Implementing Three-Tier Application on Cisco Unified Computing System Using Open Source Stack

Red Hat Linux, Apache, JBoss & MySQL

White Paper

July 2011

Contents

1. Executive Summary

Enterprise web applications primarily constitutes a three-tier architecture comprising of a user interface, functional process logic ("business rules"), application data storage and data access logic. All the tiers are generally developed and maintained as independent modules, oftenly on separate platforms.

Apart from the usual advantages of modular software with well-defined interfaces, the three-tier architecture is intended to allow any of the tiers to be upgraded or replaced independently as requirements or technology change. For instance, upgrading of presentation tier to Rich User Interface would only affects the UI code residing at the presentation layer.

A three-tier architecture is considered to be the most suitable architecture for large, Web-based enterprise applications. The multi layered partitioning of the application enables rapid design and development of the enterprise system. The modularity makes it easier to make changes to just one tier without affecting the others. Separating the functions into distinct tiers makes it easier to monitor and optimize the performance of each layer. Scaling the application deployment through Load balancing and clustering can take place independently at each layer and it becomes simpler to scale the system across multiple processors on different machines.

Even though highly scalable three-tier web deployments provide numerous benefits, high system maintenance and difficulty to upgrade or migrate physical servers, challenge customers to achieve full benefits of the technology. IT departments are constantly facing existing rigid, inflexible hardware platforms. To address these challenges and also as a natural next step in the Cisco® Data Center 3.0 vision, Cisco has developed the Cisco Unified Computing System™ (UCS). The Cisco Unified Computing System is the next-generation data center platform that unites compute, network, and storage access. The platform is designed using open industry-standard technologies and aims to reduce total cost of ownership (TCO) and increase business agility. Cisco UCS servers are powered by Intel® Xeon® processors.

This document describes how the Cisco Unified Computing System™ can be used to deploy a three-tier web application on open source servers such as, Apache, JBoss and MySQL. It further outlines a performance benchmark executed to elaborate upon scaling of application by addition of multiple AppServer instances. The Cisco Unified Computing System provides the compute, network, and storage access components of the system, deployed as a single cohesive system. The document introduces the Cisco Unified Computing System and provides instructions for implementation. It concludes with an analysis of the performance gains achieved over application throughput and end user response time through multiple instances of AppServer deployed on Cisco Unified Computing System.

2. Introduction

Enterprise web deployments are typically multi-tier applications with primary focus on three tiers; web server, application server, and database server.In real world scenarios. Three-tier web deployments enjoy several benefits such as Performance , it helps in scaling the application instances with a cluster of application servers , thus providing on-demand application performance during unprecedented demands of user scalability.Failover,ease of configuring redundant application server instances in production deployments. This leads to low down times and high reliability of deployed systems. Security, decoupling webserver and application server systems thus providing secure middle tier application servers that host business logic and data access services.

Irrespective of the scale of web deployments, scalability, reliability and availablity of enterprise web applications remain major challenges, for architects, IT adminsitrators and support engineers. Limitations of application architecture and distruptive scaling of hardware results in high application downtime and increased end user response time, leading to end user dissatisfaction. This has resulted in lesser revenues or even complete shutdown of enterprise web business.

To cater to small, medium and large scale hardware infrastructure needs of web applications, Cisco is now equipped to provide the entire clustered infrastructure for a web deployments. The Cisco Unified Computing System provides compute, network, virtualization, and storage access resources ,with the capability to scale to up to 320 servers and incorporate both blade and rack-mount servers in a single system.The Cisco Unified Computing System provides an ideal foundation to cater ever increasing demand for improved performance and scalability of web based deployments.

In the present performance benchmarks , test were conducted to determine the benefits of clustered deployment through single application server instance, deployed over Cisco UCS Blade and Rack-Mount server platform. End-user response time and application throughput are used as the key performance indicators in evaluating performance benefits. The three-tier solution described in this paper is designed to support real-world stock trading application users. The platforms tested include:

For Large Scale implementation

- Cisco UCS B200 M2 Blade Server with Intel Xeon X5680 3.33-GHz processor with 24 GB total memory
- Cisco UCS B250 M2 Extended Memory Blade Server with Intel Xeon X5680 3.33-GHz processor with 96 GB total memory
- EMC CLARiiON AX4, used as the storage platform

For Small and Medium implementations

- Cisco UCS C200 M2 High Density Rack Mount server with Intel Xeon X5670 2.93-GHz processor with 24 GB total memory
- Cisco UCS C210 M2 General Purpose Rack Mount Server with Intel Xeon X5670 2.93-GHz processor with 48 GB total memory

Subsequent sections in this document detail the Cisco Unified Computing System, solution implemented, performance results, and insights into extension of study to achive maximum benefits through Cisco Uniffied Computing System.

3. Solution Component Overview

3.1 Cisco Unified Computing System

The Cisco Unified Computing System addresses many of the challenges faced during three-tier application deployment and maintenance. For instance, non-distruptive scaling by addition of new servers to manage ever increasing demand of higher throughput.

The Cisco Unified Computing System is a next-generation data center platform that unites compute, network, storage access, and virtualization into a cohesive system designed to reduce total cost of ownership (TCO) and increase business agility. The Cisco Unified Computing System server portfolio consists of the Blade Server platform, B-Series and the C-Series Rack Mount platform. We chose the Cisco UCS B-Series Blade Server platform for this study. The system integrates a low-latency; lossless 10 Gigabit Ethernet unified network fabric with enterprise-class, x86-architecture servers. The system is an integrated, scalable, multi-chassis platform in which all resources participate in a unified management domain.

The main system components include:

Compute—the system is based on an entirely new class of computing system that incorporates blade servers based on Intel Xeon 5500 Series Processors. The Cisco UCS blade servers offer patented Cisco Extended Memory Technology to support applications with large datasets and allow more virtual machines per server.

Network—the system is integrated onto a low-latency, lossless, 10-Gbps unified network fabric. This network foundation consolidates what today are three separate networks: LANs, SANs, and high-performance computing networks. The unified fabric lowers costs by reducing the number of network adapters, switches, and cables, and by decreasing power and cooling requirements.

Virtualization—the system unleashes the full potential of virtualization by enhancing the scalability, performance, and operational control of virtual environments. Cisco security, policy enforcement, and diagnostic features are now extended into virtualized environments to better support changing business and IT requirements.

Storage access—the system provides consolidated access to both SAN storage and Network Attached Storage (NAS) over the unified fabric. Unifying storage access means that the Cisco Unified Computing System can access storage over Ethernet, Fiber Channel, Fibre Channel over Ethernet (FCoE), and iSCSI, providing customers with choice and investment protection. In addition, administrators can pre-assign storage-access policies for system connectivity to storage resources, simplifying storage connectivity and management while helping increase productivity.

Management—the system uniquely integrates all the system components, enabling the entire solution to be managed as a single entity through the Cisco UCS Manager software. The Cisco UCS Manager provides an intuitive graphical user interface (GUI), a command-line interface (CLI), and a robust application programming interface (API) to manage all system configuration and operations. The Cisco UCS Manager helps increase IT staff productivity, enabling storage, network, and server administrators to collaborate on defining service profiles for applications. Service profiles are logical representations of desired physical configurations and infrastructure policies. They help automate provisioning and increase business agility, allowing data center managers to provision resources in minutes instead of days.

Working as a single, cohesive system, these components unify technology in the data center. They represent a radical simplification in comparison to traditional systems, helping simplify data center operations while reducing power and cooling requirements. The system amplifies IT agility for improved

business outcomes. The Cisco Unified Computing System components illustrated in Figure 1 include, from left to right, fabric interconnects, blade server chassis, blade servers, and in the foreground, fabric extenders and network adapters.

3.1.1 Innovations Supporting Business Benefits

Each of the system's business benefits is supported by a rich set of technical innovations that contribute to this first implementation of the Cisco unified computing vision:

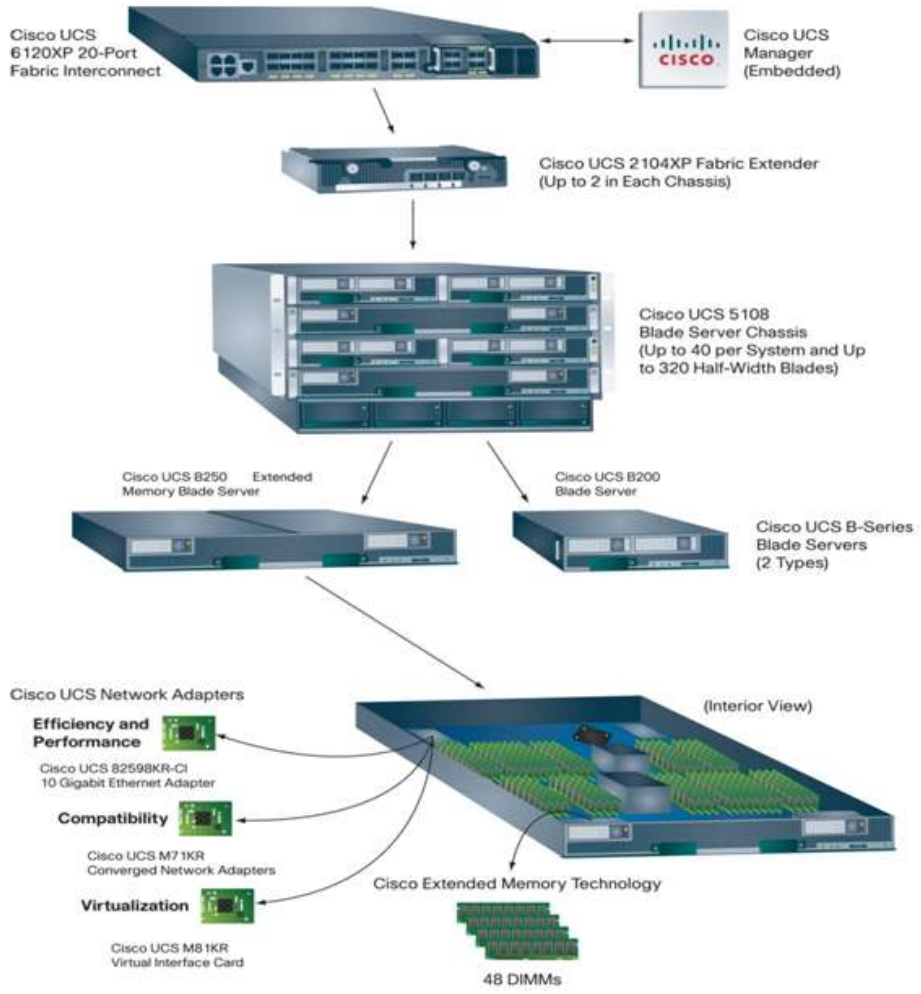
- Embedded system management through Cisco UCS Manager
- Just-in-time provisioning with service profiles
- Unified fabric using 10-Gbps Ethernet
- Cisco VN-Link virtualization support
- Cisco Extended Memory Technology
- State-of-the-art performance using Intel Xeon Processors
- Energy-efficient platform design

3.2 Cisco Unified Computing System Components

Cisco Unified Computing System used for three-tier application performance and scalability analysis, is based on Cisco B-Series Blade Servers; however, the breadth of Cisco's server and network product line suggests that similar product combinations will meet the same requirements. The Cisco Unified Computing System uses a form-factor-neutral architecture that will allow Cisco C-Series Rack-Mount Servers to be integrated as part of the system using capabilities planned to follow the product's first customer shipment (FCS). Similarly, the system's core components -- high-performance compute resources integrated using a unified fabric -- can be integrated manually today using Cisco C-Series servers and Cisco Nexus™ 5000 Series Switches.

The system used to create the three-tier applicaton configuration is built from the hierarchy of components illustrated in 0

Figure 1. Cisco Unified Computing System Components



- The Cisco UCS 6120XP 20-Port Fabric Interconnect provides low-latency, lossless, 10-Gbps unified fabric connectivity for the cluster. The interconnect provides connectivity to blade server chassis and the enterprise IP network. Through an 8-port, 4-Gbps Fibre Channel expansion card, the interconnect provides native Fibre Channel access to the EMC CLARiiON storage system.
- The Cisco UCS 2104XP Fabric Extender brings the unified fabric into each blade server chassis. The fabric extender is configured and managed by the fabric interconnects, eliminating the complexity of blade-server-resident switches. Two fabric extenders are configured in blade server chassis. It uses two of the four available 10-Gbps uplinks to connect to one of the two fabric interconnects.
- The Cisco UCS 5108 Blade Server Chassis houses the fabric extenders, up to four power supplies, and up to eight blade servers. As part of the system's radical simplification, the blade server chassis is also managed by the fabric interconnects, eliminating another point of management. Single chassis is used in the present deployment
- The blade chassis supports up to eight half-width blades or up to four full-width blades. The present configuration uses 2 Cisco UCS B200 M2 Blade Servers and 1 Cisco UCS B250 M2 Blade servers. Each server is equipped with two six-core Intel Xeon 5680 processors at 3.33 GHz. Cisco UCS B200M2 server (half-width server) was configured with 24 GB of memory and Cisco UCS B250 M2 (Full-width server) was configured with 96 GB through the use of a Cisco Extended Memory Technology.

B200 M2 Blade Server can be expanded upto 96GB of physical , but considering the size of work load and the number of instances in Application Server cluster, we have configured it with 6 DIMMs of 4GB each giving a total physical memory of 24G. Similarly, B250 M2 server with Cisco Extended Memory technology can be expanded to 384G of physical memory, but we have configured 24 DIMM sloots with 4GB each , giving a total physical memory of 96G.

3.2.1 Cisco UCS Manager

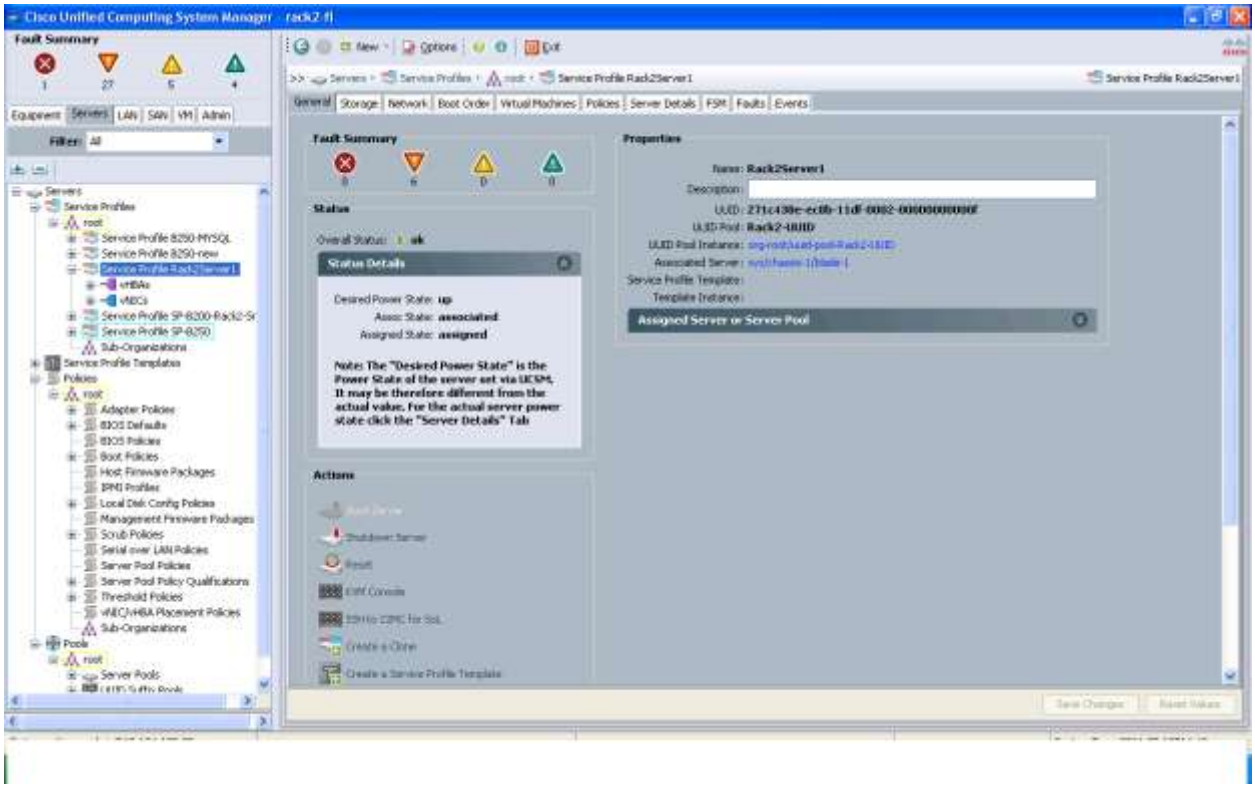
Cisco UCS Manager serves as an embedded device manager for all Cisco Unified Computing System components. The Cisco UCS Manager creates a unified management domain that serves as the central nervous system of the Cisco Unified Computing System. In one sense, Cisco UCS Manager takes the place of the system management tools associated with a traditional computing architecture by integrating computing, networking, and virtualization resources into one cohesive system. Cisco UCS Manager implements policy-based management using *service profiles* to help automate provisioning and increase agility.

