’s for different servers which are connected with the LB. Currently we have 3 nodes setup for testing the environment. It can be expandable in future according to requirement. The Custom LB application is supported for n no. of servers to connect with it.



Enter IP : 192.168.112.103

192.168.112.101
192.168.112.102
192.168.112.103

	Server Name	CPU	RAM	DISK	Server Weight	Server Capacity	Ratio for Serving Requests	Splitting Total Requests
▶	192.168.112.101	0%	84%	0%	34	3	1	31
	192.168.112.102	0%	24%	0%	10	10	3	94
	192.168.112.103	0%	13%	0%	5	20	7	220

Figure 8 Screenshot of LB

The above screenshot of the LB represents the usage of resources of different servers. It also represents the server weight which is obtained by calculating the values of load on each web server at the back end. From the server load and weight, we are calculating server load handling capacity for each web server to serve the clients. Then we calculate the capacity ratio of each web server to serve requests. Finally the total requests to be handled by the servers will be split as per this ratio and the requests are handled by each server as per their respective ratios.

We are analysing 3 web servers. In the table and the graph (shown above), x-axis represents total number of requests being served and y-axis represents the number of web servers in our environment (3 servers). In above analysis, server (192.168.112.103) has less load and hence has more capacity to server more clients. That’s why it is processing more no. of requests i.e. 220 as compared with other two servers due to its high configuration and more available resources. The server 192.168.112.101 has the least resources available and hence it serves the least amount of requests (31 requests) compared to 192.168.112.102 and 192.168.112.103 which have better configurations (92 requests and 220 requests). The total requests are being split based on their serving capacity.

Secondly we performed testing for existing algorithm for comparison on same environment to test the performance difference between the two algorithms.

7.2 DATA ANALYSIS & INTERPRETATION

We have performed different tests with different scenarios. We have done testing with 500 users, 1000 users and 1500 users.

In the 500 users scenario, we have simulated 500 users (threads) with ramp up time 0 seconds means all concurrent users. In the Jmeter *View Result Tree* is used for checking the result of ping test performed by Jmeter. In this it shows list of all the requests made. If the request color is green in the list, it means that the request has successively completed. Also we can check the details for request type done for request and what's the response has been generated by the request. We have 3 different web servers and on each server we have different sites/home pages set up as showing Server 1, Server 2 and Server 3 responses for each respectively. When the response is generated from the Jmeter, it will show the HTML part of the request response. If the request on Jmeter is not successive then it will shows its color in red representing the failed request.

The request which is failed is showing the error html part in response. From test case of 500 users the CPU, RAM, Disk Usage and Active connections graphs are observed.

In the 1000 users scenario, we have simulated 1000 users (threads) with ramp up time 0 seconds means all concurrent users. From test case of 1000 users the CPU, RAM, Disk Usage and Active connection graphs are observed.

In the 1500 users scenario, we have simulated 1500 users (threads) with ramp up time 0 seconds means all concurrent users. From test case of 1500 users the CPU, RAM, Disk Usage and Active connection graphs are observed.

7.3 PERFORMANCE EVALUATION

We have prepared 2 similar environments to compare the results

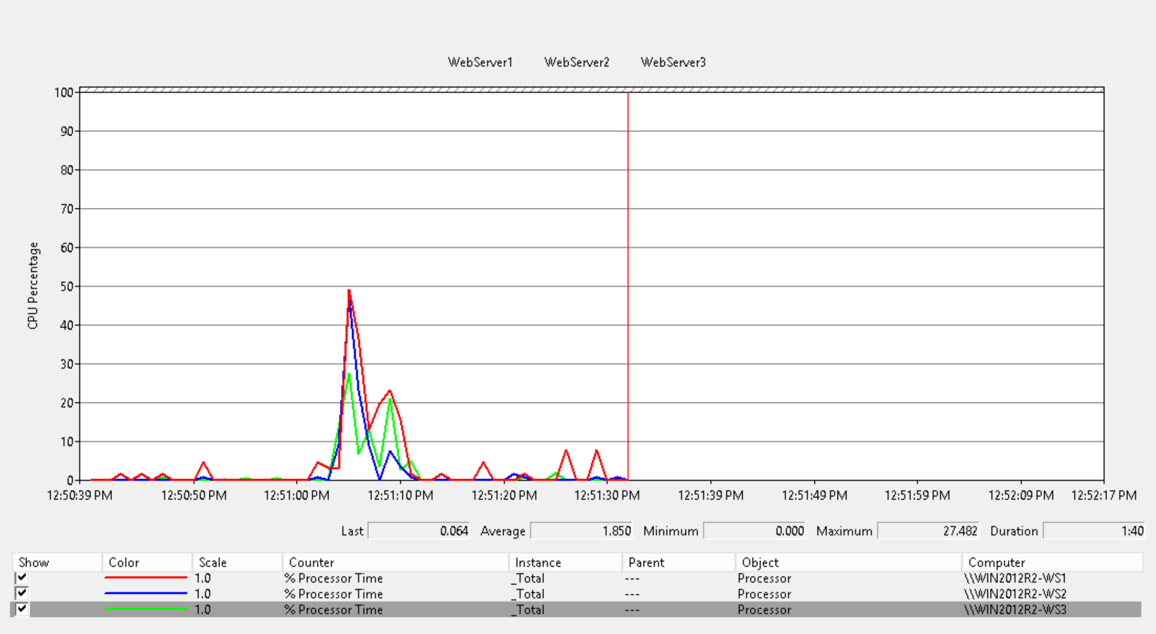
1. Existing Load Balancing Technology
2. Custom Load Balancing technology (in this lab)

