Server Load Balancing Using Adaptive Routing on an MPLS Enabled Network

Joseph Grzybowski

Abstract
As companies demand greater scalability, performance, and availability from their network-based solutions, server load balancing continues to grow in importance. Today a variety of techniques are used to perform load balancing. Two of the most popular are round-robin DNS and content switching. This paper proposes a new load-balancing technique which uses adaptive routing on an MPLS-enabled network. We propose that service requests destined for a web cluster be distributed across the mirrored servers by changing dynamically the route cost metrics. MPLS’s ability to create pinned label-switched paths is used to maintain session persistence. In this paper we present the architecture and operation of this new load-balancing technique. We then compare our new approach to the two most popular load balancing methods.

1. Introduction
In recent years there has been a tremendous surge in the popularity and use of the internet. This surge is a result of the increasing availability of home PCs, wide spread deployment of broadband connectivity, and ever more creative ideas from companies seeking to leverage this resource to obtain a competitive advantage. Web server accesses have increased a thousand fold, failed connections result in lost sales, and static web pages have been replaced by customized content.

As web server usage grew in quantity, importance, and complexity, it was soon discovered that one server is not enough. Scalability problems were encountered. In some cases it just wasn’t possible to buy a big enough server. Reliability quickly becomes an issue when revenue depends on it, and a failed server results in substantial losses. Users expect shorter response times as they increasingly migrate from dialup to broadband access.

In order to meet these needs server load balancing (SLB) solutions are used. These solutions allow the replacement of a single web server with multiple servers, often referred to as a cluster. Incoming requests are distributed across these servers. By allowing the use of a cluster to service client requests as if it were one big server many benefits are attained. Scalability, reliability, and performance become possible.

A variety of SLB solutions are presently in use and more are developed everyday. Internet growth shows no signs of stopping. Server clusters are recognized as the best way to keep up with this growth. We are proposing an MPLS based SLB solution.

2. Background
Multiprotocol Label Switching (MPLS) is a new set of standards being developed as a way to improve upon the speed and flexibility of conventional IP routing. The IETF MPLS working group is the leading force in this effort. Many aspects of the evolving MPLS standards are well on their way to becoming RFC standards.

In conventional routing schemes, packets traverse an IP network through a process referred to as hop-by-hop destination-based unicast. In this architecture, the packet header is examined by each router it traverses on its way from source to destination. Each router extracts the packet destination address and determines to which router the packet is forwarded next. This technique has two weaknesses: 1. The need to decode IP information embedded in layer 3 of the IP stack for each packet, then applying a forwarding rule to the result before forwarding each packet, directly impacts routing speed. 2. The independent nature of this destination-based decision-making is inflexible, causing all packets having the same destination to follow the same route.

MPLS seeks to eliminate these routing weaknesses by separating the decision-making control component from the forwarding component. This is done by using labels. An MPLS label is a short identifier that is prepended to each packet as it enters the network at the ingress router. The specific label that is assigned to any particular packet is determined through a classification procedure that segments inbound traffic into orthogonal groups called Forward Equivalency Classes (FEC). Once a packet is labeled, it can be forwarded across the network using fast layer 2 switching techniques.

The assignment of packets to FECs plays a pivotal role in the control component of the MPLS architecture. By examining the destination address, source port, diffserv bits, or any other relevant aspects of the packets, an MPLS ingress router can classify inbound traffic. Once the traffic has been assigned a FEC, it can be matched to a predefined label-switched path (LSP), Figure 1, along which it will traverse the network. Because packets are examined and classified only once upon entry to the network, additional information beyond layer 2 and layer 3 can be examined without seriously impacting performance. The ingress router can thereby make more sophisticated and flexible routing decisions than are possible in conventional routing.

An LSP is an MPLS layer 2 construct along which all the packets of a FEC are transported. LSPs typically begin at an MPLS ingress router, traverse a number of interior label switch routers (LSRs), and terminate at the network exiting point or egress LSR. Each LSR along the LSP uses a predefined inbound label mapping table (ILM) to forward the packets based only upon their FEC label. Because the LSP was established previously, the decision...
required at each router is much simpler than in conventional routing, thereby increasing the speed and throughput of the network.