In managing the services within the Cisco Unified Computing System, configuration details are applied to Service Profiles, instead of many tedious touches of a physical server and the associated LAN, SAN, and management networks. Service Profile includes all the firmware, firmware settings, and BIOS settings for example, definition of server connectivity, configuration, and identity. This model allows for rapid service instantiation, cloning, growth, shrink, retirement, and re-use in a highly automated fashion. One capability is for a project-by-project instantiation of compute resources with integrated governance, and a life-cycle that returns the physical compute hardware to a pool for other business unit usage. Should the project require a re-build of a previous infrastructure, the stateless nature of Cisco UCS provides for a rapid standup of the prior environment (assuming a SAN boot, or physical disk storage scenario). With other vendors, there is a loosely coupled system of packages having many meshed and customized interconnections. When any of these items is versioned, the effects of updating are not isolated to a given component—they impact other components within a solution. Cisco UCS, with its single source of information and configuration, and open schema, handles upgrades, as well as daily operations, in a simple, straightforward manner.

In the present setup we have used two B200 M2 Servers for Apache and Application Server each, we have configured a Service Profile for one of the B200 M2 servers and simply cloned the profile for the second server. This allows fast provisioning of new servers in the Cisco UCS configuration.

The subsequent figure illustrates the Cisco UCS Manager with Service Profile associated to a B200 M2 server.

Figure 2. Cisco UCS Manager View



3.2.2 Cisco UCS B200 M2 Blade Server

The Cisco UCS B200 M2 Blade Server is a half-width, two-socket blade server. B200 M2 blade server uses two Intel Xeon 5600 Series Processors, with up to 96GB of DDR3 memory, two optional hot-swappable small form factor (SFF) serial attached SCSI (SAS) disk drives, and a single mezzanine connector for up to 20 Gbps of I/O throughput. The server balances simplicity, performance, and density for production-level virtualization and other mainstream data center workloads.

Figure 3. Cisco UCS B200 M2 Blade Server

3.2.3 Cisco UCS B250 M2 Extended Memory Blade Server

The Cisco UCS B250 M2 Extended Memory Blade Server is a full-width, two-socket blade server featuring Cisco Extended Memory Technology. The system supports two Intel Xeon 5600 Series Processors, up to 384 GB of DDR3 memory, two optional SFF SAS disk drives, and two mezzanine connections for up to 40 Gbps of I/O throughput. The server increases performance and capacity for demanding virtualization and large-data-set workloads with greater memory capacity and throughput.

Figure 4. Cisco UCS B250 M2 Extended Memory Blade Server

3.2.4 Cisco UCS C-Series Rack-Mount Servers

Cisco UCS C-Series Rack-Mount Servers [1] based on Intel Xeon 5500, 5600 and 7500 series processors, extend Cisco Unified Computing System innovations to an industry standard rack-mount form factor, including a standards-based unified network fabric, Cisco VN-Link virtualization support, and Cisco Extended Memory Technology.

Designed to operate both in standalone environments and as part of the Cisco Unified Computing System, these servers enable organizations to deploy systems incrementally—using as many or as few servers as needed—on a schedule that best meets the organization's timing and budget. Cisco UCS C-Series servers offer investment protection through the capability to deploy them either as standalone servers or as part of the Cisco Unified Computing System. Current generation of UCS C-Series Rack-Mount Servers is depicted in Figure 5. and a high level comparison of features is in Table 1.

One compelling reason that many organizations prefer rack-mount servers is the wide range of I/O options available in the form of PCI Express (PCIe) adapters. Cisco UCS C-Series servers supports spectrum of I/O options, which includes interfaces supported by Cisco as well as adapters from third parties.

Figure 5. From Left to Right, Cisco UCS C200 High-Density, C460 High-Performance, C250 Extended-Memory, and C210 General-Purpose Rack-Mount Servers

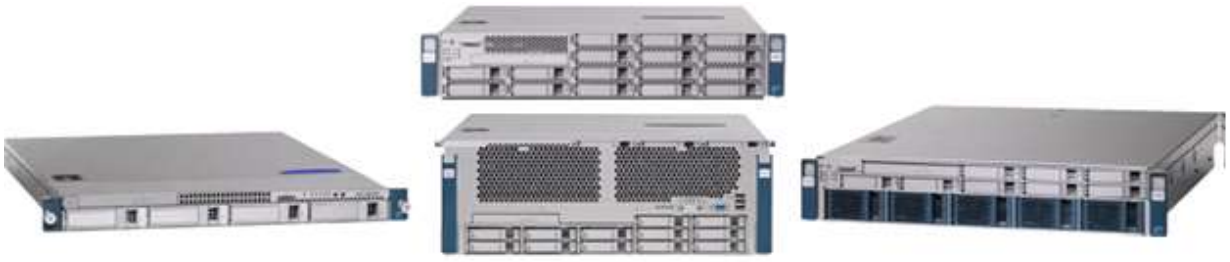


Table 1. Comparison of Cisco UCS C-Series Rack-Mount Server Features

	Cisco UCS C200 M1 and M2	Cisco UCS C210 M1 and M2	Cisco UCS C250 M1 and M2	Cisco UCS C460 M1
Ideal for	Production-level virtualization and mainstream data center workloads	Economical, high-capacity, reliable, internal storage; file, storage, database, and content-delivery	Demanding virtualization and large dataset workloads	High-performance, enterprise-critical stand-alone applications and virtualized workloads
Maximum memory	192 GB	192 GB	384 GB	512 GB
Internal disk drive	Up to 4	Up to 16	Up to 8	Up to 12
Built-in RAID	0 and 1 (SATA only)	0 and 1 (5 SATA drives only)		
Optional RAID	0, 1, 5, 6, and 10	0, 1, 5, 6, 10, 50, and 60	0, 1, 5, 6, 10, 50, and 60	0, 1, 5, 6, 10, 50, and 60
Integrated networking	2X integrated Gb Ethernet; 10 Gb unified fabric optional	2X integrated Gb Ethernet; 10 Gb unified fabric optional	4X integrated Gb Ethernet; 10 Gb unified fabric optional	2X Gigabit Ethernet LAN-on-motherboard (LOM) ports; 2X 10 Gigabit Ethernet ports
I/O via PCIe	Two, half-length, PCIe x8 slots: one full height and one low profile slot	Five, full-height, PCIe, x8 slots: two full length and three half length slots	Five PCIe slots: Three low-profile, half-length x8 slots; 2 full-height, half-length, x16 slots	Ten, full-height PCIe slots: 4 half-length slots, 6 three quarter length slots; 2 Gen 1 slots, 8 Gen 2 slots
Multicore Processors	Up to 2 Intel Xeon 5500 or 5600 Series	Up to 2 Intel Xeon 5500 or 5600 Series	Up to 2 Intel Xeon 5500 or 5600 Series	Up to 4 Intel Xeon 7500 Series

3.2.5 Intel Xeon Processor 5600 Series

The Intel® Xeon® processor 5600 series delivers substantial increases in performance and energy-efficiency versus the previous generation Intel® Xeon® processor 5500 series (Figure 6.). It also provides embedded technologies that give business, creative, and scientific professionals the power to solve problems faster, process larger data sets, and meet bigger challenges. This processor family introduces Intel AES-NI, which accelerates core encryption and decryption processes to enable strong security with less impact on overall server performance.

Figure 6. Intel Xeon Processor 5600 Series

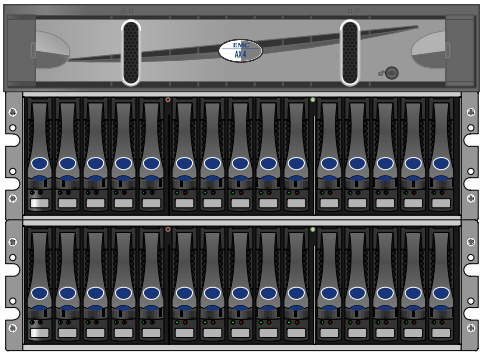
In addition to Intel AES-NI, server platforms based on the Intel® Xeon® processor 5600 series continue to support features from the previous generation processor that enable it to respond intelligently to workloads to provide additional improvements in performance and energy-efficiency.

- Intel® Turbo Boost Technology boosts performance as needed by dynamically adjusting core frequency to increase execution speed for peak workloads.
- Intel® Intelligent Power Technology adjusts core frequencies to conserve power when demand is lower.
- Intel® Hyper-Threading Technology improves throughput and reduces latency for multithreaded applications and for multiple workloads running concurrently in virtualized environments

3.3 EMC CLARiiON

AX4-5 is a new low end, scalable AX CLARiiON array which offers ease of use and scalability. The AX4-5 is an entry level CLARiiON storage system that is expandable to sixty drives. The array can scale from 4 to 60 drives, and support up to 64 HA hosts. There are two versions of the base array: a 1Gbps iSCSI front-end storage system and a 4Gbps Fibre Channel front-end storage system. The drive types supported are Serial Attached SCSI (SAS) or Serial ATA 2 (SATA II). The AX4 can be equipped with serial attached SCSI (SAS) drives for performance-oriented applications and serial ATA (SATA) drives to deliver the lowest cost per gigabyte and highest capacity per drive.

Figure 7. EMC CLARiiON AX4 Series



3.4 Three-Tier Application Architecture

In context of a web application environment, three-tier architecture segregates the application into three specific layers. The content displayed to the end user via a web browser is defined as the *Presentation Layer* and is served via a Web Server. The Mid-tier is responsible for all the business logic processing and is defined as the Application tier. This is where most of the Application Servers come into play such as open-source JBoss App Server. The back-end comprises of data layer, and manages the persistence of application information. *Data-tier* is powered by a RDBMS such as open source databases, MySQL.

Figure 8 defines the functioning of a three-tier web application involving Presentation, Application and Data Access tiers.

Figure 8. Presentation, Application, and Data Access Tiers

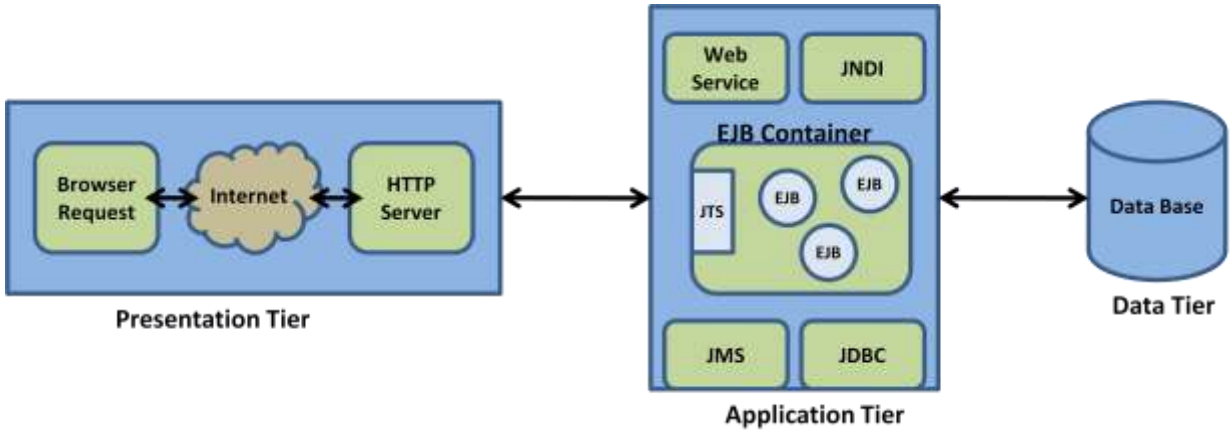


Figure 8 details the workflow of three tier web application into five simplistic events.

- Web Browser send HTTP request to Web Server
- Web Server intercepts the incoming request and request a service on Application Tier
- Application Tier services incoming request with business logic and interaction with Data Tier
- Response is send back to Presentation Tier ,with defined user interfaces in HTML
- HTML response is passed back to client.

Some of the major benefits providing through three-tier web architecture are illustrated below:

a) Maintainability: Three-tier architecture follows a modular approach; each tier is independent of each other. Thus each tier can be updated without affecting application as a whole.

b) Scalability: Scalability is a major benefit to incorporate three-tier architecture, we can scale each tier as and when required without dependence on the other tiers. i.e. independently scaling each tier. For instance in scaling of web servers, provisioning of servers at multiple geographical locations enables faster end user response time, avoiding high network latency. Figure 9. illustrates a location independence view.

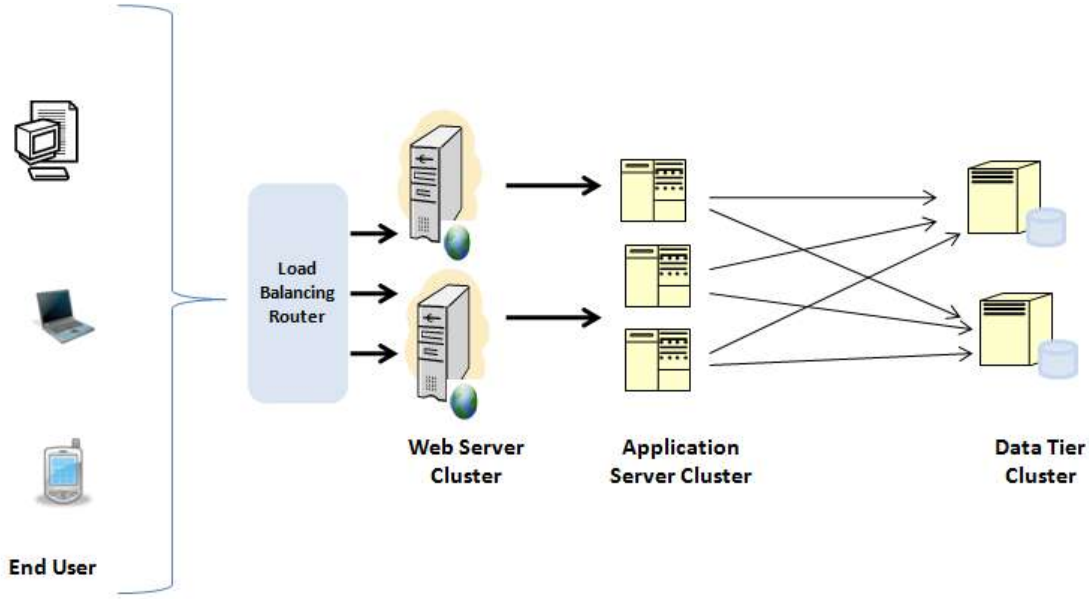
Figure 9. Location Independent

Another aspect is scaling of application servers which require high computing resources, thus a farm of clustered application servers can provide on demand application performance.

In our performance execution and analyzing on the Cisco UCS platform we would demonstrate the benefits of scaling-out the Application tier and thus providing 75% improvement in application throughput without comprising end user response time.

c) Availability and Reliability: Applications using three tier approach can exploit three-tier modular architecture and use scalable components and servers at each tier, thus providing redundancy , avoiding single point of failure and in-turn improving availability of overall system.

Figure 10. Deployment for Reliability and Availability



The important building blocks of a three-tier application are Web-Server, Application Server and Data Base Server. In order to define an open-source three-tier solution we have considered Apache, JBoss and MySQL as open-source servers.

4. Open Source Components Overview

As discussed in previous section, the building blocks of three-tier application architecture are Web-Server, Application Server and DB Server. There are several technologies implementing 3-tier web architecture. Some of the widely know technolgoies can be illustrated as LAMP stack, ASP .NET framework and JAVA/J2EE platform.