We have run the same test scenarios on Windows Network Load Balancing and then the same test cases were run against the custom load balancing method.

This new custom algorithm is based on following system parameters

1. CPU
2. RAM
3. Disk Usage
4. Requests per second

Each Graph of CPU will denote the concurrent reading of the CPU of 3 web-servers respectively as below.



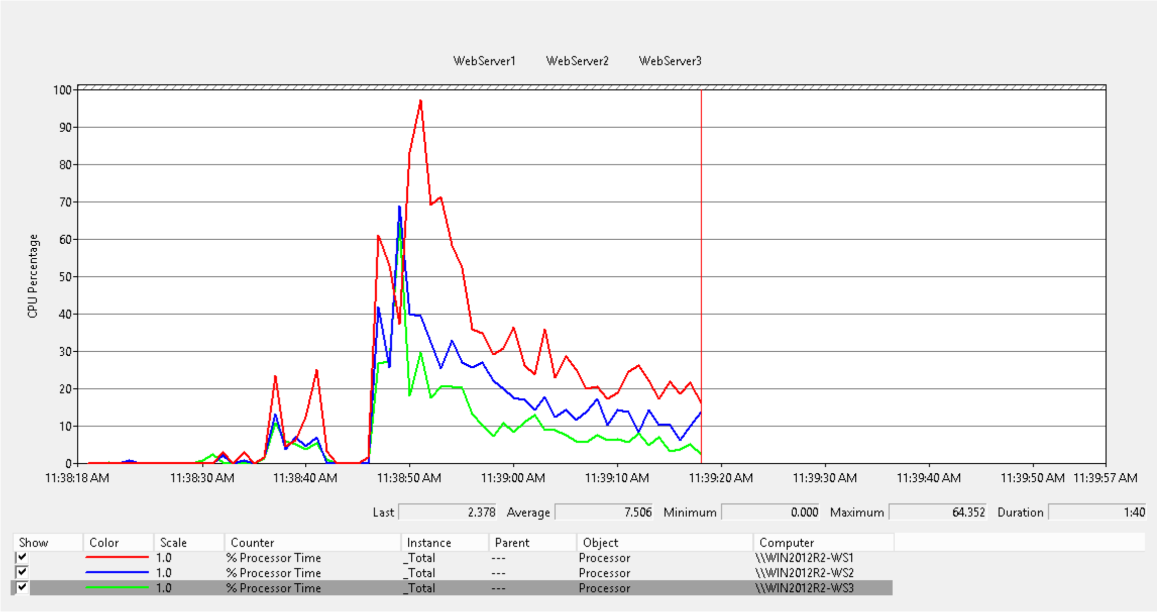
Last 0.064 Average 1.850 Minimum 0.000 Maximum 27.482 Duration 1:40

Show	Color	Scale	Counter	Instance	Parent	Object	Computer
☑	🟥	1.0	% Processor Time	_Total	---	Processor	\\WIN2012R2-WS1
☑	🟦	1.0	% Processor Time	_Total	---	Processor	\\WIN2012R2-WS2
☑	🟩	1.0	% Processor Time	_Total	---	Processor	\\WIN2012R2-WS3

Figure 9 CPU Usage with New Algorithm

Under new load balancing technique, from the above graph it is observed that the system which has more resources available (in Green and Blue) has a little spike in CPU which represents that they have received requests and started processing requests as they come. The CPU is not high because they have more resources available compared to server in Red. The spike on Red is high because it was already low on resources. Our custom algorithm distributes the maximum load on the server with high configuration and least load on the server the low configuration.