When a packet is received on an LSR port, the following chain of events is set into motion: First the packet is examined for an MPLS label. If an MPLS label is not found, then the LSR may act as an ingress router and assign the packet a FEC and label it. The LSR may optionally choose not to use label switching and simply forward the packet in the conventional IP manner. We will not here examine the conventional IP forwarding option further.

The second stage in the LSR processing of a labeled packet consists of an ILM table lookup in order to determine the next router to which the packet should be forwarded. If the LSR determines itself to be the next hop for the packet, then that LSR is considered to be the packet’s egress router. An egress LSR will then ignore the packet’s MPLS label, routing the packet based upon its layer 3 IP destination address, as in the conventional scheme. If the LSR is not the egress router for the packet, and MPLS switching is required, an MPLS label swap is performed.

Label swapping is performed in order to allow for local rather than global label assignments. With local label assignments, each LSR need only coordinate label assignments with its immediate neighbors. Global label assignments require label coordination across all LSRs in a network, resulting in scalability and complexity issues. An LSR performs a label swap by referencing its ILM table. It locates the record in the table which matches the inbound label to an outbound label and performs the swap. The final step in processing a labeled packet is to forward it on to the next LSR along the LSP, as previously determined from the ILM table. The packet is then received by the next LSR and this procedure is repeated. In this way each packet is classified, labeled, and transmitted along the appropriate LSP until it reaches its egress router, and ultimately its destination.

3. Proposed Server Load Balancing Solution

Our proposed server load balancing solution uses MPLS’s ability to establish LSPs and to assign traffic for transmission on each path based upon the FEC to which it belongs. By classifying packets based upon source address, source port, destination address, and destination port, we establish FECs that correspond to individual TCP sessions. Then, by setting up LSPs which end at a number of separate servers and distributing the TCP sessions across the corresponding paths, load balancing is achieved.

3.1 OSPF’s Role in LSP Routing

MPLS label switched paths can be established in a number of ways. For the purpose of our SLB solution we are particularly interested in MPLS’s ability to work in conjunction with the LSR’s IP routing tables when creating LSPs. This method allows LSRs to create paths which coincide with least cost routes as determined by an interior gateway routing protocol such as OSPF.

OSPF is a widely deployed routing protocol that allows network routers to exchange network topology information. Link state advertisement datagrams are exchanged between routers as each router tells its neighbors about the networks to which it is attached and the associated link cost metric. Once all of the routers have shared their link state information with one another, the network is said to have converged. At this time every router has the same view of the network’s topology. The routers then utilize the Dijkstra algorithm to generate least cost trees and the resultant routing tables. MPLS can then use these routing tables to set up paths across the network.

Our SLB technique takes advantage of OSPF’s ability to allow for the existence of multiple links connected to the same subnet. These links may or may not have the same cost metric assignments. Because OSPF identifies links according to the subnet to which they are connected, links which appear to connect to the same network may actually connect to two physically distinct networks that have the same IP network address. This characteristic allows us to have multiple servers with the same address residing on our network. These servers work together as a single virtual server, thereby maintaining application-connection integrity.

3.2 Mirrored Server Architecture

In our SLB solution, we use mirrored servers that have identical IP addresses to create a virtual server. Each member server has identical information and is therefore equally capable of servicing incoming requests from clients. The mechanism for maintaining synchronization across all the member servers within a virtual server is beyond the scope of this article. Because each server shares a common IP address, the actual server with which a client is communicating is transparent to the client. This is a critical feature for allowing SLB that is transparent to the clients.

