Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 51 (2007) 4492–4510
Computer Networks
www.elsevier.com/locate/comnet

Designing an overload control strategy for secure e-commerce applications

Jordi Guitart *, David Carrera, Vicenç Beltran, Jordi Torres, Eduard Ayguadé

Barcelona Supercomputing Center (BSC), Computer Architecture Department – Technical University of Catalonia, C/Jordi Girona 1-3, Campus Nord UPC, Mòdul C6, E-08034 Barcelona, Spain

Received 19 October 2006; received in revised form 9 March 2007; accepted 29 May 2007
Available online 6 July 2007

Responsible Editor: E. Cohen

Abstract

Uncontrolled overload can lead e-commerce applications to considerable revenue losses. For this reason, overload prevention in these applications is a critical issue. In this paper we present a complete characterization of secure e-commerce applications scalability to determine which are the bottlenecks in their performance that must be considered for an overload control strategy. With this information, we design an adaptive session-based overload control strategy based on SSL (Secure Socket Layer) connection differentiation and admission control. The SSL connection differentiation is a key factor because the cost of establishing a new SSL connection is much greater than establishing a resumed SSL connection (it reuses an existing SSL session on the server). Considering this big difference, we have implemented an admission control algorithm that prioritizes resumed SSL connections to maximize the performance in session-based environments and dynamically limits the number of new SSL connections accepted, according to the available resources and the current number of connections in the system, in order to avoid server overload. Our evaluation on a Tomcat server demonstrates the benefit of our proposal for preventing server overload.

Keywords: Security; SSL; e-commerce; Application servers; Overload control; Service differentiation; Admission control; Session-based; Scalability characterization

1. Introduction

In recent times, e-commerce applications have become commonplace in current web sites. In these applications, all the information that is confidential or has a market value must be carefully protected when transmitted over the open Internet. Security between network nodes over the Internet is traditionally provided using HTTPS [1]. With HTTPS, which is based on using HTTP over SSL (Secure Socket Layer [2]), you can perform mutual authentication of both the sender and receiver of messages and ensure message confidentiality. Although providing these security capabilities does not introduce a new degree of complexity in web applications

* Corresponding author. Tel.: +34 93 405 40 47.
E-mail addresses: jguitart@ac.upc.edu (J. Guitart), dcarrera@ac.upc.edu (D. Carrera), vbeltran@ac.upc.edu (V. Beltran), torres@ac.upc.edu (J. Torres), eduard@ac.upc.edu (E. Ayguadé).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2007.05.010

structure, it increases remarkably the computation time needed to serve a connection, due to the use of cryptographic techniques, becoming a CPU-intensive workload.

At the same time, current sites are subject to enormous variations in demand, often in an unpredictable fashion, including flash crowds that cannot be easily processed. For this reason, the servers that host the sites must be ready to handle situations with a large number of concurrent clients.

Dealing with situations with a large number of concurrent clients and/or with a workload that demands high computational power (for instance secure workloads) can lead a server to overload (i.e. the volume of requests for content at a site temporarily exceeds the capacity for serving them and renders the site unusable). During overload conditions, the response times may grow to unacceptable levels, and exhaustion of resources may cause the server to behave erratically or even crash, causing denial of services. In e-commerce applications, which are heavily reliant on security, such server behavior could translate to sizable revenue losses.

Overload prevention is a critical goal so that a system can remain operational in the presence of overload even when the incoming request rate is several times greater than the system capacity, and at the same time is able to serve the maximum number of requests during such overload, while maintaining response times at acceptable levels. With these objectives in mind, several mechanisms have been proposed to deal with overload, such as admission control, request scheduling, service differentiation, service degradation and resource management.

However, the design of a successful overload prevention strategy must be preceded by a complete characterization of the application server scalability, which should consist of something more complex than simply measuring the application server performance with different number of clients and determining the load levels that overload the server. A complete characterization should also supply the causes of this overload, in order to identify which factors are the bottlenecks in the application server's performance that must be considered in an overload prevention strategy. For this reason, this characterization requires powerful analysis tools that allow an in-depth analysis of the application server's behavior and its interaction with the other system elements (including distributed clients, a database server, etc.). These tools must support and consider all the levels involved in the execution of web applications if they want to provide meaningful performance information to the administrators because the origin of performance problems can reside in any of these levels or in the interaction between them.

Additionally, in many web sites, especially in e-commerce, most of the applications are session-based. A session contains temporally and logically related request sequences from the same client. Session integrity is a critical metric in e-commerce. For an online retailer, the higher the number of sessions completed, the higher the amount of revenue that is likely to be generated. The same statement cannot be made about individual request completions. Sessions that are broken or delayed at some critical stages, like checkout and shipping, could mean loss of revenue to the web site. Sessions have distinguishable features from individual requests that complicate the overload control. For this reason, simple admission control techniques that work on a per request basis, such as limiting the number of threads in the server or suspending the listener thread during overload periods, may lead to a large number of broken or incomplete sessions when the system is overloaded (despite the fact that they can help to prevent server overload).

This paper contributes a complete characterization of the scalability of Java application servers running secure e-commerce applications. We have decided to focus on the Tomcat [3] application server, which serves as a reference implementation of the Sun Servlet and JSP specifications. The characterization is divided in two parts. The first part consists of measuring Tomcat's vertical scalability (i.e. adding more processors) when using SSL and determining the impact of adding more processors when the server is overloaded. The second part involves a detailed analysis of the server's behavior using a performance analysis framework, in order to determine the causes of the server overload when running with different numbers of processors. This analysis demonstrates the convenience of developing a strategy to prevent server overload, and identifies the factors that must be considered for its implementation.

With this information, we design an overload control strategy that is based on SSL connection differentiation and admission control. SSL connection differentiation is accomplished by proposing a possible extension of the Java Secure Sockets Extension [4] (JSSE) package, which implements a Java version of the SSL protocol, to distinguish between SSL connections depending on whether the connection