When we ran the same test with existing Windows load balancing (graph below), we noticed Server1 which was already low on resources (in Red) went almost to 90% to process the requests. Hence we have achieved better CPU handling with this new load balancing algorithm.



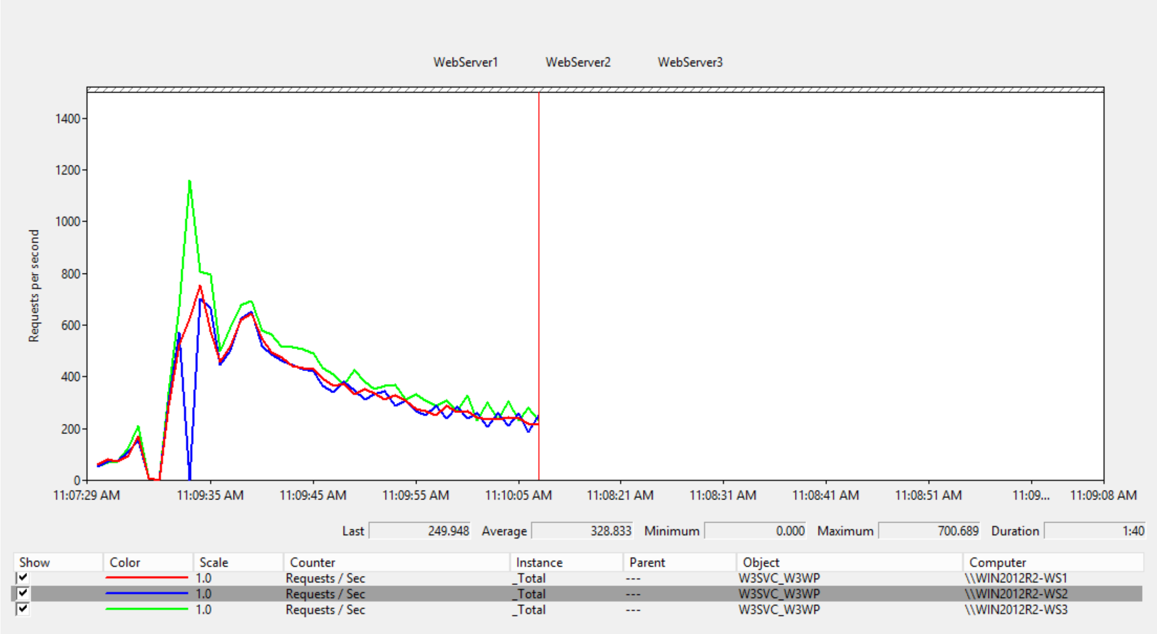
Last 2.378 Average 7.506 Minimum 0.000 Maximum 64.352 Duration 1:40

Show	Color	Scale	Counter	Instance	Parent	Object	Computer
☑	🟥	1.0	% Processor Time	_Total	---	Processor	\\WIN2012R2-WS1
☑	🟦	1.0	% Processor Time	_Total	---	Processor	\\WIN2012R2-WS2
☑	🟩	1.0	% Processor Time	_Total	---	Processor	\\WIN2012R2-WS3

Figure 10 Existing Algorithm CPU

From this above graph it is also observed that the system which has low configuration is working harder compared to other 2 servers which are better in resources.

The blue has intermediate rise and the last green color has average spike overall because the last server is of high configuration. Existing algorithm does not distribute the load on servers evenly, which resulted in Server 1 becomes busier for a longer period of time.