In addition to mirroring, our SLB solution also requires each member server to run an internal IP and MPLS routing daemon. By doing this we achieve two things. First it allows for physically distinct servers to look as if they reside on the same logical subnet. By using an internal routing daemon the actual server IP address is walled off from the rest of the network. Each server appears to the rest of the network as just another appearance of the same virtual server subnet. This circumvents issues relating to ARP and to OSPF security. Secondly, this internal routing daemon allows the servers to have direct control over the link cost advertised by their internal routers. This lets the server adjust the link cost whenever desired. By running an internal routing daemon, the need to patch external router operating systems is eliminated.

Our proposed solution uses MPLS’s ability to set up LSPs based upon the least-cost routes identified in the LSR’s IP routing tables. Having the ability to change the cost metric of the last link to each of our mirrored servers, we can now influence to which server LSPs will be
created. If a server is lightly loaded and can handle more traffic, then its link cost metric is reduced, making it more likely to be the destination of LSPs being created. As a server’s load increases, a corresponding increase in its link cost is used to direct traffic away from it, towards servers that are less loaded. This is referred to as adaptive link cost routing.

At this point we encounter one of the problems that exist in conventional IP routed networks. The nature of hop-by-hop destination-based unicast routing does not allow for partial shifts in traffic that has the same destination address. Conventional IP routing recognizes one least-cost route and uses that route for all traffic (assuming that the equal-cost routing capabilities of OSPF are disabled). In a conventionally routed network, cost metric changes would result in all the traffic destined for the server’s address to be shifted to the lower-cost route. We would end up with large shifts in traffic away from a heavily loaded server over to a lightly loaded server, quickly overloading that server, and shifting all the traffic back to the original, now lightly loaded server. This would repeat in an oscillatory fashion. MPLS’s ability to pin LSPs resolves this unwanted behavior.

Not only can MPLS set up LSPs across a network, but it also has the ability to pin a path to a specific route. A pinned LSP is one that once created through a specific set of LSRs does not reroute itself when a lower-cost path becomes available. This ability is important to prevent large traffic shift oscillations. Once a class of traffic has been assigned to a FEC and a pinned LSP, it will not reroute itself to any lower-cost routes that may appear during its lifetime. Working in conjunction with pinned LSPs, adaptive link cost routing can now be used to direct newly classified traffic, new sessions, to the least-loaded, least-cost server, without causing a wholesale shift in all traffic to the least-cost server. Adaptive link cost changes no longer have any effect on previously established client server relationships.

Client to server affinity also serves to maintain session continuity, which is important for several reasons. At the TCP layer it is necessary to maintain TCP sessions in order for communication to take place. If the traffic from a client were to be redirected suddenly to a new lower-cost server, established TCP sessions would be broken and a new session would have to be initiated. This would greatly impact the response times that clients experience. At the application layer it is also beneficial to maintain a relationship between a client and a server. SSL encryption sessions may have been established, half-full shopping carts may exist, and database updates might be in midstream.

In this section we presented a new SLB technique. We demonstrated MPLS’s ability to assign different TCP sessions to unique LSPs. We explored how LSPs can be created and routed based on LSR IP routing tables. We showed how LSR IP routing tables can be manipulated by controlling OSPF link cost metrics. Finally we concluded that OSPF link cost metrics can be used to control the assignment of TCP sessions across mirrored servers. By controlling the distribution of TCP sessions across the multiple mirrored servers, we achieve server load balancing.

4. Alternative Load Balancing Solutions
4.1 Round-Robin DNS
Round-robin DNS is one of most widely deployed SLB solutions. Developed in 1994 by the NCSA, its simplicity and minimal cost have made it very popular. According to a report published by E-Soft Inc. in February 2004, approximately 6.2% of their 14,500,417 DNS lookup queries resolved to multiple IP addresses. They presume that these multiple DNS entries are there for redundancy and server load balancing [8].

When using round-robin DNS, each member server of the load-balanced solution is assigned a unique IP address. All of these IP addresses are listed on the website's authoritative DNS server in the form of A records. As clients perform DNS lookups for the website, the authoritative DNS server responds, providing member server’s IP addresses in round-robin order. Client
transactions are distributed across the website’s member servers on a rotating basis.