LAMP is a stack of open source software that includes the GNU/Linux operating system, Apache Web server, MySQL database and scripting languages Hypertext Preprocessor (PHP), Perl and/or Python.The exact combination of software included in a LAMP package may vary, especially with respect to the web scripting software, as PHP may be replaced or supplemented by Perl and/or Python. Individually, Linux, Apache Web server, MySQL database, Perl, Python or PHP are each a powerful component in its own right. The key to the idea behind LAMP is to provide open source software alternatives which are readily and freely available. This has lead to them often being used together. In the past few years, their compatibility and use together has grown tremendously in small, medium and large enterpise web deployments.

ASP.NET is a web application framework developed and marketed by Microsoft to allow programmers to build dynamic web sites, web applications and web services. It is the successor to Microsoft's Active Server Pages (ASP) technology. ASP.NET is built on the Common Language Runtime (CLR), allowing programmers to write ASP.NET code using any supported .NET language. The ASP.NET SOAP extension framework allows ASP.NET components to process SOAP messages. It free, fully supported Web application framework that helps create standards-based Web solutions. It provides a rich set of features and easy integration with databases, web services and rich internet applications. It provides

features such as WebForms, MVC, dynamic data scaffolding, an AJAX framework, templates and themes.

Java 2 Platform, Enterprise Edition or J2EE is known as Java EE in version 5. The current version is called Java EE 6. Java Platform, Enterprise Edition or Java EE is a widely used platform for server programming in the Java programming language.Java EE includes several API specifications, such as JDBC, RMI, e-mail, JMS, web services, XML, etc., and defines how to coordinate them. Java EE also features some specifications unique to Java EE for components. These include Enterprise JavaBeans, Connectors, servlets, portlets (following the Java Portlet specification), JavaServer Pages and several web service technologies. This allows developers to create enterprise applications that are portable and scalable, and that integrate with legacy technologies. A Java EE application server can handle transactions, security, scalability, concurrency and management of the components that are deployed to it, in order to enable developers to concentrate more on the business logic of the components rather than on infrastructure and integration tasks.The J2EE platform takes advantage of many features of the Java 2 Platform, Standard Edition (J2SE), such as "Write Once, Run Anywhere" portability, JDBC API for database access, CORBA technology for interaction with existing enterprise resources, JMS servers such as IBM MQ-series, Microsoft Message Queue Service (MSMQ), and Tibco Enterprise Messaging Service for communication between distributed applications and a security model that protects data even in internet applications

As elaborated in previous sections , the important builing blocks for 3 tier web stack are web server as the presentation layer , Application Server as the Application or mid-tier and database as the the data tier. Figure 11. illustrates the open-source as well as licensed servers which are majorily used in small , medium and large scale enterpirse web deployments.

Figure 11. Three-Tier Commercial and Open-Source Servers

We have considered an open-source stack, Apache, JBoss and MySQL using Red Hat Enterprise Linux as the OS for all the 3 tiers.These are predominantly used for Building an open-source three-tiered Java/J2EE stack .

4.1 Apache

Apache is generally recognized as the world's most popular Web server (HTTP server).

The Apache Web server provides a full range of Web server features, including CGI, SSL, and virtual domains. Apache also supports plug-in modules for extensibility. Apache is reliable, free, and relatively easy to configure.It has several modules that allow distribution of requests among servers, for scalability, redundancy and increased availability. Apache is developed and maintained by a open community of developers under Apache Software Foundation. It is available for a wide variety of operating systems,

including Unix, GNU, FreeBSD, Linux, Solaris, Novell NetWare, AmigaOS, Mac OS X, Microsoft Windows, OS/2, TPF, and eComStation.In our present setup we have considered Apache 2.2.17 which comes with several new features such as Smart Filtering, Improved Caching, AJP Proxy, Proxy Load Balancing, Graceful Shutdown support, Large File Support, the Event MPM, and refactored Authentication/Authorization.

We have considered Apache as the presentation layer for configuring a web-based three-tier solution.

4.2 JBoss

JBoss Application Server is the open source implementation of the Java EE suite of service. It is a Java EE certified platform for developing and deploying enterprise Java applications, Web applications, and Portals. JBoss Application Server provides the full range of Java EE 5 features as well as extended enterprise services including clustering, caching, and persistence.

Due to the fact it is Java-based, JBoss Application Server is cross-platform, easy to install and use on any operating system that supports Java.

JBoss application Server is considered as ther Mid-tier for deployment on Cisco UCS platform to demonstrate a three-tier solution deployed over open-source technologies.

4.3 MySQL

MySQL database has become the world's most popular open source database because of its high performance, high reliability and ease of use. It is also the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, MySQL, PHP / Perl / Python.). MySQL runs on more than 20 platforms including Linux, Windows, Mac OS, Solaris,IBM AIX. MySQL Database Server comes with several enterprise class features, such as , ACID Transactions to build reliable and secure business critical applications. Stored procedures to improve developer productivity. Triggers to enforce complex business rules at the database level. Views to ensure sensitive information is not compromised. Information schema to provide easy access to metadata. Distributed transactions (XA) to support complex transactions across multiple databases.

In our present performance setup over UCS platform we have deployed MySQL Community Server (GPL) version 5.5.8 configured with INNODB 1.1.4 as the storage engine.

4.4 Red Hat Linux

In the present three-tier setup over UCS platform, we have considered Red Hat Enterprise Linux version 5.5. Red Hat Enterprise Linux and JBoss Enterprise Application Platform are part of Red Hat solutions providing one stop shop for Enterprise level Application Server as well as Linux distribution. Red Hat Enterprise Linux is an enterprise platform suited for a broad range of applications across the IT infrastructure. It provides greater flexibility, efficiency, and control and works across a wide range of hardware architectures, hypervisors, and clouds

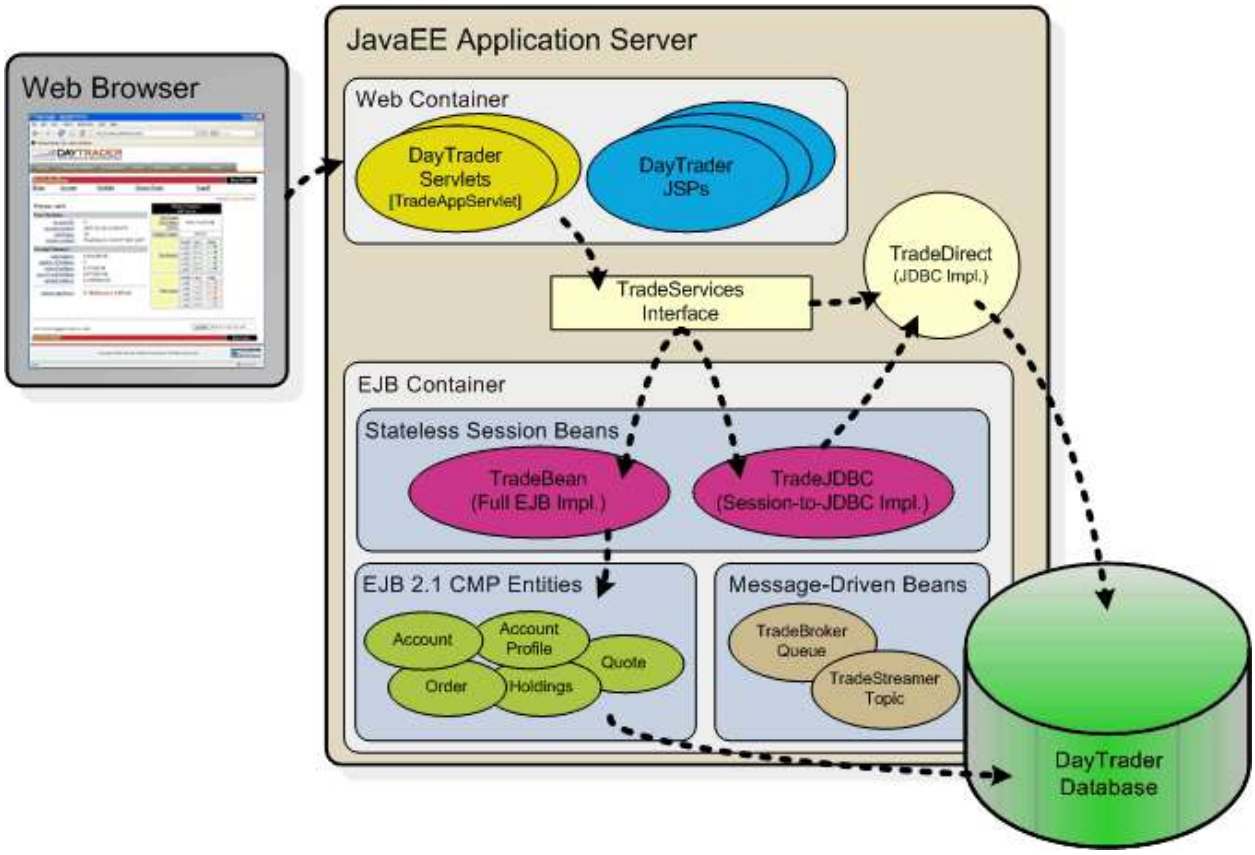
5. Workload Description – DayTrader Application

DayTrader is benchmark application built around the paradigm of an online stock trading system. This application allows users to login, view their portfolio, lookup stock quotes, and buy or sell stock shares. DayTrader is built on a core set of Java EE technologies that includes Java Servlets and JavaServer Pages (JSPs) for the presentation layer and Java database connectivity (JDBC), Java Message Service (JMS), Enterprise JavaBeans (EJBs) and Message-Driven Beans (MDBs) for the back-end business logic and persistence layer.

DayTrader is used to analyse the performance of a 3-tier application deployed over UCS platform using open-source stack, for example Apache, JBoss and MySQL. As DayTrader demonstrates a typical three-tier online stock trading application , we would use it as a medium to demonstrate UCS hardware performance, scalability and sizing estimates.

DayTrader application is considered for the present benchmark, as this application is generally used to measure and compare the performance of Java Platform, Enterprise Edition (Java EE) application servers offered by a variety of vendors, such as Oracle, IBM and Red Hat.

Figure 12. High-Level Overview of DayTrader Application and Various Components



6. Open Source Three-Tier Application Design and Implementation for Large Scale Architectures

6.1 Deployment Architecture

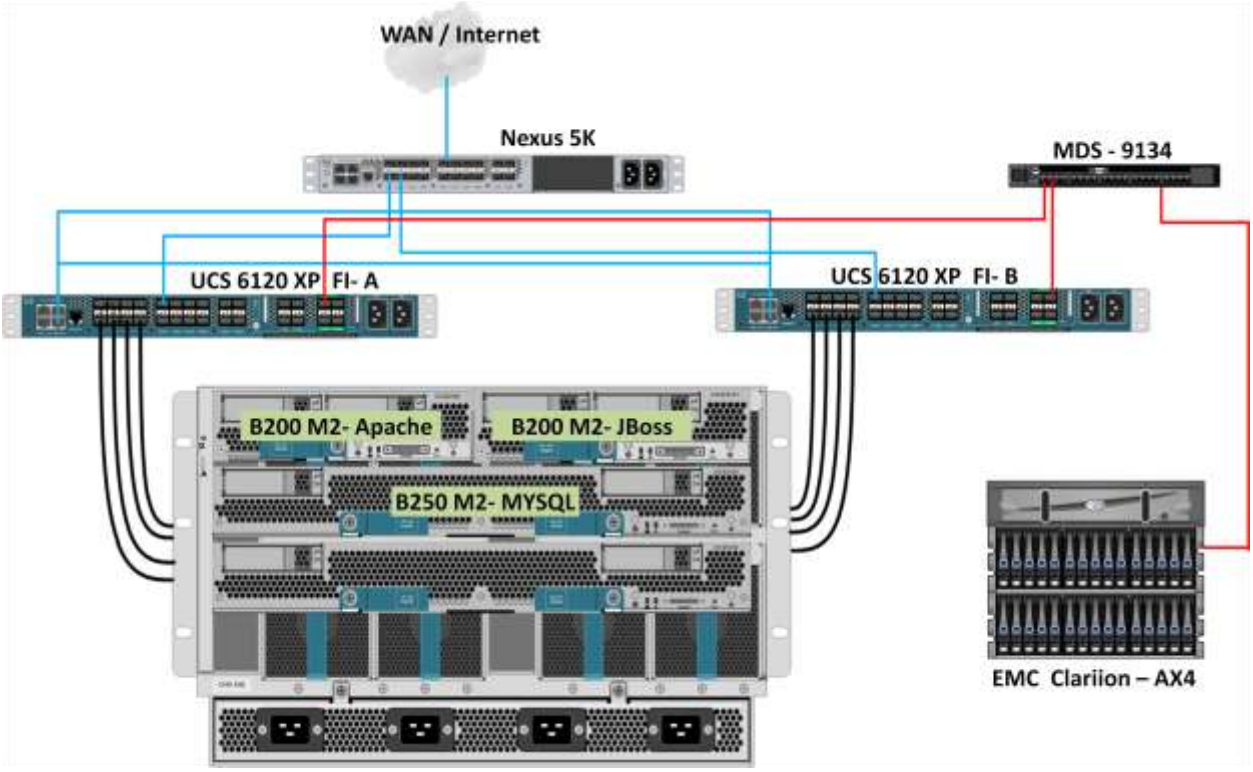
The configuration presented in this document is based a typical three-tier web deployment which has the following main components (Table 2).

Table 2. Configuration Components

WebServer	Apache 2.2 is deployed on UCS B200M2 blade server equipped with two six-core Intel Xeon 5680 processors at 3.33 GHz with a physcial memory of 24G
Application Server	JBoss5.1 is deployed on the second UCS B200M2 blade server equipped with two six-core Intel Xeon 5680 processors at 3.33 GHz with a physcial memory of 24G

Database	MySQL-server-5.5.8-1.rhel5.x86_64 is deployed on Cisco Full width blade sever – B250M2 which is equipped with with two six-core Intel Xeon 5680 processors at 3.33 GHz and configured with 96G of physical memory through the use of a Cisco Extended Memory Technology
Storage	EMC Clariion AX4 is configured with RAID1/0 using 500G of storage space
Operating System (64 bit)	Red Hat Enterprise Linux Server release 5.5

Figure 13. Deployment Architecture Using the Components in Table 2



6.2 Installation and Configuration

Apache, JBoss and MySQL are installed on RHEL5.5 using UCS Blade servers. Some of the important installation and configuration procedures are detailed in subsequent sections.Each section is sub-divided into two important steps for example, install server, configure and sune Server.

6.2.1 Apache

Apache 2.2 is installed on Cisco B200 M2 server powered with two six-cores Intel Xeon 5680 processors at 3.33 GHz and 24 GB of memory. In the present solution we have load balanced JBoss App Server 5 cluster with Apache2 and mod_proxy over http. Some important steps to achieve the same are detailed below.

```
Install Apache:
        1) Download, Build, and Install Apache 2.0 from apache.org.
        2) tar -xzvf httpd-2.2.17.tar.gz
        3) ./configure --enable-so --enable-deflate --enable-proxy --enable-proxy-http --enable-proxy-
connect --enable-proxy-ajp --enable-proxy-balancer  --with-mpm=worker
        4) Make
        5) Make install
        6) Ensure Apache functions properly by starting, testing, and stopping it.
             /usr/local/apache2/bin/apachectl start
             Browse through http://localhost
            /usr/local/apache2/bin/apachectl stop

Configure and Tune Server:
        1) Edit httpd.conf to include mod_proxy and mode_proxy_http modules
        2) Add Balancer members configuration which incljudes loa dbalacing algorithm
            <Proxy balancer://mycluster>
                Order deny,allow
                Allow from all
                BalancerMember http://10.104.109.53:8080/ route=node1 keepalive=On
                BalancerMember http://10.104.109.54:8080/ route=node2 keepalive=On
                ProxySet stickysession=JSESSIONID
                ProxySet lbmethod=byrequests
            </Proxy>
        3) Add virtual Host configuration
            <VirtualHost *:80>
                ServerName 10.104.109.50
                ProxyRequests Off
                <Location /daytrader>
                        ProxyPass balancer://mycluster/daytrader
                        ProxyPassReverseCookiePath /daytrader /
                </Location>
            </VirtualHost>
        4)Define and Tune MPM (Multi-Processing Module)
            Use worker MPM module.
            Edit ThreadsPerChild and MaxClients directives under worker.c module.
                ThreadsPerChild=50
                MaxClients=10000
                ServerLimit=2000
```

Some of the important considerations for tuning as well as sizing Apache servers are detailed below:

1. In the above configuration we have two instances of JBoss and Apache redirects the incoming load with *byrequest* alogrithm, the idea behind this scheduler is to distribute the requests among the various workers to ensure that each gets their configured share of the number of requests.

