

Scott Forsyth

Posting on IIS, ASP.NET, Web farms, and
general system admin

Understanding Reverse Proxy Servers – and the Mailman

Saturday, August 8, 2009

[IIS \(/web/20160423203606/http://weblogs.asp.net/owscott/Tags/IIS\)](http://web.archive.org/web/20160423203606/http://weblogs.asp.net/owscott/Tags/IIS)

[IIS7 \(/web/20160423203606/http://weblogs.asp.net/owscott/Tags/IIS7\)](http://web.archive.org/web/20160423203606/http://weblogs.asp.net/owscott/Tags/IIS7)

Ok, the goal isn't to learn about the mailman, but he's going to come in handy later.

Proxy servers have been around since the early days of computing and they play a large role on the web today: sometimes obvious, sometimes not. They can be used for good or they can be used for harm, like man-in-the-middle attacks. Today I want to provide a visual representation of a reverse proxy server used for load balancing. I also want to address some concepts and potential issues and solutions that often come up with proxy based load balancing.

Definition

Proxy: Dictionary.com

(<http://web.archive.org/web/20160423203606/http://dictionary.reference.com/browse/proxy>): "the agency, function, or power of a person authorized to act as the deputy or substitute for another."

Proxy server: Wikipedia (http://web.archive.org/web/20160423203606/http://en.wikipedia.org/wiki/Proxy_server):

In computer networks, a proxy server is a server (a computer system or an application program) that acts as a go-between for requests from clients seeking resources from other servers.

Reverse Proxy: Wikipedia

(http://web.archive.org/web/20160423203606/http://en.wikipedia.org/wiki/Reverse_proxy): A reverse proxy or surrogate is a proxy server that is installed in a server network. Typically, reverse proxies are used in front of Web servers.

The proxy server drawn out

This can be easily represented with the following diagrams. This is oversimplifying the meaning slightly, but it communicates the essence of forward and reverse proxies.

Proxy Server



(http://web.archive.org/web/20160423203606/http://weblogs.asp.net/blogs/owscott/Proxy_5515FA94.jpg)(http://web.archive.org/web/20160423203606/http://weblogs.asp.net/blogs/owscott/ReverseProxy_5515FA94.jpg)

The difference between a forward and reverse proxy server is essentially where it lives and who it targets. If it's on the perimeter of the client's network (like a corporate network or ISP) then it's a proxy server. If it sits in front of servers or devices on the web (like in a data center) then it's a reverse proxy server.

Proxy servers can be used for any numbers of applications, some of them include:

Forward Proxy Server Examples

- Caching web pages to increase the *perceived* speed of the internet. AOL has been known over the years for their poor implementation of proxy servers for their users. Many corporate networks use proxies successfully.
- Filtering to ensure that inappropriate or unapproved content isn't allowed to the end user. These are common in corporate environments, schools and colleges. A NetNanny-type of product functions as a proxy server/service.

Bright Data Ltd. v. Oxylabs, UAB

IPR2023-00840

Oxylabs EX2006

Page 1 of 4

Reverse Proxy Server Examples

- Caching content for performance for a web server at the data center. Speeds up a website for all users of that website.
- Load balancing a few servers to distribute load. Allows scalability or redundancy.
- Webpage treatments for client-side performance gains.

As you can see, there are many reasons for proxy servers, and they are in operation all over the web. It's possible that a proxy server is being used for you to see this website right now.

Direct Server Return

I want to briefly mention another type of method used by some load balancers, called Direct Server Return (DSR). It's helpful to contrast this with a Proxy Server.

Load Balancer using Direct Server Return

 ProxyDSR

(http://web.archive.org/web/20160423203606/http://weblogs.asp.net/blogs/owscott/ProxyDSR_1D242C7E.jpg) Notice that the load balancer isn't as aggressive as the ones in the previous diagrams. It passes on the request from the client to the web server and then it minds its own business and stays out of the rest of the communication. In fact, with this solution, the web server barely even realizes that the load balancer played a role. In this role, a load balancer is not taking the role of a reverse proxy server. It does not stand in the middle for the whole process.

Additionally, even though routers and switches are middle-men between the client and server, they are not considered proxy devices either.

Reverse Proxy Servers and Load Balancers

Now on to the main points that I want to cover. Recently I've started to work with Microsoft's new Application Request Routing (<http://web.archive.org/web/20160423203606/http://www.iis.net/extensions/ApplicationRequestRouting>) (ARR) load balancer which I've been extra impressed with. I plan to post more over time on how to really leverage this as a full blown flexible, stable and scalable load balancing solution. Over the last few years I've been using DSR based load balancers for the most part. As a result of working with ARR, I've run into a few concepts and addressed a few issues that I want to cover here.

The Mailman as a Proxy

The biggest issue that comes up with proxying requests is that it's nearly impossible for the proxy (or reverse proxy) server to stay hidden. Consider the mailman who delivers mail to your house each day. He's not the original sender, but he's the person that, at first glance, appears to be the sender. How many jokes are made of the mailman building a romantic relationship with the wife?