Round-robin DNS is often the first solution companies deploy when expanding a website beyond one server. It is simple, inexpensive, adds no overhead to web transactions, and it works with any application that uses DNS resolution. However, round-robin DNS has two major problems.

In order to speed up DNS name resolution, today’s networks frequently use name caching on non-authoritative name servers. Because of name caching, not all DNS lookups result in a response originating from the authoritative server. A client requiring name resolution will frequently have it resolved by a local non-authoritative name server. Whenever this happens, the round robin nature of the DNS lookup is bypassed. When a large number of clients use the same non-authoritative name server, a load bias results. The server whose address has been cached by this popular name server receives an inordinately high number of the incoming transactions.

The second major problem with round-robin DNS load balancing is its inability to react to unexpected member server failures. The A record entries that a name server uses to perform name resolution are static. When a member server fails, the authoritative name server continues to respond to DNS lookups in a rotating fashion. Every n-th lookup results in the IP address of the failed server being returned.

The problem of distributing IP addresses of failed member servers can be minimized. At additional expense, the authoritative name server for the website can be adapted to recognize a failed member server. The DNS server can then remove the address of the failed server from rotation. However, name caching reduces the effectiveness of this approach.

In order to alleviate the adverse effects of name caching, companies can reduce the time to live (TTL) parameter the authoritative name server sends as part of its responses. Reducing this TTL value causes non-authoritative name servers to discard the cached IP address more quickly. Setting the TTL to zero says that the IP address should never be cached. A reduced TTL causes a greater number of DNS queries to be resolved by the authoritative name server, affecting the speed of the DNS lookups. To make matters worse, not all name servers adhere to the TTL parameter that was passed to them. Many DNS servers cache the IP address for as long as they see fit.

4.2 Content Switching

Content switching is a server load-balancing technique that utilizes a specialized device at the front end of a server cluster to distribute incoming traffic. This front-end device is called a content switch. Its purpose is to classify inbound traffic and to dispatch it to the appropriate back-end member server for processing. Clients that are accessing a content-switched virtual server always address their communications to the IP address of the content switch and not directly to any member server.

Content switching can be performed using a variety of techniques: (1) The content switch may act as a proxy device between the client and the member server. (2) The content switch may use IP tunnels to route traffic to member servers. (3) The content switch may perform a network address translation (NAT), changing the packet’s destination IP address and forwarding it on to the appropriate member server. Because NAT is the most common method used by content switches, we will concentrate on it and disregard the other methods [3].

When performing NAT the content switch intercepts the inbound packet, decides to which server it should be forwarded, and replaces its destination IP address with that of the appropriate member server. The packet is then forwarded. The member server processes the packet and responds.

An important characteristic of NAT content switching is whether a server’s response goes directly to the client or goes back through the front-end device for processing before being returned to the client. We use the terms one-way and two-way when referring to these options. “One-way” refers to when packets pass through the content switch on their way into the cluster but bypass the switch on the way out. In a “two-way” system, both inbound and outbound traffic traverses the content switch.

There are two primary concerns when deciding whether to use a one-way or two-way architecture. Typically, using a two-way content switching architecture keeps member server modifications to a minimum. The content switch performs all address translation in both directions, keeping the translation transparent to both the client and the server. The server has no need to spoof IP addresses or use loop-back devices [3]. However, because the outbound traffic from servers is frequently much greater than its inbound traffic, requiring traffic to traverse the front-end switch in both directions results in the need for the switch to process a large number of packets. This results in performance and scalability issues. Consequently most content switching architectures are one-way.

The content switch’s greatest strength lies in its ability to route inbound traffic based not only on the layer 3 and 4 information in a packet, but also based upon the application content of each packet. For instance, a packet containing the URL for file A may be directed to server A because it knows that server A can respond to this request the fastest (the file may be cached). While at the same time, a packet requesting file B may be directed to server C because server C happens to be the least loaded server. This example demonstrates the ability of a content switch to take both the status of member servers and the requested material into consideration when distributing inbound traffic. Content switches have the ability to utilize member servers’ resources better and maintain smoother load balancing than other methods.

