**Power Serving:** *Providing High-Availability and Load Balancing using Open-Source tools*

**The Meaning:** My intentions are to create a web cluster that serves as a high availability cluster and a load balancing cluster together to leverage two much desired functions of any web server environment. The requirements include being able to serve web documents both static and dynamic while being secure and smart enough to thwart small and fairly large scale attacks. Caching, reporting, and LDAP authentication will all be features included in this configuration as to mimic a real, industry-like environment. Things like PHP, MySQL, and Python will be implemented to further provide the popular features most modern websites utilize. This project has been of interest to me for some time after delving into advanced web serving topics this past summer as well as early in my college career, I’ve always been looking for ways to further optimize, protect, and create new methods of deploying one of the most popular protocols on the Internet.

**The Method:** The method I intend to use includes utilizing the ever growing [nginx](http://nginx.org/en/%22%20%5Ct%20%22_blank) web server software along with [Heartbeat](http://www.linux-ha.org/wiki/Heartbeat), an open source machine monitor meant for industrial strength application. Each machine will run Debian Squeeze (6.0.\*), a server based, extremely stable UNIX operating system used by an ever-growing amount of organizations. I will use VMware Workstation by creating a Team of VM’s involved in the project along with their own network segment configured just the way we need it.

The format I intend for basically utilizes dedicated Linux load balancers monitoring each other (using Heartbeat) and the actual web servers (using LVS tools or equivalent OSS) to know where to direct traffic (hopefully the least connected machine) while ensuring one of them is up to answer requests on the floating/logical IP clients will use to connect. To break this down, the load balancers will share one public IP that one of them will use at a time. If that main one goes down, another will step in and assume the primary role (taking the public IP). Each load balancer will have access via its internal interface to each actual web server. It will know which ones are available and the server load across all of them in order to make timely decisions on where to direct traffic to. Each actual web server will have access to a single NFS share storing the web content as to not have inconsistencies in the web data.

**An Example:**



**Expected Results:** This project will help delve me into the world of high availability and help me understand the differences in available open source software solutions that aim to provide this functionality. I expect the results to be extremely proactive in meeting the goals set forth for this project as there are many online articles and testimonials pointing to this solution being worthy in all sorts of high-load environments. My intentions are to document my creation of this environment and provide it publically so others can follow in the same path depending on their needs.

***By Mario Loria of Michigan Technological University for the Web Application & Server Administration undergraduate CNSA course***