WHITE PAPER

The Evolution of Application Acceleration:

From Server Load Balancers to Application Delivery Controllers

www.crescendonetworks.com

Corporate Headquarters
6 Yoni Netanyahu Street
Or-Yehuda 60376, Israel
Phone: +972-3-538-5100

US Headquarters
633 Menlo Avenue, Suite 230
Menlo Park, CA 94025
Phone: (866)830-0400

Introduction

Web applications are redefining the way many businesses operate. Behind the scenes, the architecture that supports these applications has evolved as well to meet the ever-growing demands for performance, throughput and availability.

In the early days of web applications, high volume sites used server farms to scale applications in a horizontal fashion, adding more web servers as demand increased. The server load balancer played a critical role in this architecture, enhancing availability and performance by distributing the load between available servers. These traditional load balancers worked at the packet or TCP connection level, without looking into the content of the data it was directing.

Today's web environment is much more complex, with rich media applications, more transactions and interactivity, and enormous traffic volumes. While load balancing is still an important function, it is only one of the many services that support high-volume web applications. A new breed of device has evolved to handle this increasingly complex environment – the application delivery controller (ADC).

This paper briefly explores the evolution from load balancer to application delivery controller, and its evolving role within the data center.

Traditional Server Load Balancing

Initial load balancing technology evolved from routers and switches; devices used simple NAT (Network Address Translation) and TCP connection metrics to distribute requests across a pool of servers, balancing the load evenly between multiple servers. These traditional load balancers could also ensure that requests from the same user (IP address) reached the same server, to ensure session persistency or "stickiness".

However, as web applications evolved, these load balancers needed to be able to make more intelligent decisions about how to distribute load. They now had to direct traffic based on the content of the request (i.e. requested URL, various HTTP headers, etc). They also had to observe HTTP persistence, complicated by the fact that a user did not always use the same IP address through a single session. HTTP cookies helped mitigate this problem.

To address this situation, load balancing devices that had previously made decisions based on information in the IP and TCP layers had to dig deeper, analyzing and manipulating HTTP headers. Load balancing vendors devised creative ways to do this, using TCP techniques such as splicing and delayed binding to analyze layer 7 information for making load balancing decisions.

While these changes were adequate for simple load balancing applications, they imposed a significant performance burden on the load balancing device. And these basic mechanisms did not scale well for HTTP applications that need continuous inspection of layer 7 header information.

The emergence of the SSL (Secure Sockets Layer) protocol for securing web applications further complicated the situation. To make content level load balancing decisions for SSL sessions, load balancing devices were forced to interact with an outside appliance that would decrypt the requests, pass them to the load balancer for routing decisions, and then re-encrypt the response. This further slowed the load balancers and often resulted

in complicated network topologies where multiple devices were necessary to handle a single request/response chain with the servers.

The Need for an Architectural Change

The nature and volume of web traffic has grown dramatically in recent years. Today's websites and businesses serve global user populations with advanced, interactive web-based applications. Software-as-a-Service (SaaS) companies stake their business on the performance and reliability of their web applications. Emerging social networking and Web 2.0 increase interactivity and rich media, putting further strains on web-based applications that now have to scale to handle very large volumes of traffic and data.

For application providers, maintaining the end user Quality of Experience (QoE) is critical. Poor or unreliable performance can threaten revenues or profitability as businesses use web applications to serve employees, partners and customers. By eliminating boundaries and lowering barriers to entry, the pervasive web also increases baseline expectations for performance and richness of experience, as online competition is closer than ever before.

The traditional server load balancing architecture is insufficient for this evolving and complex application environment. In addition to server load balancing, web servers need to handle SSL, data compression and rich media. They must operate in an ever-changing threat environment and handle unpredictable, 'flash crowd' traffic events. And they need to do all of this handling constantly growing traffic volumes, without a similarly expanding operating budget.

In response to this environment, the server load balancer itself has evolved and changed, abandoning the switch/router model for a new, proxy-like architecture that no longer operates on a packet-by-packet level. These next generation devices go beyond load balancing to address other features such as TCP management, SSL offloading, and enhanced security. These devices go by many names, including application front ends and next-generation server load balancers. For clarity, we'll refer to this next generation device as an **Application Delivery Controller** (ADC).

Application Delivery Controllers

Application Delivery Controllers (ADCs) are a new breed of devices that not only have a better understanding of higher-layer protocols, but also provide seamless integration between multiple functions that need to be performed at HTTP levels. Their applications extend well beyond basic load balancing into application performance optimization.

An ADC operates more like a network host than a router or a switch. It uses a proxy model and is capable of terminating TCP connections rather than simply acting as a relay agent for them. This important capability enables a wide variety of other functions.

The ADC is the actual TCP endpoint for all user connections; the ADC then opens and maintains independent TCP connections with each of the servers in the server pool.