As always, there is a tradeoff. Content switching SLB solutions require all inbound traffic to flow through the front-end device. This device becomes a bottleneck if its capacity is exceeded. It also presents a single point of failure if it’s not paired with a redundant failover switch. Also, content switches are required to operate at extremely fast speeds attained through the use of specialized hardware called ASICs (application-specific integrated circuits) and proprietary software operating systems. The proprietary nature of the content switch often translates into a high cost for deployment.

5. Solution Comparison

5.1 Granularity

One characteristic that is used to describe server load-balancing techniques is called granularity. Granularity refers to the unit size at which traffic can be controlled in its distribution across member servers. For our comparison purposes we have identified four granularity levels: 1-Extra Fine (URL), 2-Fine (TCP Session), 3-Medium (Host Address), and 4-Coarse (Network Address).
The extra fine or URL level of granularity represents an ability to distribute incoming traffic based upon application layer information. This granularity is typically achieved by content switches operating at layer 7. These switches have the ability to examine a packet’s application layer contents, an URL for example, and route the packet to the most appropriate server.

Fine or TCP session granularity represents the ability to differentiate packets that belong to different TCP sessions. This is done using the socket information that is contained in the TCP and IP headers of each packet. At this level client-server affinity is inherent. This level is especially good for websites which are using SSL encryption. Our proposed MPLS load-balancing technique operates with this level of granularity as well.

Medium level granularity allows for the distribution of incoming traffic based upon the source host IP address. All packets having the same host IP address are directed to the same member server. Because outbound NAT is used in many corporations, this level of granularity does not always result in what is expected. When a NAT device is in use, traffic originating from many different hosts will appear as if it is from a single host IP address. For SLB solutions which depend on the source address (or a hash of the source address) for distribution, NAT results in a load imbalance or bias.

Coarse granularity refers to the distribution of traffic across member servers based on the network from which the transaction is originating. At his level all packets that originate from the same network go to the same cluster server. Round-robin DNS typically operates at this level. Because of local name caching, many clients on a network are issued the same server IP address.

5.2 Cost
Cost is always a primary consideration when comparing products. The selection of a SLB solution is no exception. Rather than perform a dollars-and-cents analysis of the alternatives, we estimate costs based upon the solutions’ relative complexity and proprietary natures.

Round-robin DNS’s popularity as a load balancing solution is a result of its simplicity and low cost. Requiring only an additional A record entry at the authoritative DNS server for each additional member server, round-robin is by far the least expensive of the solutions (cost=1).

Implementation of our proposed MPLS load balancing solution has two requirements. First it needs an MPLS-enabled backbone and second it requires an MPLS routing daemon on each member server. As MPLS becomes more widely deployed for QOS and traffic engineering purposes, we will be able to leverage its existence. The routing daemon at the server and its modification to perform adaptive link cost routing as the server’s load changes still needs to be written, but it is expected that it can be developed at minimal cost from already existing daemons such as MPLS for Linux. We estimate our solution’s cost as average (cost=2).

Front-end content switches are by far the most expensive of the load-balancing options. These solutions require high-speed ASIC hardware and frequently use proprietary software. Front-end switches offer the finest traffic flow granularity, but at the greatest cost. We estimate their cost as high (cost=3).

5.3 Scalability
The scalability of a solution refers to its ability to grow as the demands being placed upon it are increased. Scalability is one of the primary objectives that SLB is meant to achieve. A server load-balancing solution is scalable if its size (the number of transactions it can handle) can be steadily increased through the addition of servers without running into problems.

While as yet there are no simulated models or real-world implementations of our MPLS load-balancing technique, we estimate that its scalability is good (1). This solution has no bottleneck points, and does not exhibit any unrestrained growth in overhead due to the addition of member servers.

Round-robin’s DNS scalability is considered average (2). While we have heard of concerns about DNS’s ability to operate correctly with more than 5 servers [6], E-Soft Inc.’s February 2004 DNS Load Balancing Report clearly indicates that more than forty percent of the companies using round-robin DNS have more than 5 servers. E-Soft’s data also seems to indicate a marked drop in round-robin DNS usage at the 21 server point.