2. Under *'Proxy Balancer'* tag , all the JBoss nodes are added in accordance to the instance ip address and node name. In case , new nodes are added to JBoss cluster, we just need to add another Balancer Member ip with 'route' as the JBoss node name. The *stickysession* parameter is used to determine which URL session name or cookie to use when looking for the route for the request.

3. In the Apache configuration, we chose to use *mpm_worker* as the multi-processing module; it implements a multi-threaded server,spawning multiple child processes with many threads in each process. Worker generally is a good choice for high-traffic servers because it has a smaller memory footprint than the *prefork MPM* which implements a non-threaded, pre-forking web server.Worker MPM is only available from Apache 2.0 release.On the other side MPM prefork

implements a single control process which is responsible for launching child processes ,listening for connections and serves them when they arrive.This was the only multi-processing module available in apache1.3 release. Prefork is supposed to generally be better for single or dual cpu systems, and worker is better for multi-CPU systems. As we have a two-socket , six-core Xeon processor, mpm_worker is a preferred choice.Multi-processing module for Apache must be chosen during configuration, and compiled into the server.For instance , in the present configuration we had to configure the required MPM option during compilation of apache for linux.The detailed steps are detailed in the preceeding *Install Apache* section.

4. *ThreadsPerChild:* The 'worker' MPM implemented from version 2.0, is similar to 'prefork' mode except that within each child process there will exist a number of worker threads according to *ThreadsPerChild* directive. A request will be handled by a 'thread' within a child process rather than each request being handled by a separate child process ,as in case of prefork MPM. If some of the threads in a process are already handling requests, when a new request arrives, this is handed over to the thread which is ready and idle in the same process. If all worker threads within a child process were busy when a new request arrives the request would be processed by an idle worker thread in another child process.We have increased the ThreadsPerchild value from a default value of 25 to 50 , as this allows lower memory usage and better utlization of per process resources, though more clients are impacted if one client triggers crash

5. Another important apache configuration directive is *MaxClients* directive, this effectively sets the maximum number of simultaneous requests the apache server can handle without queuing the request in the network device queue. For the worker mpm, the rule of thumb to define relationship between MaxClient and ThreadsPerChild is described by the following formula:

   MaxClients = ServerLimit * ThreadsPerChild

   We have chosen MaxClient as 10000 and ThreadsPerChild as 50, due to which we have raised the *ServerLimit* to 2000.

6.2.2 JBoss Application Server

JBoss 5.1 is installed on Cisco B200 M2 server powered with two six-core Intel Xeon 5680 processors at 3.33 GHz and 24 GB of memory.JBoss Server handles the incoming request from Apache and processes various transaction issues from DayTrader End User.

Some important configuration steps to deploy JBoss as cluster and theron connect with MySQL are described as follows.

```
Define JBoss cluster instance configuration
        1)Edit server.xml to define cluster node name as in the startup paramters
            Copy $JBoss_Home/server/all to $JBoss_Home/server/nodetemplate
            Edit server.xml and add
                <Engine name="jboss.web" defaultHost="localhost"
jvmRoute="node${jboss.messaging.ServerPeerID}">
            jboss.messaging.ServerPeerID would be provided in the startup paramters
        2) Add/Edit HTTP Connector details
            <!-- A HTTP/1.1 Connector on port 8080 -->
            <Connector protocol="HTTP/1.1" port="8080" address="${jboss.bind.address}"
connectionTimeout="20000" redirectPort="8443" maxThreads="5000" enableLookups="false"
acceptCount="100" />
        3) Define mysql-persistence-service.xml
            Delete $JBoss_Home/server/nodetemplate/deploy/messaging/hsqldb-persistence-service.xml
            Copy $JBoss_Home /docs/examples/jms/mysql-persistence-service.xml to
             $JBoss_Home/server/nodetemplate/deploy/messaging directory
            Set Clustered to true in mysql-persistence-service.xml
        4)  Edit DefaultDS datasource under mysql-ds.xml  to point to MySQL
        5) Copy mysql-connector-java-5.1.13-bin.jar to $JBoss_Home/server/nodetemplate/lib
        6) Copy nodetemplate directory as node1 and node2 directoy .
            This ensures two nodes with name as node1 and node2

OS tuning
        Enabled hugepages on the server
            vm.nr_hugepages =  8192

Configure SUN JDK1.6 parameters
         Edit run.conf under $JBoss_Home for JVM tuning
            JAVA_OPTS="-Xmn2G -Xms8G -Xmx8G -Xss256k -XX:MaxPermSize=512m -XX:SurvivorRatio=6"
            JAVA_OPTS="$JAVA_OPTS -XX:+AggressiveOpts -XX:+AggressiveHeap -XX:+UseParallelGC
                -XX:CompileThreshold=10000 -XX:ReservedCodeCacheSize=32m -XX:+UseLargePages
                -XX:+PrintGCDetails -XX:ParallelGCThreads=20"

Sever StartupOptions for each node
        ./bin/run.sh  -c node1 -b 10.104.109.53 -Djboss.platform.mbeanserver
    -Djboss.messaging.ServerPeerID=1 -Djgroups.bind_addr=127.0.0.1  -Djava.net.preferIPv4Stack=true &
        ./bin/run.sh  -c node2 -b 10.104.109.54 -Djboss.platform.mbeanserver
    -Djboss.messaging.ServerPeerID=2 -Djgroups.bind_addr=127.0.0.1  -Djava.net.preferIPv4Stack=true &
```

The startup options use for execution of JBoss instance are defined as follows:

1. In the startup options we have mentioned the *messaging.Server.PeerID* value which is used in server.xml of each node for JBoss cluster. This parameter defines a unique node number, for use with load balancing and to avoid node conflicts.
2. We have used multiple ip on single NIC for each node in JBoss cluster , as according to the JBoss wiki, using multiple IP addresses is the "preferred solution" especially for Production environments.
3. In JBoss startup parameters , the -b command line option is used to bind each instance of JBoss AS to a particular IP address:

   ./run.sh -c node1 -b 10.104.109.53

4. Beginning with JBoss 5.0, JBoss AS includes JBoss Messaging (JBM) as its JMS implementation. JBM requires that each server in the cluster have a unique integer id (whether

the servers are running on the same physical machine or not). This id can be set via the command line, so, to start two instances bound to different IP addresses you would:

./run.sh -c node1 -b 10.104.109.53 -Djboss.messaging.ServerPeerID=1

and

./run.sh -c node2 -b 10.104.109.54 -Djboss.messaging.ServerPeerID=2

5. To enable visibilty of JBoss MBeans through jconsole , we need to use the platform MBean server as the server wherein JBoss installs its mbeans. *-Djboss.platform.mbeanserver* used in the JBoss startup paramters , enables this option , thus alllowing monitoring of MBeans through JConsole.

Another important aspect to configure JBoss for clustered production environemnt , is to use a different database other than the HSQLDB. HSQLDB does not have any transaction isolation. For simple non clustered development evaluation HSQLDB may suffice, but its highly recommended to use transaction safe database server. We have used mysql with InnoDB as the transaction-safe storage. Some of the important steps to configure mysql with JBoss JMS are detailed under *"Define JBoss cluster instance configuration".* In mysql database we have created a jbossjms user and defined the connection proerties under mysql-persistence-service.xml.

6.2.3 MySQL

MySQL-server-5.5.8-1.rhel5.x86_64 is deployed on Cisco Full width blade sever – B250M2 which is equipped with with two six-core Intel Xeon 5680 processors at 3.33 GHz and configured with 96G of physical memory through the use of a Cisco Extended Memory Technology.

In the present solution we would be using INNODB as the default storage engine. MySQL is connect to EMC Clariion AX4 as the data storage system.

Some important stepts in installation and configuration of MySQL5 are defined below.

```
Install & Configure MySQL5
    1) rpm –ivh MySQL-server-5.5.8-1.rhel5.x86_64.rpm
    2) Edit my.cnf to enable InnoDB as default storage engine
    3) Edit important tuning parameters under my.cnf
        Max_connections = 2000
        key_buffer_size = 40G
        table_open_cache = 8192
        thread_cache_size = 512
        query_cache_size= 512M
        thread_concurrency = 24
Enable slow query logging in my.cnf
        log-slow-queries=/var/log/mysqlslowqueries.log
        long_query_time = 3
```

The above configurations are based upon a estimates for 3000-4000 concurrent users using DayTrader application.

6.3 Performance Analysis

6.3.1 Objective

This section evaluates the performance of a three-tier web application deployed on Cisco UCS blade servers using open-source technologies. As discussed in the previous sections , DayTrader is being used as a reference application deployed on the Cisco UCS platform.

Performance and Scalability evaluation of the application deployed on Cisco Unified Computing System is evaluated on three important critierias.

- Throughput–Maximum transaction/sec achieved from the deployed application with the condition of acceptable application response time or saturation of availble system resources
- Response Time–Time taken to execute deployed application transaction. In present solution we would be measuring response time for Buy, Quote and Portfolio transactions
- Multi-Instance Application Server Cluster–Performance improvement either on basis of maximum throughput or lower response time achieved by deploying multiple instances of application server in a cluster.

DayTrader is an appropriate benchmark for this study as this application represents the real world three-tier web deployment scenario leveraging J2EE components such as servlets, JSP files, enterprise beans, message-driven beans (MDBs) and Java database connectivity (JDBC). These components provide a set of services such as login/logout, stock quotes, buy, sell, account details and portfolio using standards-based HTTP and Web services protocols. These are some of the important transactions provided in several online stock trading scenarios.

As we are aware that stock trading scenarios are response time sensitive i.e. web transactions require ambitious sub-second response time.This requirement helps to demonstrate very low network latency of the Cisco UCS platform which helps in achieving low response time requirements.

We further evaluate performance benefits of multiple instances of Application Servers in a cluster deployed on a single Cisco UCS half-width blade server over use of a single AppSerbver instance. The performance benefits are mesaures in terms of application through put and end user response time of business critical transactions.

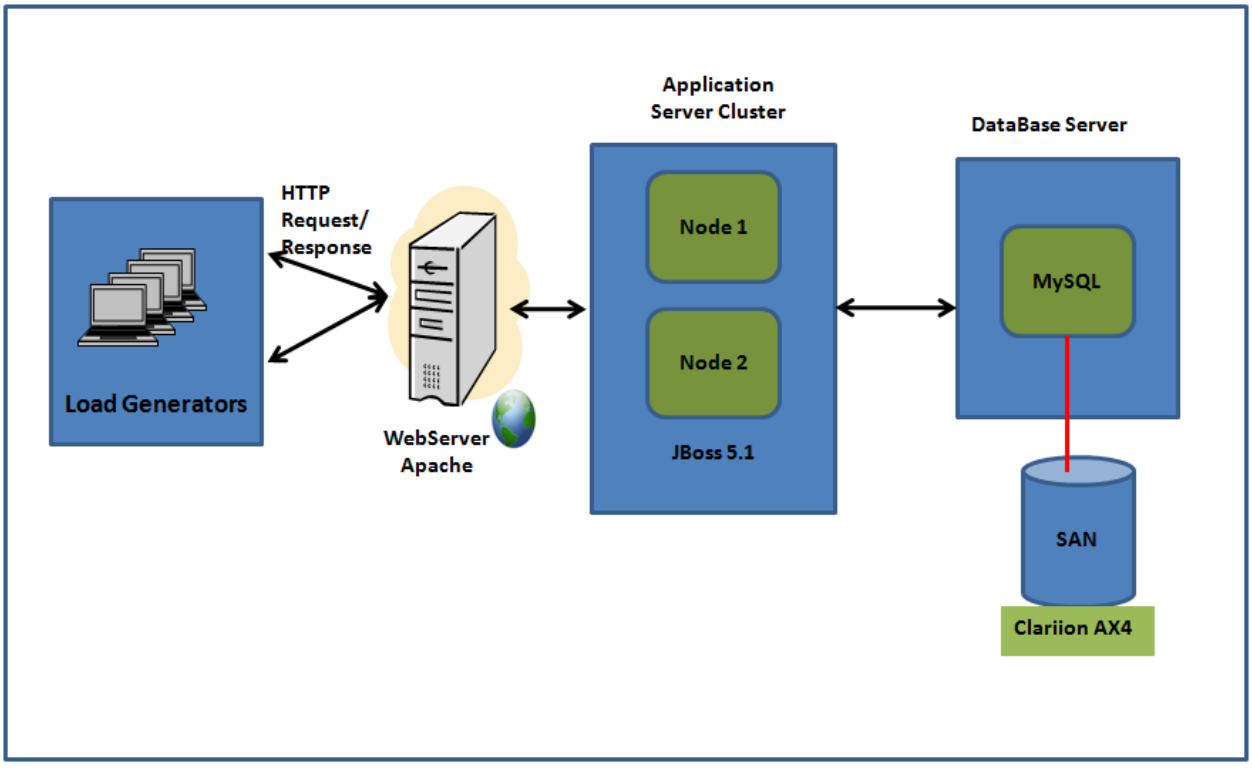
Some of the other important advantages of using a multiple instance application server are *Application isolation* , we can deploy an independent application to each server instance. Each server instance has separate settings, and because each server instance runs in its own instance of the JVM, problems encountered by one application have no effect on other applications.

Another advantage *is Load balancing and failover*, wherein, we can deploy the same application to each server instance and add the instances to a cluster. The web server connector optimizes performance and stability by automatically balancing load and by switching requests to another server instance when a server instance stops running.

6.3.2 Benchmark Configuration

Figure 14. demonstrates the performance benchmark setup using apache, JBoss and MySQL servers on the Cisco UCS platform.

Figure 14. Performance Benchmark Setup



6.3.3 Workload Mix

Performance and Scalability Benchmark wasexecuted with DayTrader JSP/Servlet-based web client which provides a basic set of operations that one would expect to find in any stock trading and portfolio management application, such as Buy, Quote and Portfolio transactions. These high level user operations trigger specific buisness operations within the buisness logic and persistence layers to perform the desired task executed on DayTrader application.

All the three transactions were executed with independent load driver threads ensuring unique users for each type of transaction.The percentage of usersa didicated to each transactions were as follows

- 40% of users executed Buy transaction
- 40% of users executed Quote transaction
- 20% of users executed Portfolio transaction

DayTrader application provides multiple run time modes for the application such as Full EJB3, Direct (JDBC) and Session (EJB3) To Direct mode.The run-time mode for the executed benchmark described in subsequent result sections is configured with Full EJB3 using Hibernate as the JPA layer.

The test were executed with 500 to 3000 concurrent user load , on single and multiple node application server cluster.In order to compare with a real work scenario workload , MySQL database was populated with around 3 million users and 100 million quotes.

6.3.4 Performance Metrics

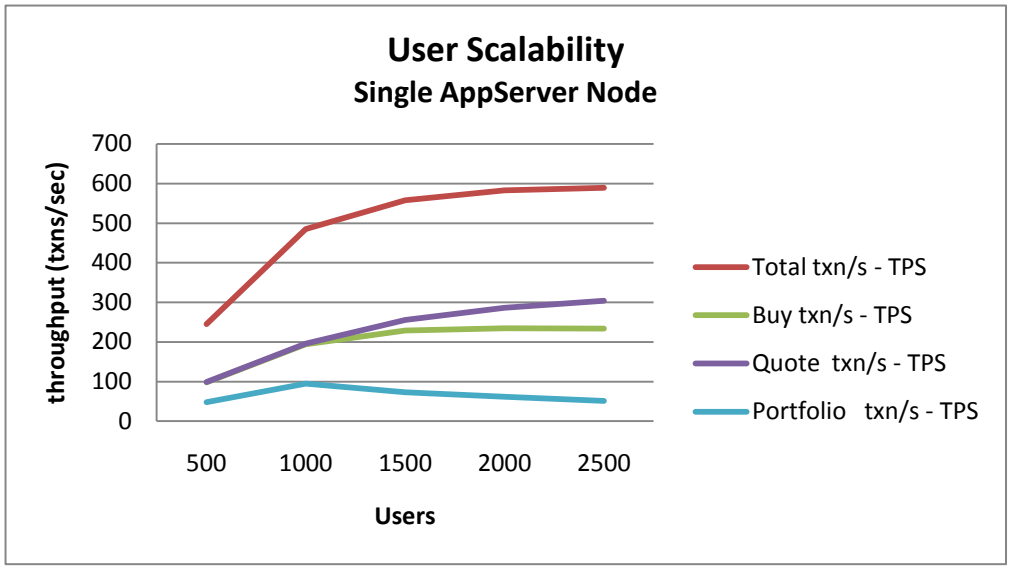
During each test run , several metrics were considered to evaluate the performance of each tier ensuring that Key Performance Indicators (throughput and response time) were at acceptable levels and there were no performance bottlenecks. Some of the metrics were

- CPU utlilization at each tier
- GC pause time of AppServer JVM
- Transaction Throughput
- WebServer Data throughput

6.3.5 Performance Results for Single Application Server Instance

Figure 15. illustrates the variation of application throughput of single node appserver instance for 500 to 2500 concurrent users executed with a workload, as defined in earlier sections.