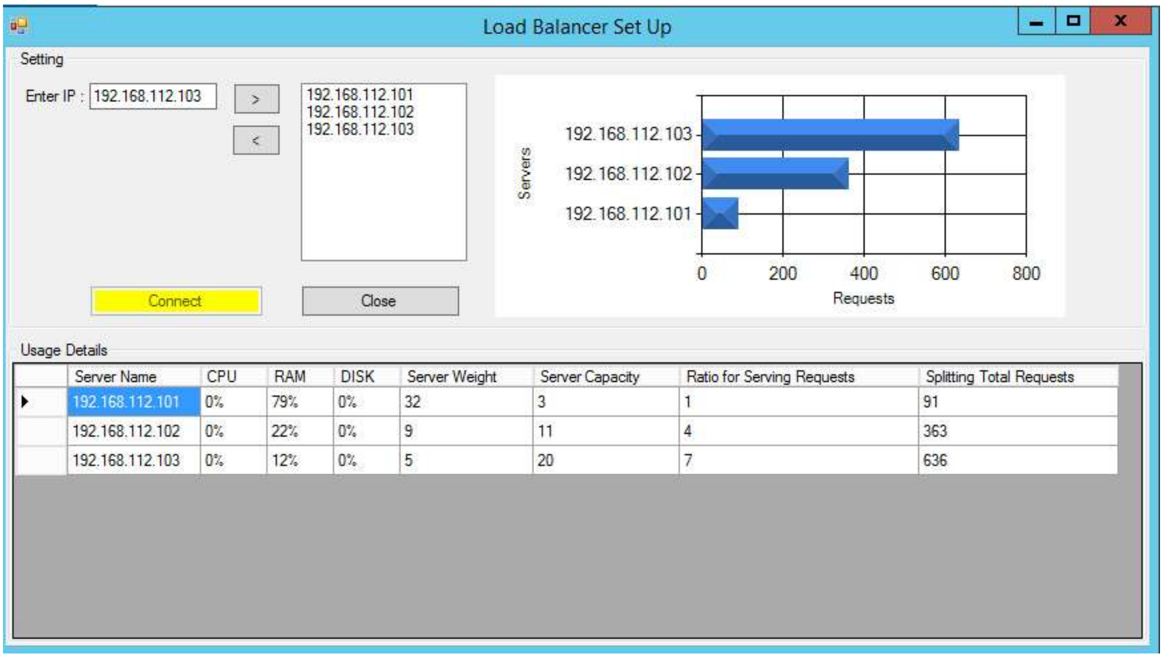


Last 249.948 Average 328.833 Minimum 0.000 Maximum 700.689 Duration 1:40

Show	Color	Scale	Counter	Instance	Parent	Object	Computer
☑	🟥	1.0	Requests / Sec	_Total	---	W3SVC_W3WP	\\WIN2012R2-WS1
☑	🟦	1.0	Requests / Sec	_Total	---	W3SVC_W3WP	\\WIN2012R2-WS2
☑	🟩	1.0	Requests / Sec	_Total	---	W3SVC_W3WP	\\WIN2012R2-WS3

Figure 11 Requests handled by the New Algorithm

In the above Figure 11 the request distribution is observed with new algorithm. The new algorithm distributes more requests on server which has more resources available (expressed with green line). Blue represents server 2 which serves moderate request and red is server 1 which is serving the least amount of requests.