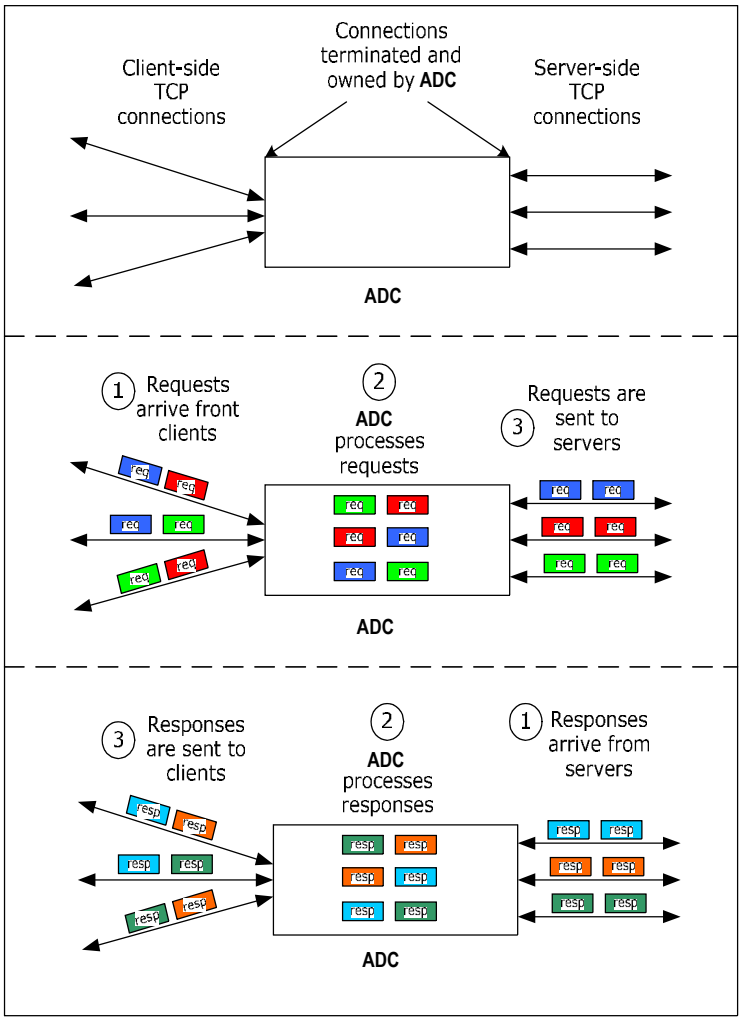
1. A user request flows over client-side TCP connections to the ADC, which now owns the request.
2. The ADC analyzes the request headers and makes decisions based on the request itself, rather than the connection the request arrived on.

3. Once the server decision is made, the ADC passes the request to a server over an already-existing server-side connection.

The process is reversed for responses from servers to clients; the ADC processes the requests and communicates the responses to clients.

Essentially, TCP is only relevant when the ADC needs to communicate a request or a response with a client or server. For decision-making and processing functions, the ADC deals with each request independently and in whole.

The following diagram is a simple illustration of how TCP-terminating ADCs operate:



Using this design, the ADC can apply many processing tasks to requests as they flow through the system, including:

- Layer 7 load balancing based on headers, URL, etc.
- Content encryption/decryption
- Security policies
- HTTP protocol header manipulation

Likewise, when the responses arrive from the servers, they also exist wholly and independently in the ADC, which can apply various processes to the responses, including:

- Content compression
- Content re-encryption
- HTTP protocol header manipulation

These are simply examples of what can be done to requests/responses as they flow through the system. The important point is that the ADC is capable of making decisions and applying policies to the requests/responses independently of the TCP connections to which they belong. This is a significant advantage for ADCs, as the transport protocol is not involved in making any HTTP or object-based decisions.

Typical ADC Functionality

This section describes some of the processing and functions that ADCs can perform in the web application traffic flow. Beyond simple load balancing, the ADC offers opportunities for optimizing and accelerating traffic for a significant performance benefit in today's high-volume and transaction-rich web environment.

TCP offload and acceleration is one of the most basic services ADCs can provide for the servers they're front-ending. Server TCP stacks cannot handle large volumes of WAN-based TCP connections gracefully. An ADC can terminate the thousands of incoming TCP connections from the users and then, in turn, establish a small number of long-lasting TCP connections between itself and the servers. Using persistent TCP connections, the ADC can send requests from all the incoming connections to the server over a small number of server-side connections. This mechanism (often referred to as "TCP Multiplexing" or "TCP Pooling") eliminates a significant TCP processing task from the servers, freeing processing power for the application itself, rather than its overhead.

SSL offload and acceleration is another significant benefit that an ADC can provide an application. It's a well known fact that the cryptographic algorithms associated with SSL significantly hamper the performance of a server. By using SSL hardware to offload the security algorithms, the ADC can terminate SSL sessions at scale and then send the requests to the servers over non-encrypted HTTP, removing the SSL overhead from the servers. Even if end-to-end security policies mandate that all requests must reach the servers securely, the ADC can use lighter encryption keys and longer-lasting SSL sessions between itself and the server to significantly minimize the impact of the secure session processing on the server.