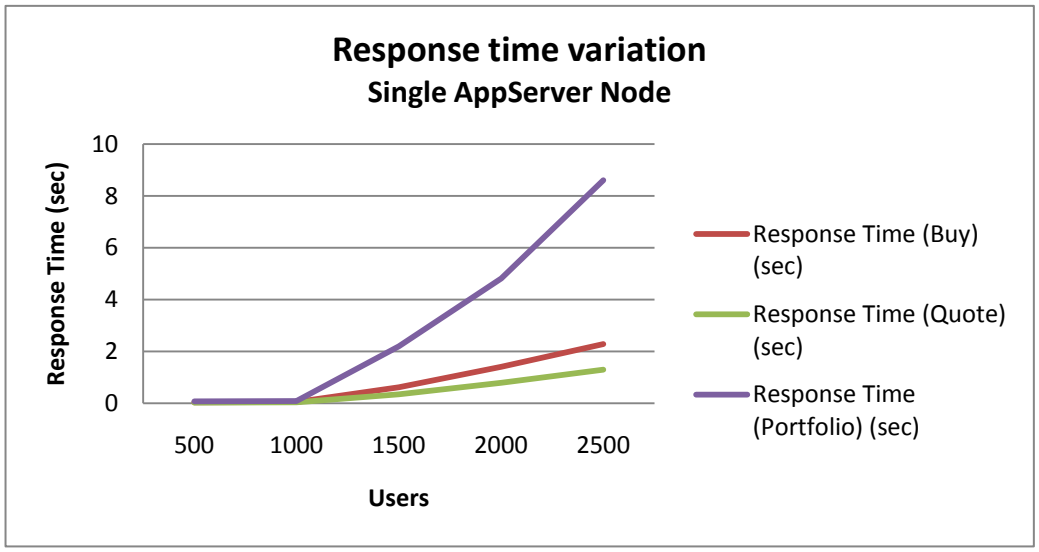
Figure 15. User Scalability



As per Figure 15, the total application throughput varies from 550 to 600 transactions/sec for concurrent user base of 1500 to 2500 users. This demonstrates a saturation state of application at around 1500 concurrent user with single AppServer instance.We also see that the portfolio transaction throughput drops beyond 1000 concurrent users; an important reason for this behavior can be explained by the fact that porfolio transaction which contains all the buy transactions of the user, is a memory intensive transaction and due to limited JVM memory available , the transaction throughput starts to drop beyond a certain limit. A transaction with such behavior is a good candidate for deploying a caching solution over the application server.

Figure 16 illustrates the response time behavior of different transactions executed as per workload matrix with 500 to 2500 concurrent users.

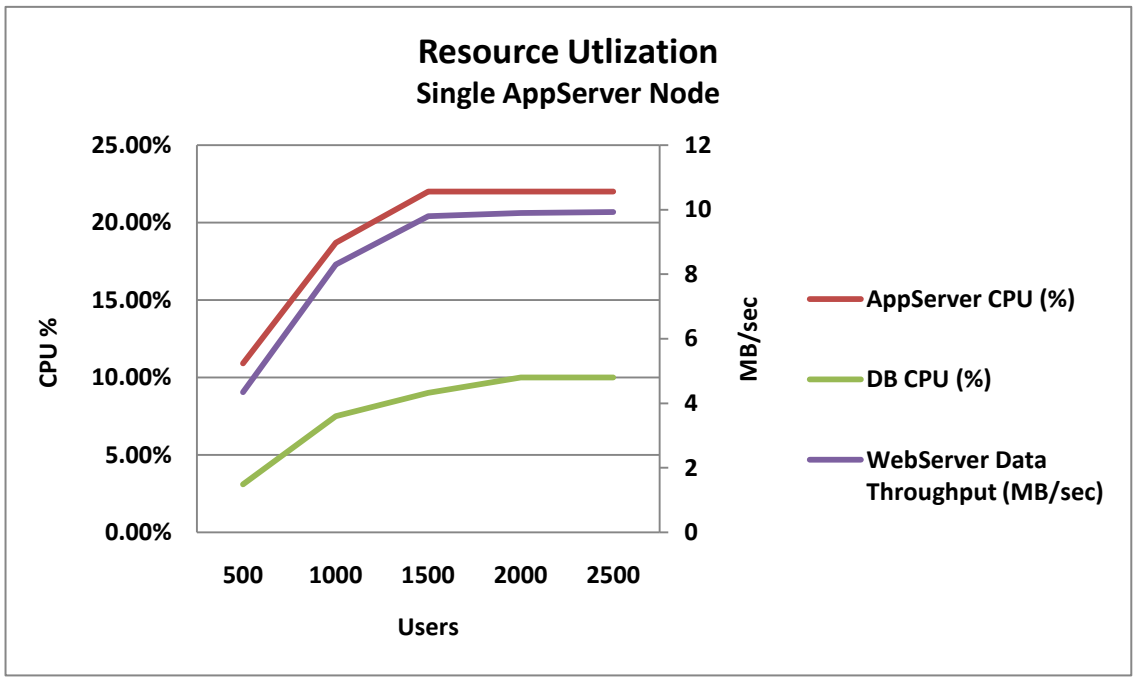
Figure 16. Response Time Variation



As per Figure 16, the applicatin achived sub-second response time for Buy and Quote transaction for a concurrent user load of around 1500 user, there upon the response time degrades with increased load on the application.

Figure 17 demonstrates the CPU utlization for AppServer and Database Tier and data throughput of WebServer Tier.

Figure 17. Resource Utilization

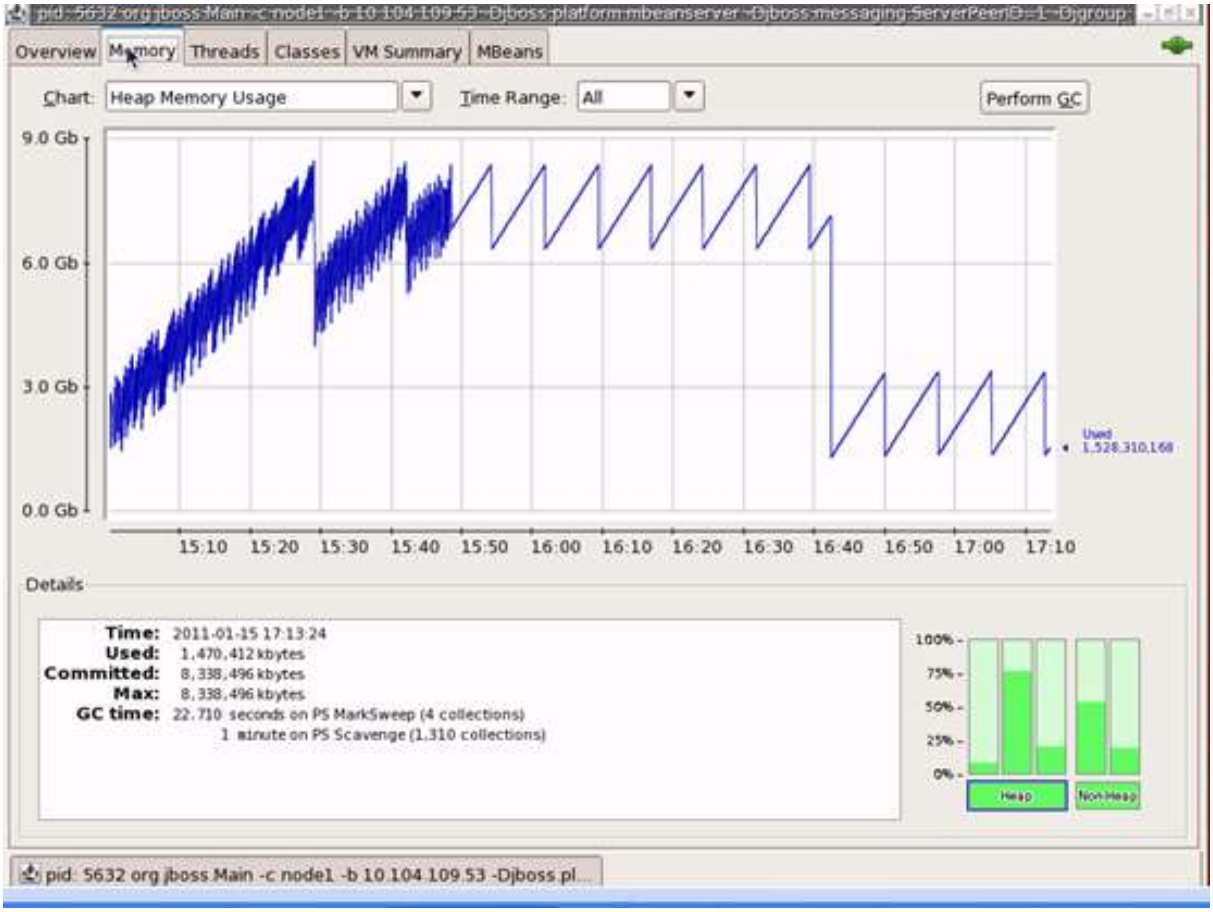


As demonstrated in previous illustrations, the application saturates at around 1500 concurrent user load . This is evident with the steady resource utlization of WebServer, AppServer and Database for a user

base of 1500 to 2500 concurrent users, thus demonstrating a scale well point of around 1500 concurrent users.

As described in this section, user throughput saturates at aroudn 550-600 transactions/sec and the response time drastically increases beyond 1500 concurrent user. As we are aware that themaximum heap size is restricted to 8G and as we increase the number of user hitting the applications, there is more heap memory utilized , as more objects are allocated over the fixed JVM heap size. This leads to exesive gabage collections cycles and also increase in grabage collection pause time. This is evident in subsequent Figure 18.

Figure 18. JConsole view for 2500 user – Single Node



In the above JConsole view of the JVM Heap Memory Usage, we can see after 15:20, the Full GC time interval reduces. Also the memory cleared through Full GC also reduces and the GC trigger more frequently. This causes lower application time and also a reason for reduced application response time. Please note , no nursery GC seen after 15:50 and drop in heap memory usage beyond 16:30 is due to the fact that test was stopped after 15:50.

Several ways can be deployed to reduce frequency of GC, thus increasing more application time.Some methods can be defined as follows

1. Reduction in allocation of temporary objects, for example, identifying code snippets wherein excessive temporary objects are instantiated. This process generally takes a lot of time, as it requires study of heap space through profiling and thereon working upon change code to reduce temporary obect allocation.

2. Increasing heap size, but this again tends to increase GC pause time due to large heap size.
3. Deploying caching solution such as open source ehcache or licensed caching solutions such as Coherance or Extreme Scale.
4. Addition of new instances through a clustred middleware deployment.

The fourth option, deploying additional application server instance in a clustered configuration, is easy to deploy and generally a widely accepted methodolgy to improve scalability of the web application. In the next section we have configured another application server instance to the same deployment and demonstrated a reduction in response time and improvement in overall application throughput

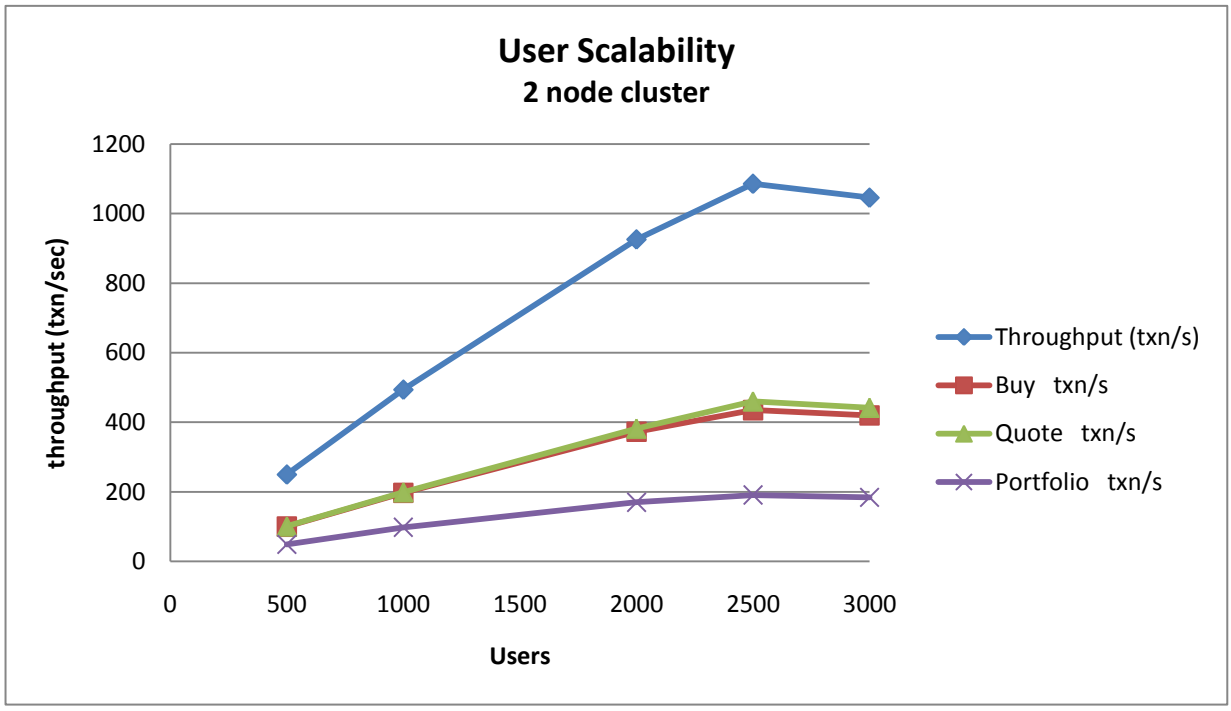
6.3.6 Performance Results for Multiple Application Server Instance

This section demonstrates the performance and scalability improvements while using AppServer cluster with multiple appServer instances. It defines the benefits of scaling out the application server instances to achieve increased throughput while sustaining the response time benchmarks achieved in single appserver instance configuration.

All test results described use the same load driver and application as used in Single node performance test executions.

Figure 19. illustrates the throughput achieved for a two node appserver cluster.

Figure 19. User Scalability for 2 Node Cluster



As depicted in Figure 19, the application achived a saturation point at around 2500 concurrent user scale.Beyond 2500 users, we can see a dip in throughput beyond 2500 user load we have hit the knee in the load curve and this would be evident in subsequent illustration of response time graph for 2 node cluster.