We consider content switching to have poor scalability (3). In this solution the content switch itself has become the bottleneck. If a content switch is found to be reaching its load limit, then the switch must be replaced by a larger switch.

5.4 Availability
Availability is probably the most important characteristic of a web server and by extension a load balanced server cluster. We rank availability over scalability and performance, due to the fact that a failed or unreachable server might as well not exist at all.

When examining the load balancing solutions to see how they fare in the area of availability we consider two things: 1. How does the technique react to a member server failure, 2. Are there single points of failure.

Our proposed MPLS load balancing technique does well in the area of availability when compared to round-robin DNS and content switching.

As noted above, Round-robin DNS does not recognize when a member server has failed and continues to direct clients requests to it. The static nature of DNS entries and the wide spread use of DNS caching results in a solution that fails to or at best is slow to react to a server failure. Round-robin DNS server load balancing has “poor” availability characteristics.

Content switching on the other hand recognizes failed servers but instead introduces a possible single point of failure. It is possible to minimize this weakness by using redundant content switches, but this results in doubling the cost of the solution. Content switching has “average” availability characteristics.

Our proposed technique does not introduce any new single points of failure and is responsive in the event of a member server failure. Because the initial traffic distribution function occurs at the MPLS ingress router, as long as there is access to the network, via an ingress router, load balancing can be performed. In the event of a server failure, that server’s link cost goes to infinity when it fails to respond to OSPF Hello messages. LSP will not be built on a route with infinite cost. Our proposed MPLS load balancing solution has “good” availability characteristics.
6. Conclusion

In order to meet the high transactional and availability demands placed on web services today and in the future, many companies are employing server load-balancing solutions. Round-robin DNS and content switching are two of the more popular approaches in use today. These techniques are known to have reliability and scalability issues, and in the case of content switching, high cost.

Table 1. Summary of Load-Balancing Techniques

<table>
<thead>
<tr>
<th></th>
<th>Round Robin DNS</th>
<th>Content Switching</th>
<th>MPLS Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability</td>
<td>Ave – 2</td>
<td>Poor – 3</td>
<td>Good – 1</td>
</tr>
<tr>
<td>Availability</td>
<td>Poor – 3</td>
<td>Ave – 2</td>
<td>Good – 1</td>
</tr>
<tr>
<td>Granularity</td>
<td>Network – 4</td>
<td>URL – 1</td>
<td>Session – 2</td>
</tr>
<tr>
<td>Cost</td>
<td>Low – 1</td>
<td>High – 3</td>
<td>Average – 2</td>
</tr>
</tbody>
</table>

In this paper we explored a new server load balancing technique that uses MPLS. We showed how the technique works and then compared it to the two most popular load-balancing methods (Table 1). In our comparisons we found that our technique is as good as or better than the existing techniques in most of the evaluation categories.

Due to the proposed technique’s potential, further research and development is warranted. As a newly proposed idea, this technique has many aspects that can benefit from further study.

One aspect of this proposed technique that warrants further investigation is the use of OSPF as the underlying routing protocol. We chose to use OSPF as our routing protocol primarily because it is widely understood. More work needs to be done determining how its convergence time effects the operation of this technique, if the use of OSPF areas will improve the technique, or whether another protocol such as BGP or IS-IS is more appropriate.

It is suspected that scalability can be improved. Establishing and maintaining a separate LSP for every session is likely to be more than necessary. We believe that a better solution may be possible using a small number of LSPs from each ingress LSR to the member servers. By pre-establishing the LSPs and assigning FECs to these LSPs for each session in accordance to the LSPs’ overall costs when sessions are established, repetitive LSPs can be avoided.

Server load balancing using adaptive link cost routing shows much potential. As MPLS becomes more widely deployed we believe that our proposed SLB solution, with its ability to leverage the MPLS infrastructure, will be of great interest.

References