Content compression is yet another major service that an ADC can offer an application. Because the ADC is dealing with whole transactions at the request and response level, it can compress content on its way from the servers to the clients. Since all popular browsers can now handle compressed content, this feature can be seamlessly integrated into the network without any change to the application itself. Compression has huge benefits for web applications, reducing client response times and minimizing the amount of outbound bandwidth used, along with the associated costs. In addition, compressing objects at the ADC removes that CPU burden and responsibility from server resources.

Load balancing now becomes a natural extension of the ADC's inherent functionality. Since the device already has connections to all the servers, all it needs to do is make an intelligent decision to pick the best server for the request. Because the ADC operates at the transaction level, parsing the HTTP headers is much easier, allowing complex load balancing tasks such as URL switching or cookie persistence to be performed with ease. Beyond server load balancing, the ADC is well situated to implement global load balancing – monitoring and distributing traffic between multiple, geographically separate data centers for optimal resource utilization.

Feature consolidation is the ability to consolidate many features into a single platform. Individually, each of the features listed above adds value to the application environment. If implemented properly within the ADC, multiple features can work together seamlessly, eliminating the need for individual point products. For example, requests arriving over SSL sessions can be load balanced and the responses compressed before being re-encrypted. This level of integration is only possible because the features are offered in a single platform. Before this technology was prevalent, IT administrators had to employ separate boxes (or redundant pairs of boxes) to accomplish the various tasks not integrated in a single ADC platform. By reducing the number of point products needed, ADCs simplify network design and implementation, and reduce the number of failure points within the network.

Crescendo Networks' AppBeat DC

Application performance is one of most critical challenges facing organizations today. Crescendo Network's AppBeat™ DC addresses this challenge by providing IT organizations with a simple and powerful way to improve application performance and availability. It accelerates and optimizes essential web applications by offloading and consolidating common tasks, so that server resources can be dedicated to the application itself. AppBeat DC has been independently validated in third-party tests as the clear performance leader in the Application Delivery Control (ADC) market, with performance far exceeding the competition.

AppBeat DC achieves its industry-leading performance using innovative hardware and software technologies. The underlying hardware for AppBeat DC, the Maestro platform, is the only industry solution to implement Layer 2-7 functionality in dedicated hardware with fully integrated TCP termination/optimization, load balancing, compression, and SSL acceleration. Each function runs on a separate, purpose-built engine with dedicated CPU and memory resources. As a result, AppBeat DC can enable all of the functions at the same time without any performance slow-down. This feature concurrency distinguishes AppBeat DC from other application acceleration solutions that slow down as more features are enabled.

With AppBeat DC, organizations can accelerate application performance, improve the end user experience, increase security, and reduce data center expenditures.

Accelerate Application Performance

AppBeat DC delivers industry-leading application acceleration at multi-gigabit rates. With powerful, purpose-built hardware and innovative technologies, AppBeat DC performs multiple application acceleration functions concurrently, enabling unparalleled performance even under heavy load.

Improve End User Experience

Fast, consistent and reliable performance creates a better application experience and shortened response time for the user. Patent-pending Short-Lived Transaction (SLT) technology, zero-latency compression and server normalization techniques improve the performance delivered to end users by 30-70%.

Increase Security and Application Assurance

AppBeat DC shields servers from malicious attacks and mediates flash crowd events. Removing the impact of peak- load periods on application response times ensures consistent application availability for customers.

Reduce Data Center Expenditures

By consolidating and offloading critical functions, AppBeat DC increases available server capacity by 300-500%. In addition, efficient hardware-based compression reduces bandwidth requirements by up to 75%. Using AppBeat DC, IT organizations can reduce existing and planned expenditures for a clear and immediate ROI.

Conclusion

While load balancers have long served an important role in the data center, today's high-volume web applications require more than simple load balancing. The Application Delivery Controller can consolidate several critical front-end functions for web applications, including TCP management, SSL offload, compression, and global load balancing. Running these different features in a single box optimizes application performance and enhances the end user experience, while reducing data center costs through consolidation and enhanced server utilization.

When evaluating ADCs, it is critical to look not only at the device's performance and range of features, but also at the performance of the device when multiple or all features are enabled at once. For many devices, using several of the features (such as SSL offload and compression) degrades overall performance and reduces the benefit provided by the ADC. To truly optimize the web application infrastructure, you need an ADC that supports feature concurrency without performance degradation.

Crescendo Networks' AppBeat DC delivers the full capabilities of the ADC. It supports a range of features, including global load balancing, TCP management, SSL offload, application assurance and data compression. Using multiple, dedicated processors, the purpose-built hardware is able to perform all of its optimization and acceleration functions concurrently without affecting overall throughput or performance.

About Crescendo Networks

Crescendo Networks is the recognized performance leader for accelerating and optimizing the delivery of business-critical, web-enabled applications. The company's unique multi-tier application delivery architecture dramatically improves the operation of today's demanding application infrastructure. The world's largest corporations and fastest growing web properties rely on Crescendo for the application performance and efficiency needed to ensure usability, facilitate rapid business growth, lower IT costs and capture additional revenue. To learn more about Crescendo Networks' application delivery solutions, visit www.crescendonetworks.com