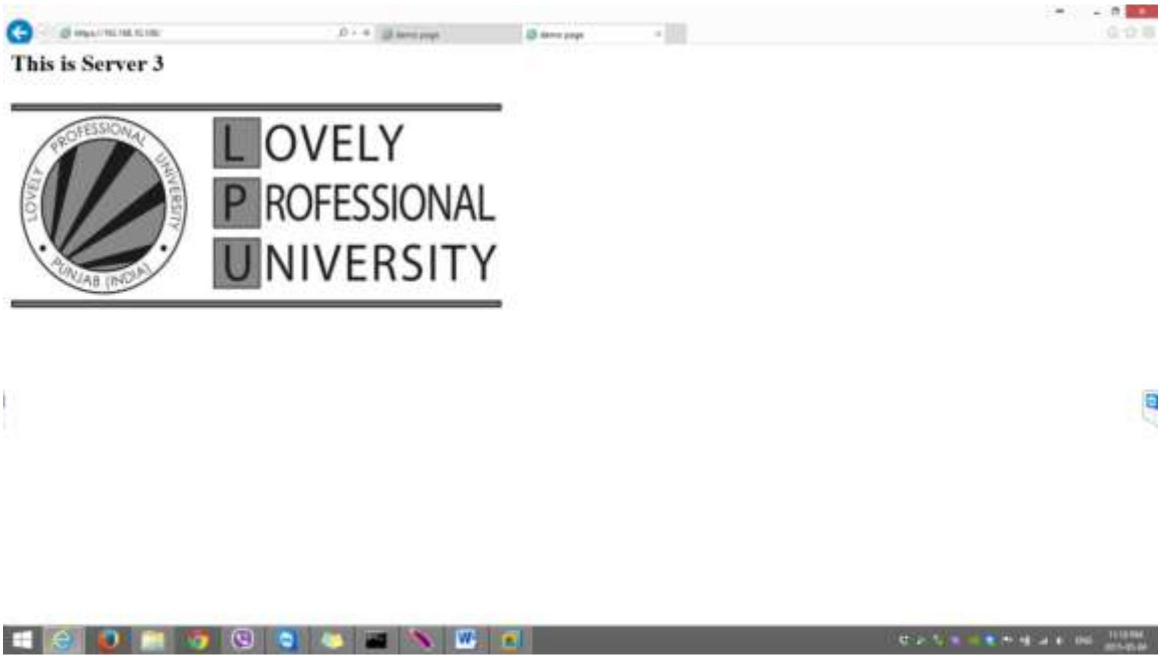
Enter IP : 192.168.112.103

192.168.112.101
192.168.112.102
192.168.112.103

	Server Name	CPU	RAM	DISK	Server Weight	Server Capacity	Ratio for Serving Requests	Splitting Total Requests
▶	192.168.112.101	0%	79%	0%	32	3	1	91
	192.168.112.102	0%	22%	0%	9	11	4	363
	192.168.112.103	0%	12%	0%	5	20	7	636

Figure 12 Requests distributed by Custom Load Balancer

The above snapshot was taken from CLB (Custom Load Balancer) which is showing the distribution of requests in real time according to the load supplied to the server. The server with IP 192.168.10.103 has the least load and has more serving power and is serving the maximum requests. Server 192.168.10.102 is in average performance and the server 192.168.10.101 has least amount of resources available and hence is serving the least amount of requests.



This is Server 3

Figure 13 Output of the Webpage from Web-server 3

In this screenshot the reply is generated from server 3 which processes most of the requests. The server 3 handles most of the requests.

Chapter 8
CONCLUSION & FUTURE SCOPE

Our work is the implementation of SSL based servers and improved weight distribution algorithm in the Custom Load Balancer. Our servers are handling more no. of concurrent requests on the basis of the load on the server with different health parameters that are used by the LB to calculate the load on each server. This can lead a server to perform better and handle the requests more efficiently and throughput of system will be increased. System can use different configuration on servers to handle the traffic using the improved weighted distribution algorithm. As the weight calculation of server is a sophisticated task, it can be further enhanced using more health parameters or any other algorithm to produce more accurate results. The future work of our work can be done in sticky session that whenever a connection is created with SSL then a connection which is connected with particular server may stick with that one until the user doesn't closes the connection or session is not expired. With our work the system is more optimized so that it can put less no. of requests in waiting and more waiting queued requests will be served quickly and efficiently.

Chapter 9

REFERENCES

I. Research Papers:

[1] Lamprecht, Christiaan J., and Aad PA van Moorsel. "Runtime security adaptation using adaptive SSL." In *Dependable Computing, 2008. PRDC'08. 14th IEEE Pacific Rim International Symposium on,* pp. 305-312. IEEE, 2008.

[2] T. Dierks, E. Rescorla, "IETF RFC 5246 The Transport Layer Security (TLS) Protocol Version 1.2". 2008

[3] Chunmei Zheng, Junbao Xia,Qun Wang, Xiangyang Chu. "Design and implementation of a Load Balancing Model for Web-Server Cluster System." In *Computer Science & Education (ICCSE), 2012 International Conference on,* pp. 737-740. IEEE, 2012.

[4] Wang, Ziyu, Lixin Pang, and Yunfei Fan. "Analysis of Load-Balancing of Web Cluster Based on TLS Session Sharing." In *Education Technology and Computer Science, 2009. ETCS'09. First International Workshop on,* vol. 3, pp. 321-323. IEEE, 2009.

[5] Qun Wang , Chunmei Zhang, Xiangyang Chu "Design & Implementation of a Load Balancing model for Web-Cluster System"IEEE.2012

[6] Charles P Wright, Erich Nahum"Design, Implementation & Performance of a Load Balancer for SIP Server Clusters"IEEE.2012

[7] Mengna Zhang, Hongyang Yu. "A New Load Balancing Scheduling Algorithm based on Linux Virtual Server." In *Computer Science & Education (ICCSE), 2013 International Conference on,* pp. 737-740. IEEE, 2013.