Figure 20. Response Time Variation for 2 Node Cluster

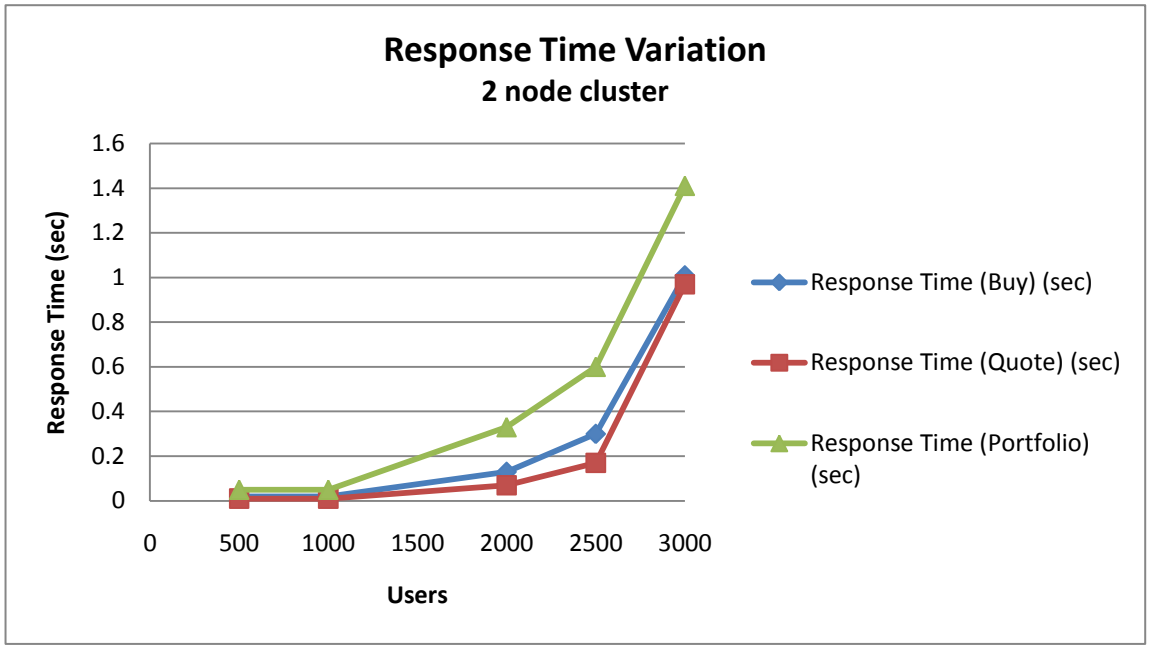
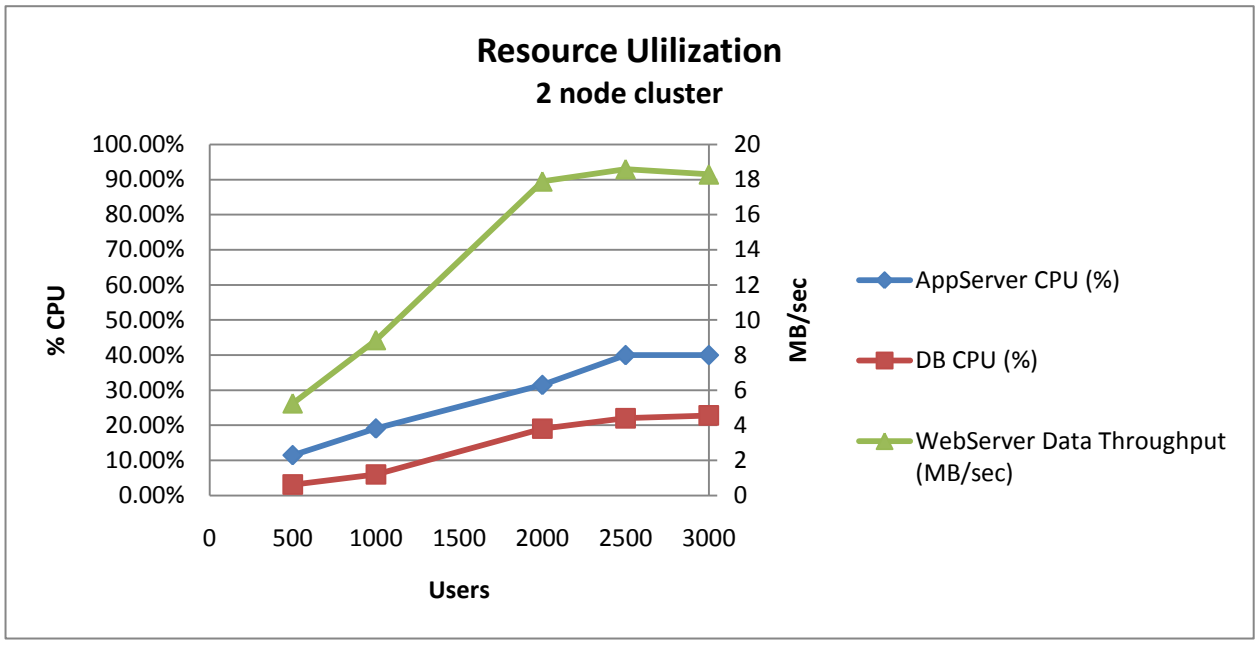


Figure 20 ilustrates response time variation of transactions executed in workload matrix. It is evident from above graph that beyond 2500 concurrent user load we can see high increase in response time; the two instance cluster solution can sustain a load of 2500 user with acceptable response time.

Figure 21demonstrates the CPU utlization for AppServer and Database Tier and data throughput of WebServer Tier for 2 node cluster solution

Figure 21. Resource Utilization for 2 Node Cluster



As demonstrated in previous illustrations, the application saturates at around 2500 concurrent user load for 2 node cluster solution. This is evident with the steady resource utlization of WebServer, AppServer and Database for a user base of 2500 to 3000 concurrent users.

As depicted in the two node cluster configuration we reach a saturation point or scale well point at around 2500 users. The reasons for this behavior is similar to those stated at the end of Section *6.3.5 Performance Results for Single Application Server Instance)*

Figure 21 and Figure 22 illustrate heap memory usage of node1 and node2 for a two node cluster.

Figure 22. JConsole View of node1 – 2 node cluster

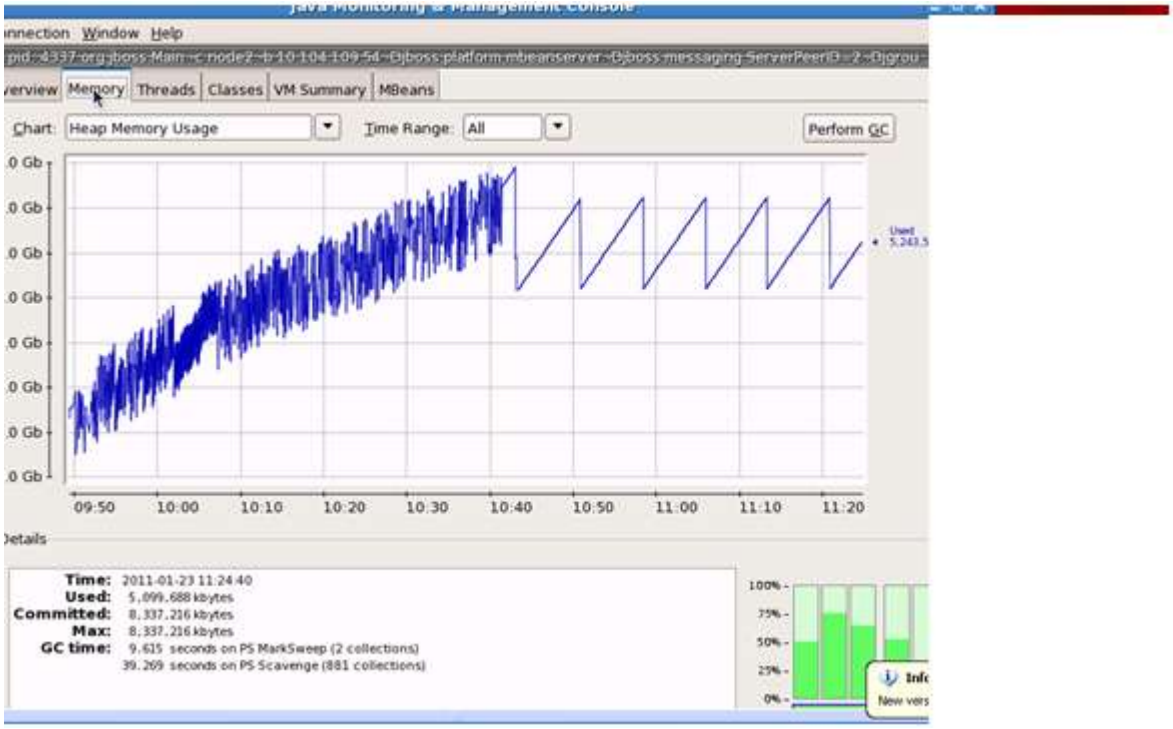
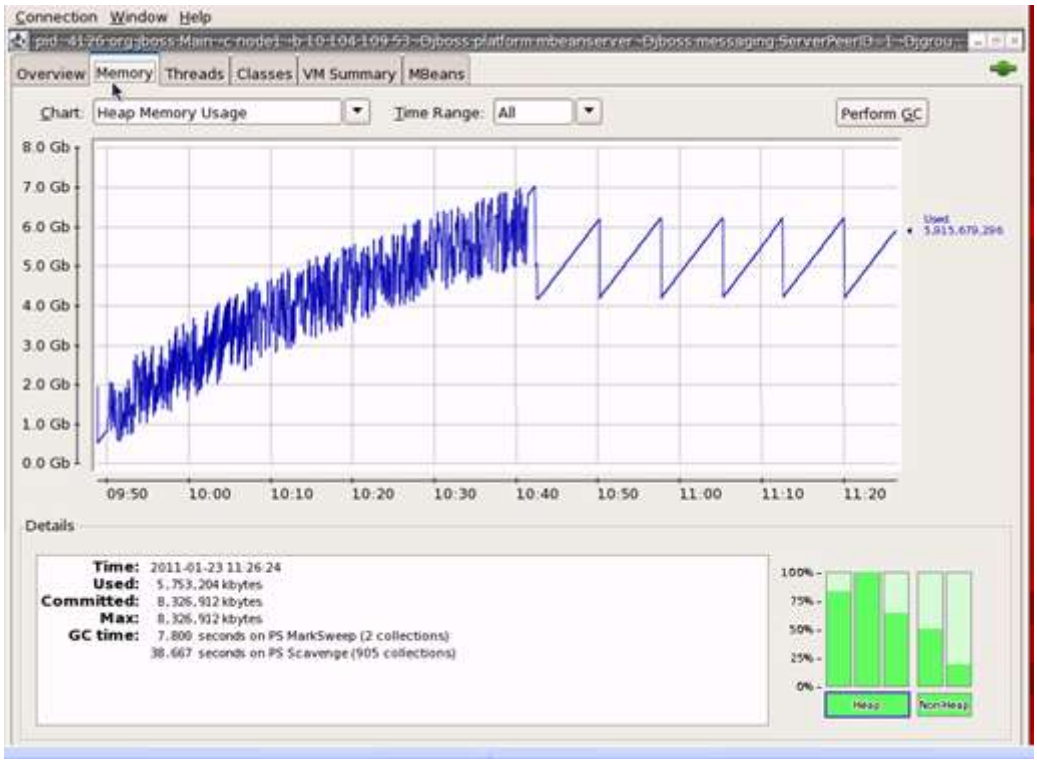


Figure 23. JConsole View of node2 – 2 node cluster



As shown in above illustration, we see similar behavior as we saw in single node heap memory usage view, but in two node cluster we achieve increased throughput and reduced response time. This is due to additional application server node added in JBoss cluster. Thus additional application server node in clustered configuration has helped to achieve a scalable solution, with higher utilization of large memory and computing power of UCS Blade B200 M2 server. Comparitive analysis between single node and multiple node configuratin is described in the next section.

6.3.7 Comparative Performance Analysis

This section ilustrates the performance benefits of single node application server as compared with 2 node appserver instance cluster.Performance is compared on two important performance indictors , application throughput and transaction response time of executed workload matrix.

As show in the subsequent graph, the saturation point for single node instance is around 600 txn/sec and around 1050 txn/sec for a two node cluster.It demonstrates around 75% increase in application throughput , thereby increasing the user scabaility from 1500 user to around 2500 concurrent users.Both the application server instances are deployed on a single Cisco UCS B200 M2 server.

Figure 24. User Scalability 1 Node vs. 2 Nodes

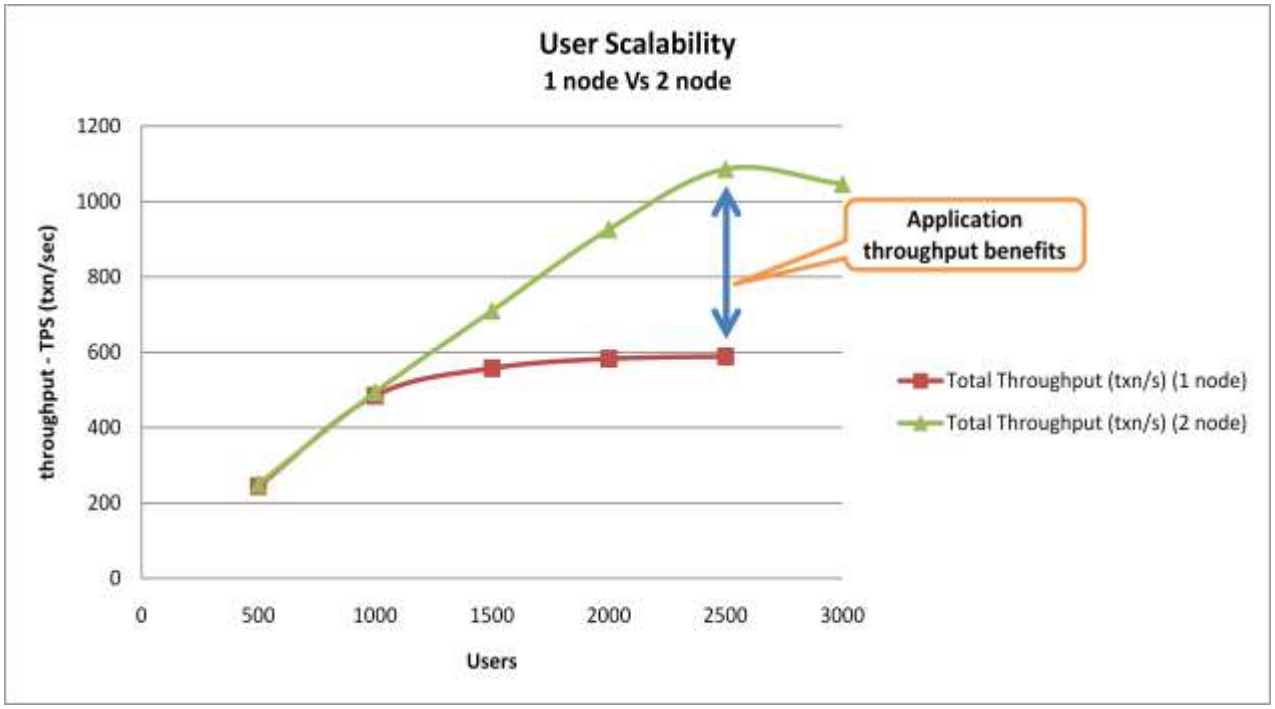


Figure 25 demonstrates the response time variation for one of the transaction defined in Workload Mix, for single node appserver and two node appserver cluster.

