SSL Backend Forwarding Scheme in Cluster-Based Web Servers

K. Karthick, P. Jennifer and A. Muthukumaravel

Department of MCA, Bharath University, Selaiyur, Chennai-73, Tamil Nadu, India

Abstract:

- 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 Sockets 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.
- It improving the performance of SSL-enabled network servers is critical for designing scalable and high-performance data centers. We examine the impact of SSL offering and SSL-session-aware distribution in cluster-based network servers.
- 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.

Key wards: Web services such as e-commerce, Protection from eavesdropping, Tampering and forgery

INTRODUCTION

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 [1-7].

For example, E-Biz reported about $1.9 billion loss in revenue in 1998 due to the long response time resulting from the Secure Sockets Layer (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. Generating dynamic content takes about 100 to 1,000 times longer than simply reading static content. Moreover, since static content is seldom updated, it can be easily cached. Several efficient caching algorithms have been proposed to reduce latency and increase throughput of front-end Web services [5-8]. However, because dynamic content is generated during the execution of a program, caching dynamic content is not an efficient option like caching static content. Recently, a multitude of network services have been designed and evaluated using cluster platforms. Specifically, the design of distributed Web servers has been a major research thrust to improve the throughput and response time [9]. It is the first Web server model that exploits user-level communication in a cluster-based Web server. Our previous work reduces the response time in a cluster-based Web server using co scheduling schemes. In this paper, first, we investigate the impact of SSL offering in cluster-based network servers, focusing on application servers [10], which mainly provide dynamic content. Second, 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. In this paper, we explore the SSL-session reuse scheme using 16-node and 32-node cluster systems with various levels of workload. Third, we propose a back-end forwarding mechanism by exploiting the low-overhead user-level communication to enhance the SSL-enabled network server performance.

XYZ To this end, we compare three distribution models in clusters: Round Robin (RR), ssl_with_session and ssl_with_bf (backend_forwarding). 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 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. Extensive performance analyses with various workload and system configurations are summarized as follows: First, schemes with reusable sessions [11], deployed in the ssl_with_session and ssl_with_bf models, are essential to minimize the SSL overhead. Second, the average latency can be reduced by 40 percent with the proposed ssl_with_bf model compared to the ssl_with_session model, resulting in improved throughput. Third, the proposed scheme provides high utilization and better load balance across all nodes. The rest of this paper is organized as follows: a brief overview of cluster-based network servers, user-level communication and SSL is provided. Section 3 outlines three distribution models, including our proposed SSL back-end forwarding scheme.

Description of the Problem

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 [12]. Those models are not able to give the output in time and the throughput also lesser than that their expected output.
- These models had made the Latency problem and minimal through put. For this problem they introduced the SSL_with_bf (Backend forwarding) model is to overcome the existing problems. We going to implement SSL_with_Backend Forwarding model in our proposed system.

Proposed System:

- In our Proposed System, We are going to implement the SSL_with_Backend Forwarding model (Algorithm) 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 Secure Socket Layer_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.

Module Descriptions

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 Servers Module**: This module is, 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**: This module is, the users can enter into this module and can view the file name 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 and from which server we are getting the file [13]. Finally we can get the decrypted file from the key pair.

**The SSL protocol**:

![SSL Protocol Diagram]

**Implementation**: Implementation is the most crucial stage in achieving a successful system and giving the user’s confidence that the new system is workable and effective. Implementation of a modified application to replace an existing one. This type of conversation is relatively easy to handle, provide there are no major changes in the system.

Each program is tested individually at the time of development using the data and has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the satisfaction of the user. The system that has been developed is accepted and proved to be satisfactory for the user. And so the system is going to be implemented very soon. A simple operating procedure is included so that the user can understand the different functions clearly and quickly.

Initially as a first step the executable form of the application is to be created and loaded in the common server machine which is accessible to the entire user and the server is to be connected to a network. The final stage is to document the entire system which provides components and the operating procedures of the system.

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Implementation is the process of converting a new system design into operation. It is the phase that focuses on user training, site preparation and file conversion for installing a candidate system. The important factor that should be considered here is that the conversion should not disrupt the functioning of the organization.

**CONCLUSION**

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 intracluster 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 skewness. Finally, ssl_with_bf is more robust than ssl_with_session in handling variable file sizes.

REFERENCES