 Mailman

(http://web.archive.org/web/20160423203606/http://weblogs.asp.net/blogs/owscott/Mailman_13AB6E08.jpg)

The mailman is a type of proxy. The trick is to make sure that you can tell who the original sender is, and that you don't get confused and give the mailman credit for a letter that's not from him. You certainly don't want your love letters to your spouse or significant other being credited to the wrong person.

In the case of the mailman, there is some evidence of the postal system proxying the mail, in the form of markings in the top right corner. However the larger markings are the 'return' and 'to' addresses. It's important for the receiver to understand which markings to pay attention to and which to ignore.

The Proxy Server Leaves a Mark

When a server acts as a proxy, it will change some of the request headers. In particular, the headers to watch for are:

- REMOTE_ADDR

- REMOTE_HOST
- Some proxy servers can also off-load SSL, which means that it will proxy from SSL to HTTP. In that case SERVER_PORT and some certification headers come into play.

In the case of IIS and the web server, it will try to set REMOTE_ADDR and REMOTE_HOST to the IP of the proxy server. Any code that depends on these headers will get confused. For example, if you check for blog spam by client IP, you may check REMOTE_ADDR from code. With the proxy server in-between, it will appear that all traffic comes from a single IP. Additionally the web server will log the traffic as coming from the proxy server.

To avoid this impacting affect caused by the proxy server, it's necessary to rewrite all relevant header and log information back again so that it appears to come from the original sender. This can be done various different ways, but I'll cover a common method, and how ARR handles this.

ARR's Header Rewriting

With ARR, this can be handled one of four ways:

1. Ignore the issue. Some people don't have any site features that depend on knowing the client IP, and they are fine with not knowing the client IP in their site statistics.
2. Update your code to use the custom headers that ARR sets, namely X-Original-URL, X-Forwarded-For, X-ARR-SSL, X-ARR-LOG-ID. This won't address the IIS logs, but it can address everything else.
3. ARR Helper. Anil Ruia, one of the primary ARR developers, has written a helper module that works in IIS7 to rewrite the relevant headers back. The ARR Helper (<http://web.archive.org/web/20160423203606/http://blogs.iis.net/anilr/archive/2009/03/03/client-ip-not-logged-on-content-server-when-using-arr.aspx>) module is not officially supported by Microsoft, but works like a charm. ARR itself will place the original client's site in a configurable request header, which is X-Forwarded-For by default. It will also place certificate information for SSL offloading in X-ARR-SSL. On the actual web server, the ARR Helper runs silently in the background and will rewrite those headers back to REMOTE_ADDR, REMOTE_HOST, SSL and REMOTE_PORT. It will also ensure that the logs recover the original client IP. It's installed at the server level and doesn't have any impact on non-load balanced sites and performance overhead is negligible (for load balanced or non-load balanced). We're running this in production at ORCS Web (<http://web.archive.org/web/20160423203606/http://www.orcsweb.com/>) and are very pleased with the results. No code changes are necessary for the site owner.
4. With ARR and URL Rewrite 2.0, currently in Beta, it can also do the same thing. ARR itself takes care of the writing on the ARR server, and URL Rewrite 2.0 can rewrite the X-Headers back to the appropriate locations. URL Rewrite 2.0 needs to be installed and configured on each of the web servers. Currently I'm running this in testing only since version 2.0 does not have a go-live license, but the end goal is to use URL Rewrite for the web server rewriting. While Anil's solution works perfectly, this will be the Microsoft supported solution moving forward.

In future blog posts, I hope to dig deeper into ARR technically, but I wanted to lay the groundwork first on the concept of proxying, and what you need to consider by having a middle man between the client and web server.

3 Comments

Quite an explanation, very clear and easy to understand.

— Saurabh (http://web.archive.org/web/20160423203606/http://blogs.msdn.com/saurabh_singh) - Friday, September 11, 2009 9:26:12 AM (/web/20160423203606/http://weblogs.asp.net/owscott/understanding-reverse-proxy-servers-and-the-mailman#comment-1203)

How would a IIS / ARR reverse proxy compare to a squid or nginx proxy, performance wise?

— Theo - Thursday, May 24, 2012 2:49:14 AM (/web/20160423203606/http://weblogs.asp.net/owscott/understanding-reverse-proxy-servers-and-the-mailman#comment-1204)

@Theo. I haven't done a comparison but I know that the overhead for URL Rewrite and ARR is very minimal and is mostly network bound. In other words CPU and memory really aren't an issue in most cases. I've worked with hundreds of URL Rewrite rules that perform effortlessly.

— **OWS**cott (<http://web.archive.org/web/20160423203606/http://weblogs.asp.net/owscott>) - Thursday, May 24, 2012 10:21:14 AM
(<http://web.archive.org/web/20160423203606/http://weblogs.asp.net/owscott/understanding-reverse-proxy-servers-and-the-mailman#comment-1205>)

Comments have been disabled for this content.

Terms Of Use (<http://web.archive.org/web/20160423203606/http://www.asp.net/terms-of-use>) - Powered by Orchard
(<http://web.archive.org/web/20160423203606/http://www.orchardproject.net/>)