Figure 25. Response Time Variation 1 Node vs. 2 Nodes

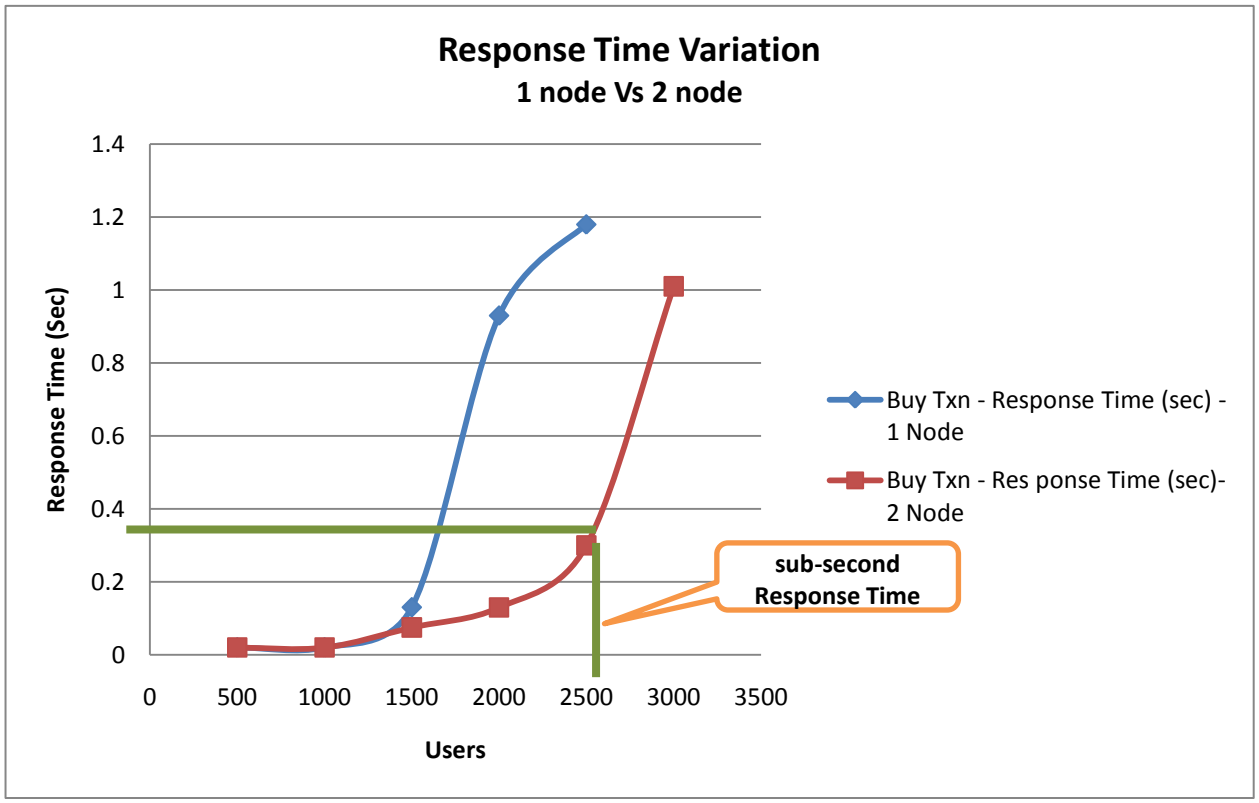
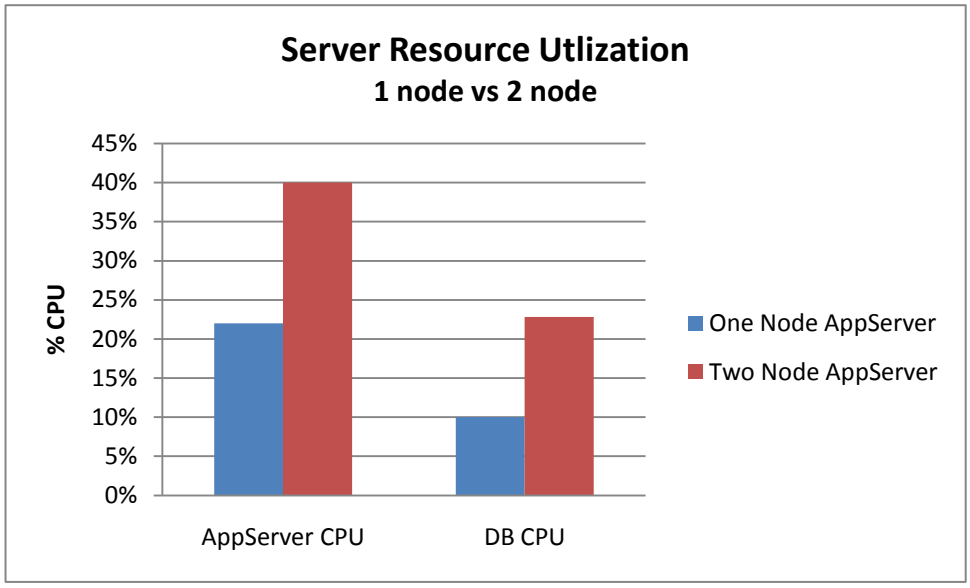


Figure 25 shows the sub-second response time achieved for single instance with around 1500 user can be sustained with 2500 concurrent user with the help of multiple appserver instances . Any increase beyond 2500 concurrent users for 2 node cluster, increases the response time drastically to more than a second.

Figure 26 illustrates the cpu utlization of 1 node and 2 node appserver for 1500 user and 2500 concurrent user scalability.

Figure 26. Server Resource Utilization 1 Node vs. 2 Nodes



As previously stated Application Server is deployed on UCS B200M2 blade server equipped with two six-core Intel Xeon 5680 processors at 3.33 GHz with a physcial memory of 24G. It demonstrates how we can use the processing power of B200M2 server to scale out the application in the same physical server. Two node application server instance uses around 40% of CPU with enough room to further scale the application or can be used during unprecedented peak application usage conditions.

Application Database Server is deployed on Cisco Full width blade sever – B250M2 which is equipped with with two six-core Intel Xeon 5680 processors at 3.33 GHz and configured with 96G of physical memory through the use of a Cisco Extended Memory Technology. The Database server utlilizes around 22% of server cpu , which again has a lot of bandwidth to support requests from additional Application server instances. It is equipped with a large physical memory which can easily be exploited in database cache for read-only transactions, such as portfolio transaction used in our present solution.

7. Open Source Three-Tier Application Design and Implementation for Small and Medium Architectures

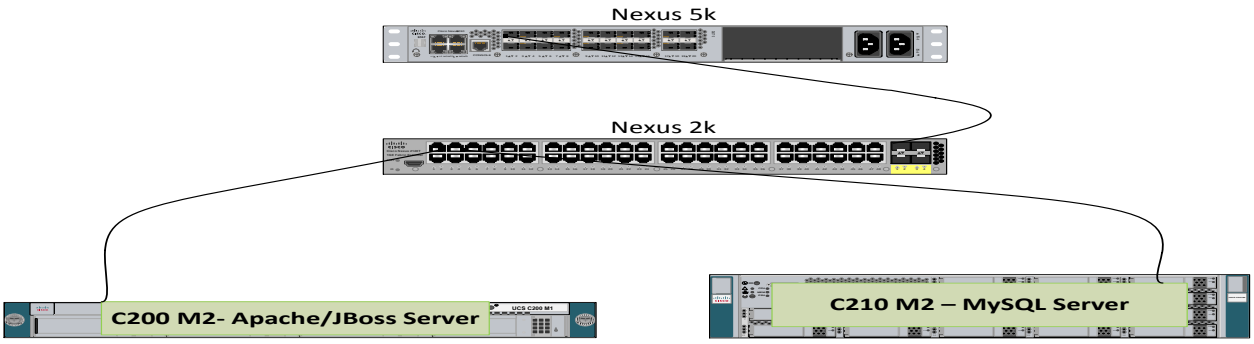
7.1 Deployment Architecture

The configuration presented in this document is based a typical three-tier web deployment which has the following main components.

WebServer	Apache 2.2 is deployed on C200M2 rack mount server equipped with two six-core Intel Xeon 5670 processors at 2.93 GHz with a physcial memory of 24 GB
Application Server	JBoss 5.1 is deployed on the same C200M2 rack mount server
Database	MySQL-server-5.5.8-1.rhel5.x86_64 is deployed on C210 M2 rack mount server which is equipped with with two six-core Intel Xeon 5670 processors at 2.93 GHz and configured with 48GB
Storage	Internal Hard Disk Disks, part of C210 rack mount server – 8x 73 GB SAS, 15K RPM, Small Form-Factor, hot pluggable hard drives enabled with RAID 10 for MySql data LUN (using MegaRAID controller)
Operating System (64 bit)	Red Hat Enterprise Linux Server release 5.5

The latest UCS manager release (version 1.4) allows to manage the rack mount servers along with the other blade servers deployed over chassis (UCS 5108), connecting to the same fabric interconnect to facilitate unified management across rack mount servers and blade servers. This is in addition to existing option of stand-alone management of rack mount servers using integrated CIMC controller.

Figure 27. Deployment Architecture for Small and Medium Implementations



7.2 Installation and Configuration

Apache, JBoss and MySQL are installed on RHEL5.5 using UCS C-Series servers. Some of the important installation and configuration procedures are detailed in subsequent sections.Each section is sub-divided into two important steps; install the server and configure and tune the server.

7.2.1 Apache

Apache 2.2 is installed on Cisco C200 M2 server powered with two six-core Intel Xeon 5670 processors at 2.93 GHz and 24 GB of memory. Unlike the previous setup (refer section 6.2.1 Apache), Apache is configured without being a load balancer.

```
Install Apache:
        1) Download, Build, and Install Apache 2.0 from apache.org.
        2) tar -xzvf httpd-2.2.17.tar.gz
        3) ./configure --enable-so --enable-deflate --enable-proxy --enable-proxy-http --enable-proxy-
connect --enable-proxy-ajp --enable-proxy-balancer  --with-mpm=worker
        4) Make
        5) Make install
        6) Ensure Apache functions properly by starting, testing, and stopping it.
             /usr/local/apache2/bin/apachectl start
             Browse through http://localhost
            /usr/local/apache2/bin/apachectl stop

Configure and Tune Server:
        1) Edit httpd.conf to include mod_proxy and mode_proxy_http modules

        2)  Add Balancer members configuration which incljudes load balacing algorithm
            <Proxy balancer://mycluster>
                Order deny,allow
                Allow from all
                BalancerMember http://10.104.110.14:8009/ route=node1 keepalive=On
                #BalancerMember http://XX.XXX.XXX.XX:8080/ route=node2 keepalive=On
                ProxySet stickysession=JSESSIONID
                #ProxySet lbmethod=byrequests
                #ProxySet nofailover=off
            </Proxy>
        4) Add virtual Host configuration
            <VirtualHost *:80>
                ServerName 10.104.110.14
                ProxyRequests Off
                <Location /daytrader>
                        ProxyPass balancer://mycluster/daytrader
                        ProxyPassReverseCookiePath /daytrader /
                </Location>
            </VirtualHost>
        5)Define and Tune MPM (Multi-Processing Module)
            Use worker MPM module.
            Edit ThreadsPerChild and MaxClients directives under worker.c module.
                ThreadsPerChild=50
                MaxClients=10000
```

7.2.2 JBoss Application Server

JBoss 5.1 is installed on the same server as Apache which is a C200 M2 server powered with two six-core Intel Xeon 5670 processors and 24 GB of memory.JBoss Server handles the incoming request from Apache and processes various transaction issues from DayTrader End User.

Some important configuration steps to deploy JBoss and connectivity with MySQL database server are described as follows. This is a single instance JBoss implementation unlike JBoss cluster configuration done in section 6.2.2 JBoss Application Server.

```
Define JBoss cluster instance configuration
        1)Edit server.xml to define cluster node name as in the startup paramters
            Copy $JBoss_Home/server/all to $JBoss_Home/server/nodetemplate
            Edit server.xml and add
                <Engine name="jboss.web" defaultHost="localhost"
jvmRoute="node${jboss.messaging.ServerPeerID}">
            jboss.messaging.ServerPeerID would be provided in the startup paramters
        2)   Add/Edit HTTP Connector details
            <!-- A HTTP/1.1 Connector on port 8080 -->
            <Connector protocol="HTTP/1.1" port="8080" address="${jboss.bind.address}"
connectionTimeout="20000" redirectPort="8443" maxThreads="2000" enableLookups="false"
acceptCount="100" />
        3) Define mysql-persistence-service.xml
            Delete $JBoss_Home/server/nodetemplate/deploy/messaging/hsqldb-persistence-service.xml
            Copy $JBoss_Home /docs/examples/jms/mysql-persistence-service.xml to
             $JBoss_Home/server/nodetemplate/deploy/messaging directory
        4)  Edit DefaultDS datasource under mysql-ds.xml  to point to MySQL
        5) Copy mysql-connector-java-5.1.13-bin.jar to $JBoss_Home/server/nodetemplate/lib
        6) Copy nodetemplate directory as node1 directoy .


OS tuning
        Enabled hugepages on the server
            vm.nr_hugepages =  8192

Configure SUN JDK1.6 parameters
         Edit run.conf under $JBoss_Home for JVM tuning
            JAVA_OPTS="-Xmn2G -Xms8G -Xmx8G -Xss256k -XX:MaxPermSize=512m -XX:SurvivorRatio=6"
            JAVA_OPTS="$JAVA_OPTS -XX:+AggressiveOpts -XX:+AggressiveHeap -XX:+UseParallelGC
                -XX:CompileThreshold=10000 -XX:ReservedCodeCacheSize=128m -XX:+UseLargePages
                -XX:+PrintGCDetails "

Sever StartupOptions for each node
        ./bin/run.sh  -c node1 -b 10.104.110.14 -Djboss.platform.mbeanserver
    -Djboss.messaging.ServerPeerID=1 -Djgroups.bind_addr=127.0.0.1  -Djava.net.preferIPv4Stack=true &
```

7.2.3 MySQL

MySQL-server-5.5.8-1.rhel5.x86_64 is deployed on Cisco C210 M2 rack mount server which is equipped with with two six-core Intel Xeon 5670 processors at 2.93 GHz with 48GB of physical memory, the data files are hosted on RAID 10 (using the LSI MegaRAID controller) conconfigured on the local hard disks.

MySQL is configured to have INNODB as the default storage engine. Some important stepts in installation and configuration of MySql are defined below.

```
Install & Configure MySQL5
    1) rpm –ivh MySQL-server-5.5.8-1.rhel5.x86_64.rpm
    2) Edit my.cnf to enable InnoDB as default storage engine
    3) Edit important tuning parameters under my.cnf
        Max_connections = 2000
        key_buffer_size = 24G
        table_open_cache = 8192
        thread_cache_size = 512
        query_cache_size= 512M
        thread_concurrency = 24
Enable slow query logging in my.cnf
        log-slow-queries=/var/log/mysqlslowqueries.log
        long_query_time = 3
```

The above configurations are based upon a estimates for 2000 concurrent users using DayTrader Application.

7.3 Performance Analysis

7.3.1 Objective

As we have evaluated the performance of Cisco UCS Blade servers before, this section evaluates the performance of a Daytrader application deployed on Cisco UCS C-Series (Rack Mount Platform) servers using open-source technologies. The study is intended to represent small deployments and hence restricted to single node JBoss instance.

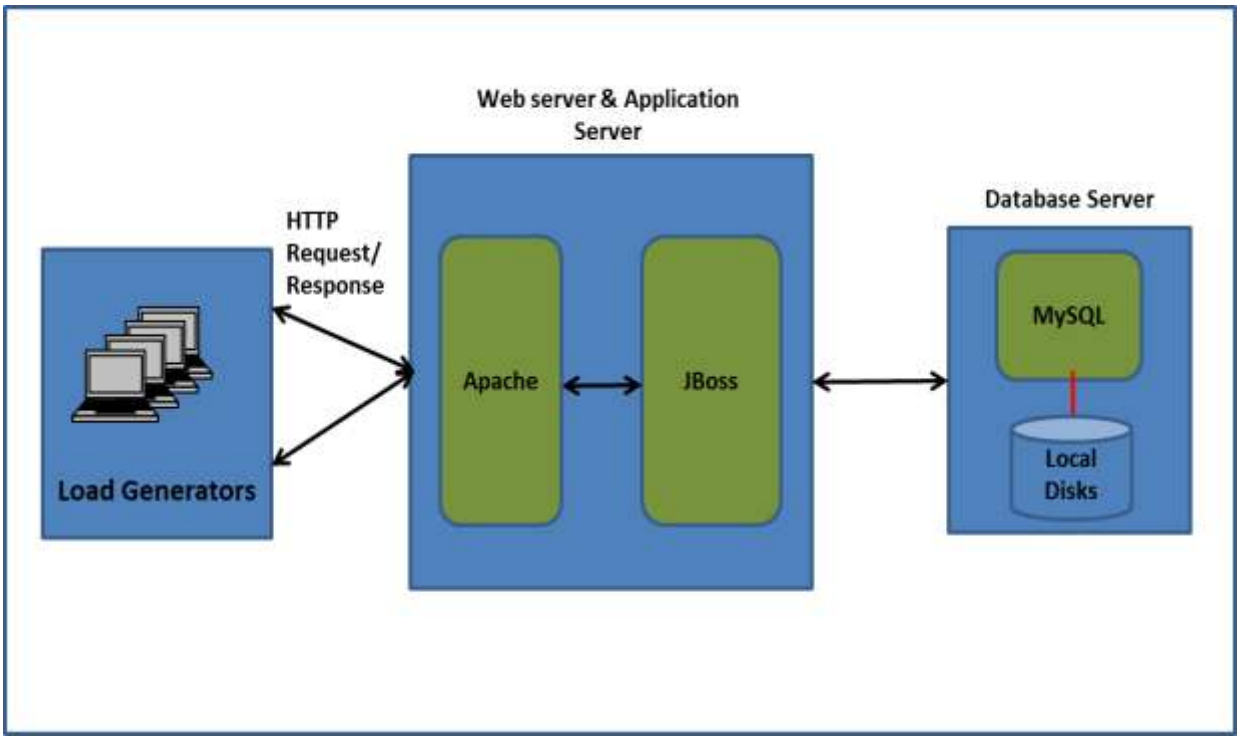
This study evaluated the Performance and Scalability of the application on these important critierias.