[8] Zhenbin Yan , Wenzheng Li "Research of a Scheduling & Load Balancing Scheme Based on Large Scale Distributed Systems"IEEE.2012

[9] Deepak C. Shadrach, Kiran S. Balagani, Vir V Phoha."A Weighted Metric Based Adaptive Algorithm for Web Server Load Balancing." In *International Symposium on Intelligent Information Technology Application (ISIITA), 2009 International Conference on,* pp. 737-740. IEEE, 2009

[10] Carlos Oliveira. "Load Balancing on Virtualized Web-Servers." In *International Conference on Parallel Processing Workshops (ICPPW), 2014 International Conference on,* pp. 340-345. IEEE, 2014.

[11] Jorg Jung, Simon Kiertscher, Sebastian Menski, Bettina Schnor. "Self-Adapting Load Balancing for DNS." In *Computer Science & Education (ICCSE), 2012 International Conference on,* pp. 564-571. IEEE, 2012.

[12] Zinke, Jörg, and Bettina Schnor. "The impact of weights on the performance of Server Load Balancing systems." In Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2013 International Symposium on, pp. 30-37. IEEE, 2013.

[13] Pateriya, Rajesh Kumar. "Web server load management with adaptive SSL and admission control mechanism." In *Computer Science & Education (ICCSE), 2012 7th International Conference on*, pp. 1178-1183. IEEE, 2012.

[14] Lamprecht, Christiaan J., and Aad PA van Moorsel. "Adaptive SSL: Design, implementation and overhead analysis." In *Self-Adaptive and Self-Organizing Systems, 2007. SASO'07. First International Conference on*, pp. 289-294. IEEE, 2007.

[15] Verma, O. P., Ritu Agarwal, Dhiraj Dafouti, and Shobha Tyagi. "Performance analysis of data encryption algorithms." In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, vol. 5, pp. 399-403. IEEE, 2011.

[16] Lamprecht, Christiaan J., and Aad PA van Moorsel. "Runtime security adaptation using adaptive SSL." In Dependable Computing, 2008. PRDC'08. 14th IEEE Pacific Rim International Symposium on, pp. 305-312. IEEE, 2008.

[17] Suresh, V. M., D. Karthikeswaran, V. M. Sudha, and D. Murali Chandraseker. "Web server load balancing using SSL back-end forwarding method." In *Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on*, pp. 822-827. IEEE, 2012.

[18] Ryosuke HATSUGA, Takamichi SAITO" Load – Balancing SSL Cluster Using Session Migration"IEEE.2007

[19] Limkar, Suresh V., Rakesh Kumar Jha, Shilpa Pimpalkar, and Santosh Darade. "Geo-Encryption-A New Direction to Secure Traditional SSL VPN." In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pp. 1070-1071. IEEE, 2011.

[20] Der-Chiang Li , Fengming M.Chang ,"An In-Out Combined Dynamic Weighted Round Robin Method for Network Load Balancing "In The Computer General". Oxford University Press.2007

[21] Fei, Chen, Wu Kehe, Chen Wei, and Zhang Qianyuan. "The research and implementation of the VPN gateway based on SSL." In Computational and Information Sciences (ICCIS), 2013 Fifth International Conference on, pp. 1376-1379. IEEE, 2013.

II. Website:

[22] Microsoft. *How SSL/TLS Works.* Available*: http://msdn.microsoft.com/en-us/library/windows/desktop/aa374757 (v=vs.85).aspx*

[23] Oracle. *What is the best webserver hardware configuration?* Available: *http://www.dba-oracle.com/t_best_webserver_hardware_configuration.htm*

[24] Microsoft .*TLS Handshake Protocol* .Available: *http://msdn.microsoft.com/en-us/library/windows/desktop/aa380513 (v=vs.85).aspx*

[25] Private Internet Access. *VPN Encryption*. Available: *https://www.privateinternetaccess.com/pages/vpn-encryption*

[26] Load Balancing Algorithms Available: http://www.loadbalancerblog.com/blog/2013/06/load-balancing-scheduling-methods-explained

Chapter 9

APPENDIX

Glossary	**Page No.**
Active Connections	7
Adaptive	17
ASSL	18
Block Cipher	19
Cipher	19
CA	2
CBC	5
CPU Usage	10
Content Generator	29
Disk Usage	32
ECB	7
Fixed Weight	19
Geo-Encryption	22
GPS	22
Handshake	3
Handler	19
Hooks	19
Health Parameters	24
Hash Algorithm	2
LB	5
Least Connection	7
Least Connection Slow Start Time	7
Max	17
Monitoring Agent	17
Negotiation	20
NIC	20
Overload	23
Pool	14
Queue	25
RS	25