RETRIEVAL OF FILES FROM CLUSTER BASED WEB SERVER

*Vamsi Thotakura,
*JDA Software, Hyderabad., Andhra Pradesh

Introduction

State-of-the-art cluster-based data centers consisting of three tiers (Web server, application server, and database server) are being used to host complex Web services such as e-commerce applications. The application server handles dynamic and sensitive Web contents that need protection from eavesdropping, tampering, and forgery. Although the Secure Socket Layer is the most popular protocol to provide a secure channel between a client and a cluster-based network server, its high overhead degrades the server performance considerably and, thus, affects the server scalability. We propose a back-end forwarding scheme, called SSL_with_bf that employs a low-overhead user-level communication mechanism like Virtual Interface Architecture to achieve a good load balance among server nodes. We compare three distribution models for network servers, Round Robin, SSL_with_Session, and SSL_with_bf, through simulation. The experimental results with 16-node and 32-node cluster configurations show that, although the session reuse of SSL_with_Session is critical to improve the performance of application servers, the proposed back-end forwarding scheme can further enhance the performance due to better load balancing. The SSL_with_bf scheme can minimize the average latency by about 40 percent and improve throughput across a variety of workloads.

OVERVIEW OF THE SYSTEM

Due to the growing popularity of the Internet, data centers/network servers are anticipated to be the bottleneck in hosting network-based services, even though the network bandwidth continues to increase faster than the server capacity. It has been observed that network servers contribute to approximately 40 percent of the...
overall delay, and this delay is likely to grow with the increasing use of dynamic Web contents. For Web-based applications, a poor response time has significant financial implications. For example, E-Biz reported about $1.9 billion loss in revenue in 1998 due to the long response time resulting from the SSL, which is commonly used for secure communication between clients and Web servers.

Even though SSL is the de facto standard for transport layer security, its high overhead and poor scalability are two major problems in designing secure large-scale network servers. Deployment of SSL can decrease a server’s capacity by up to two orders of magnitude. In addition, the overhead of SSL becomes even more severe in application servers. Application servers provide dynamic contents and the contents require secure mechanisms for protection.

LITERATURE REVIEW

EXISTING SYSTEM

In existing system, they have used to develop the project using Round Robin [RR] model and SSL_with_Session model. Those models are not effective. Those models are not able to give the output in time and the throughput also lesser than their expected output. These models had made the Latency problem and minimal throughput. For this problem, they introduced the SSL_with_bf model is to overcome the existing problems. We are going to implement SSL_with_bf model in our proposed system.

PROPOSED SYSTEM

In our proposed system, we are going to implement the SSL_with_Backend Forwarding model is to overcome the problem of existing system. This model will reduce the latency and increase the throughput than the existing system (Round
Robin model and SSL_with_Session). The SSL_with_bf model is very helpful for load balancing of the server.

This will reduce the load of the server while the server is being busy. These are the advantages of our proposed system. The SSL_with_bf scheme can minimize the average latency by about 40 percent and improve throughput across a variety of workloads.

**PLATFORM**

**SOFTWARE SPECIFICATIONS**

- **Operating System**: Windows XP
- **Language**: VB .NET 2005
- **Database**: SQL SERVER 2000

**SOFTWARE DESCRIPTION**

**FEATURES OF .NET**

- Visual Studio .Net has flexibility, allowing one or more language to interoperate to provide the solution. This Cross Language Compatibility allows to do project at faster rate.
- Visual Studio .Net has Common Language Runtime, which allows the entire component to converge into one intermediate format and then can interact.
- Visual Studio .Net has provide excellent security when your application is executed in the system
- Visual Studio .Net has flexibility, allowing us to configure the working environment to best suit our individual style. We can choose between a
single and multiple document interfaces, and we can adjust the size and positioning of the various IDE elements.

- Visual Studio .Net has Intelligence feature that make the coding easy and also dynamic help provides very less coding time.
- The working environment in Visual Studio .Net is often referred to as Integrated Development Environment because it integrates many different functions such as design, editing, compiling and debugging within a common environment. In most traditional development tools, each of separate program, each with its own interface.

**SYSTEM ANALYSIS**

We investigate the impact of SSL offering in cluster-based network servers, focusing on application servers, which mainly provide dynamic content. We show the possible performance improvement when the SSL-session reuse scheme is utilized in cluster-based servers. The SSL-session reuse scheme has been tested on a single Web server node and extended to a cluster system that consisted of three Web servers. We explore the SSL-session reuse scheme using 16-node and 32-node cluster systems with various levels of workload. We propose a back-end forwarding mechanism by exploiting the low-overhead user-level communication to enhance the SSL-enabled network server performance.

We compare three distribution models in clusters: Round Robin, SSL_with_Session, and SSL_with_bf. The RR model, widely used in Web clusters, distributes requests from clients to servers using the RR scheme. SSL_with_Session uses a more sophisticated distribution algorithm in which subsequent requests of the same client are forwarded to the same server, avoiding expensive SSL setup costs. The proposed SSL_with_bf uses the same distribution policy as the SSL_with_Session, but
includes an intelligent load balancing scheme that forwards client requests from a heavily loaded back-end node to a lightly loaded node to improve the utilization across all nodes. This policy uses the underlying user-level communication for fast communication.