- Throughput–Maximum transaction/sec achieved from the deployed application with the condition of acceptable application response time or saturation of availble system resources
- Response Time–Time taken to execute deployed application transaction. In present solution we would be measuring response time for Buy, Quote and Portfolio transactions

7.3.2 Benchmark Configuration

The setup used for the above mentioned performance analysis is illustrated in the following figure .

Figure 28. Performance Benchmark Setup



7.3.3 Workload Mix

For realistic load simulation, test scripts are modeled after DayTrader business processes (or called transactions) – Buy, Portfolio and Quote. These transactions are simulated with independent load driver threads ensuring unique users for each type of transaction.The percentage of usersa didicated to each transactions were as follows

- 40% of users executed Buy transaction
- 40% of users executed Quote transaction
- 20% of users executed Portfolio transaction

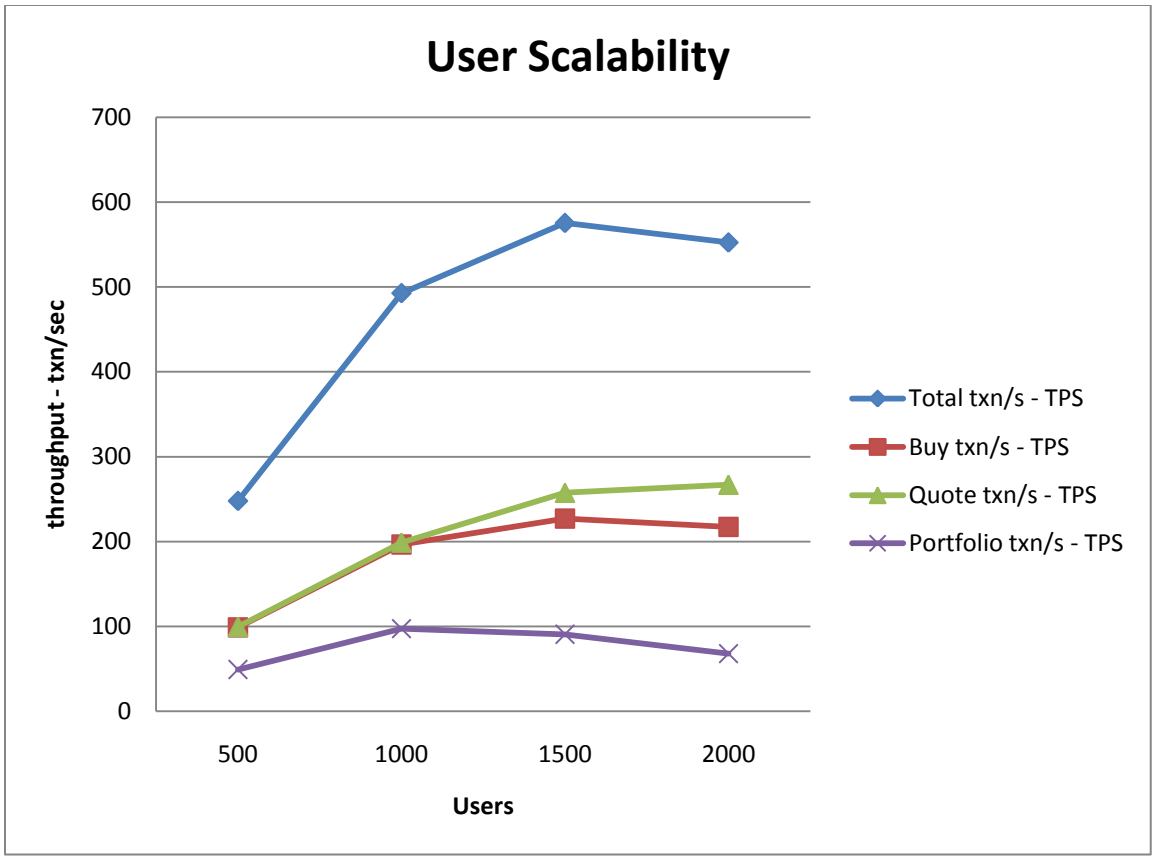
The run-time mode chosen for DayTrader application configuration (run time modes are i) Full EJB3, ii) Direct (JDBC) and iii) Session (EJB3) To Direct mode.The run-time mode for the executed benchmark described in subsequent result sections is configured with Full EJB3 using Hibernate as the JPA layer.

The test were executed with 500 to 2000 concurrent user load on single node application server.In order to compare with a real work scenario workload the database was populated with around 3 million users and 100 million quotes.

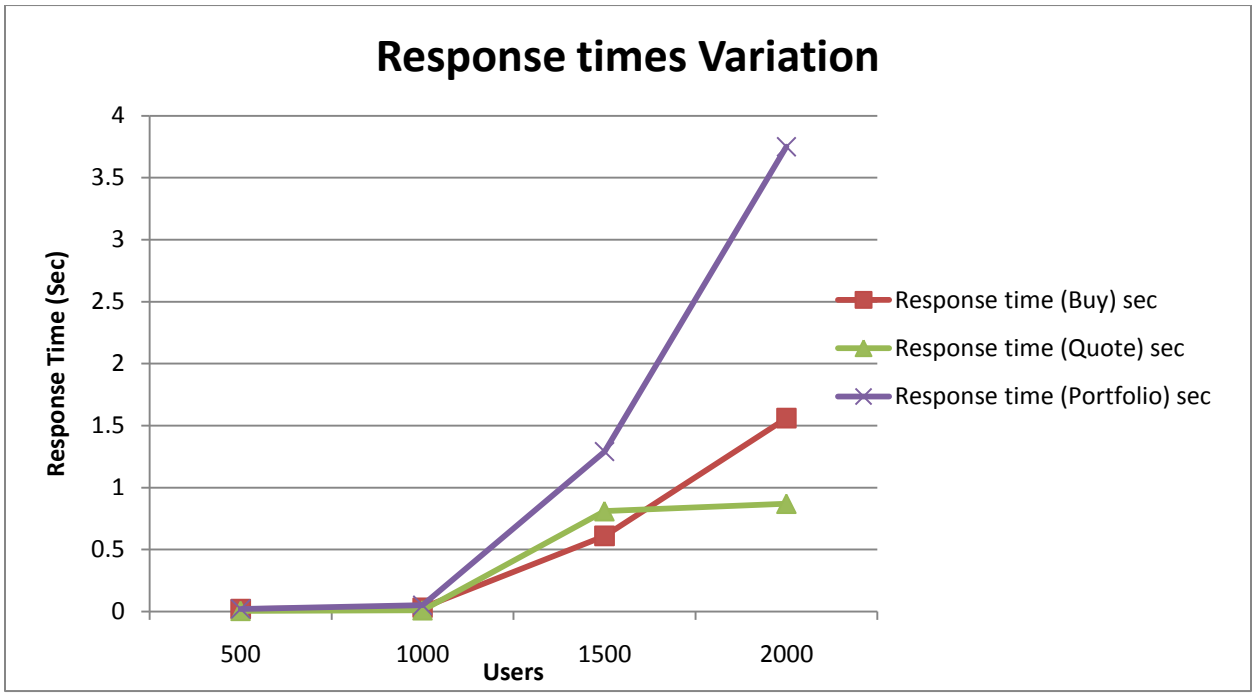
7.3.4 Performance Results

Each of the benchmark tests were executed for 50 minutes duration, keeping the threshold of transaction response times not to exceed 5 seconds.

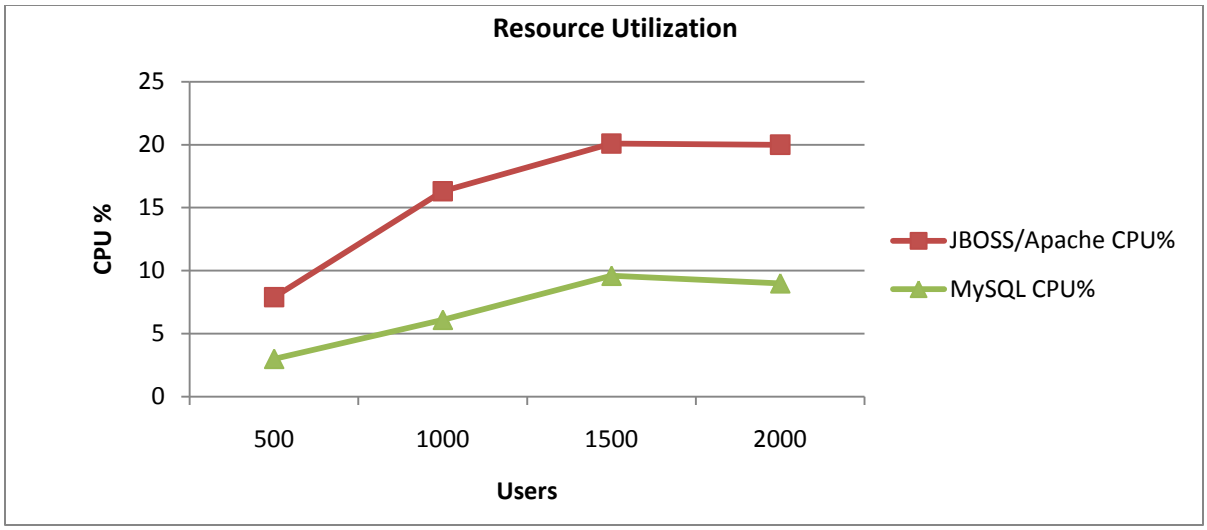
Following graph illustrates the throughput (no. of transactions completed within the test duration) behavior when load is increased from 500 to 2000.



The following graphs illustrate the behavior of the response times (during steady state of the test) when the load in increased from of 500, 1000, 1500 and 2000 users.



The CPU Utilization of JBoss/Apache (co-hosted on a single machine) and MySql DB for the test executions is captured below:



From the above graphs, it is clear that the throughput is saturated around 1500 users and response times are increasing around the same users, for single Application Server instance. The CPU utilization remains low (less than 20% on both Application and database servers); which indicates that there could be a potential scalability issue with the application. This can be mitigated by adding application server nodes, as demonstrated in Large Scale implementation.(Section 6.3.6)

8. Extension of Study to Real-World Scenarios

8.1 Virtualized Implementation

In the present benchmark, we have installed Linux over bare metal, but several enterprise web deployments are using virtualization technology, to optimize their IT landscape and reduce their hardware and operating costs. Three-tier web deployment is a good candidate for virtualization as dynamic allocation in virtualized environment allows easier provisioning and optimized system performance. Workloads are consolidated with underutilized server machines loaded on a smaller number of fully utilized machines. Fewer physical machines can lead to reduced costs through lower hardware, energy and management overhead. Several mid-tier solutions provide Hypervisor Edition which enable optimized execution of application servers over virtualized environment. These solutions can effectively utilize the full potential of compute, network, virtualization and management capabilities of Cisco Unified Computing System.

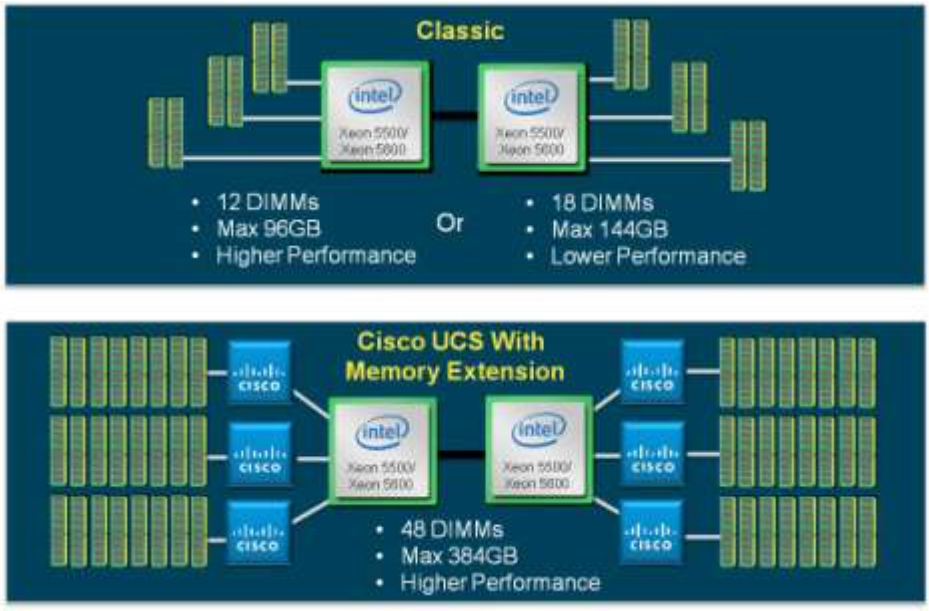
8.2 Extended Memory and Caching

A large number of Enterprise web deployments are utlizing in-memory data grid for clustered application servers.This is incorporated, in response to every increasing demand to provide fast access to web applications , for tens of thousands of active users. In-memory data grid solutions, can be executed in several different modes, such as Level-2 Cache, read-only cache, read/write cache as well as caching entire business application in main memory.This provides low end user response time , avoiding data base acess.

Caching solutions require high density servers, but modern CPUs with built-in memory controllers support a limited number of memory channels and slots per CPU. To obtain a larger memory footprint, most web application administrators are forced to upgrade to larger, more-expensive four-socket servers. CPUs that can support four-socket configurations often cost more, require more power, and entail higher licensing costs. Cisco Extended Memory Technology expands the capabilities of CPU-based memory controllers by logically changing the main memory while still using standard DDR3 memory. The technology makes

every four DIMM slots in the expanded-memory blade server appear to the CPU's memory controller as a single DIMM that is four times the size. For example, using standard DDR3 DIMMs, the technology makes four 8-GB DIMMS appear as a single 32-GB DIMM (Figure 29).

Figure 29. Cisco Extended Memory Architecture



Cisco Extended Memory Technology provides flexibility between memory cost and density. This extended memory uses a high-performance, ultra-fast technology that is implemented in its ASIC to allow 48 memory modules (DIMMs) to be addressed at high speed. The total memory address space per blade increases to 384 GB at 1333 MHz compared to 96 GB at 1333 MHz, or 144 GB at 800 MHz, on alternative hardware provided by other Intel based two-socket server vendors, which can use up to 18 memory modules (DIMMs).

Thus Cisco UCS blade and rack-mount servers, with Cisco Extended Memory technology can easily benefit middleware caching solutions providing a balanced ratio of CPU power to memory and install larger amounts of memory to make the most of compute resources.

.

9. Conclusion

Designed using a new and innovative approach to improve data center infrastructure, the Cisco Unified Computing System unites compute, network, storage access, and virtualization resources into a scalable, modular architecture that is managed as a single system.

The workload for the three-tier performance test included a Benchmark application, which characterizes performance of J2EE Application Servers. The deployed application demonstrates a real-world like scenario of Buy, quote and view portfolio transactions with a mix of 2:2:1, which is generally used in typical stock trading applications using Java technologies. It characterizes the open source servers, like Apache, JBoss and MySQL which can easily be used in an application providing sub-second response time with the capabilities of clustering and failover.

The performance analysis stated in previous section, illustrate how UCS Blade and rack mount server platforms can be used for open source web, application and database servers with single or multiple instances of application server deployed in a cluster. With the stateless nature of UCS, it provides rapid stand-up of the web server and application server hardware, further helping in imitating similar configuration profiles for staging and production environment. The performance results demonstrate the computing power of UCS Blade and rack mount servers, for instance, Blade Server platform (two node application server configuration) gives an output of around 1050 transactions/sec i.e. around 60,000 transactions per minute with sub-second response time, utilizing just around 40% CPU utilization at the application tier. It demonstrates the performance gains vis-à-vis application throughput and end-user response time achieved through a clustered configuration.

This paper has demonstrated a methodology to deploy open source three-tier architecture on Cisco UCS servers, especially on how to scale mid-tier application server from single-node to multiple-node configuration in the UCS environment, thus achieving a scalable web application deployment.

10. For More Information

http://www.cisco.com/en/US/products/ps10280/index.html

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **www.cisco.com/go/offices.**

Printed in USA

Document number: UCS-TR100031 02/11