SYSTEM DESCRIPTION

MODULES

1. Authentication Module.
2. IP Address Representation Module.

MODULES DESCRIPTION

AUTHENTICATION MODULE

This module is to register the new users and previously registered users can enter into our project. The admin only can enter and do the uploading files into the servers.

IP ADDRESS REPRESENTATION MODULE

This module is to give the IP addresses which we are going to assign those as servers. We can enter and view IP addresses from this module.

LOAD SERVER MODULE

In this module, the administrator only can enter into this module. The administrator will do the encryption of the text file and store into the servers which we are assigned in IP representation module. This module will make the both public and private key for the cryptography.

LOAD BALANCING MODULE
The users can enter into this module and can view the file name in which the administrator stored into the servers. The user can select the file from the list and can download from the server which is in idle state. We will get the response time from which server we are getting the file. Finally we can get the decrypted file from the key pair.

**SYSTEM TESTING**

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works. The final step involves Validation testing, which determines whether the software function as the user expected. The end-user rather than the system developer conduct this test most software developers as a process called “Alpha and Beta test” to uncover that only the end user seems able to find.

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. Inadequate testing or contesting leads to errors hence it should be rectified.

Another reason for system testing is its utility as user-oriented granite before implementation. The best program is worthless if it does not meet user needs. Unfortunately, the users demand is often compromised by efforts to facilitate program or design efficiently in terms of processing time or memory utilization.

The proper choice of test data is an important as the test itself. If test data as input are not valid or representative of the data to be provided by the user, then the reliability of the output is suspect.

Test data may be artificial. Properly created artificial data should provide all combinations of values and formats and make it possible to test all logic and transaction path subroutines.

A program represents the logical elements of a system. For a program to run satisfactorily, it must compile and test data correctly and tie in properly with other
programs. Achieving an error-free program is the responsibility of the programmer. Program testing checks for two types of errors: syntax and logic.

When a program is tested, the actual output with the expected output. When there is discrepancy, the sequence of instructions must be traced to determine the problem. Breaking the program down into self-contained portions, each of which can be checked at certain key points, facilitates the process. The idea is to compare program values against desk-calculated values to isolate the problem.

Testing is an important stage in the System Development Life Cycle (SDLC). The test case is a set of data that a system will process as normal input. As its philosophy behind testing is to find errors the data are created with the express intent of determining whether the system will process them correctly.

Software testing is an important element of software quality assurance and represents the ultimate of specification, design and loading. The increasing visibility of software AR a system element and the costs associated with a software failure are motivating for well planned through testing.

No system design is ever perfect communication problems, programmers negligence are time constraints create error that must be eliminated before the system is ready for user acceptance testing. Following system testing is acceptance testing or running the system with live data by the actual user.

The testing of the system is a means of assessing or measuring the system to determine quality. System testing makes a logical assumption that if all parts of the system are correct, the system will successfully be implemented. Inadequate testing or non-testing leads to errors that may not appear until months later. This creates the time lag between the cause and appearance of the problem and the effects of the system errors on files and records within the system. Hence, the aim of testing is to create bug free reliable and secured system. The implementation of newly designed package is an important phase in adopting a successful new system the implementation of the package involves testing, user training.

Acceptance and changeover the objective of testing is to discover errors. To fulfill this objectives a series of test steps-viz.-unit test, integration, validations and system test were planned and executed. A program may function perfectly in isolation but fall when interfaced with other modules. The approach is to test entity wish
successively large ones, up to the system level. The test data were collected from the concerned and the live data were used to test data.

CONCLUSION AND FUTURE ENHANCEMENT

We investigated the performance implications of the SSL protocol for providing a secure service in a cluster-based application server and proposed a back-end forwarding scheme for improving server performance through a better load balance. The proposed SSL_with_bf scheme exploits the underlying user-level communication in order to minimize the intra cluster communication overhead. We compared three application server models, RR, SSL_with_Session, and SSL_with_bf, through simulation. The simulation model captures the VIA communication characteristics and the application server design in sufficient detail and uses realistic numbers for SSL encryption overheads obtained from measurements.

Simulation with 16-node and 32-node cluster configurations with a variety of workloads provides the following conclusions: First, schemes with reusable sessions, deployed in the SSL_with_Session and SSL_with_bf models, are essential for minimizing the SSL overhead. Second, the average latency can be reduced by about 40 percent with the ssl_with_bf model compared to the SSL_with_Session model, resulting in improved throughput. Third, SSL_with_bf yields a better performance with the mixed clients, whereas the performance of the SSL_with_Session model is degraded due to the increasing skew ness. Finally, SSL_with_bf is more robust than SSL_with_Session in handling variable file sizes.

All of these results indicate that the proposed back-end forwarding scheme is a viable mechanism for improving the performance of secure cluster-based network servers.

REFERENCES


