## Architecture Summary

The diagram at the right shows how the systems are currently configured. The key technologies involved are:

* VMware – All systems are virtual servers, created from a single template to unify the base OS layer, the core configuration, and system security.
* Nginx – Acting as a reverse proxy, Nginx is running on the only systems that are accessible via public IP addresses outside of the UNH network. Nginx passes requests on to appropriate application server stacks based on the details of the URL requested.
* “LAMP” – Our core web technology is the Linux, Apache, MySQL, and PHP combination that is widely used today.
* Drupal – New web development is done in the Drupal 7 content management system. We also have older sites that were developed in Drupal 6 so we support both versions.
* ColdFusion – We have legacy applications that were developed in ColdFusion that we still support today. Most of these use Microsoft SQL Server as their back-end database.
* NFS – A single NFS file share is mounted on all our application servers allowing for sharing of the web site content across replica servers
* Memcached – The latest addition to our systems, memcached was added to address a performance issue that came to light when testing a new Drupal 7-based website for the College of Liberal Arts that made heavy use of the “Organic Groups” module. Memcached is a memory-based key-value store that integrates with Drupal and greatly reduces the need to directly query back-end databases.

Our system also utilized another feature not illustrated here. We have worked with our network operations staff to setup local domains within our DNS system for unh.dev and unh.test. These domains mean that we can replicate our production servers, have them keep the same names, and then use DNS name resolution and host files to manage the three different environments (development, test, and production) with essentially identical servers. With the one exception of the public IP address to be used for the web site domain, Nginx , Apache , and Drupal configuration files remain unchanged throughout the development cycle.

## Traffic Routing

The key aspect of our architecture is the application layer traffic routing that takes place within Nginx. The configuration for Nginx allows different “upstream” systems. These can be individual servers or, more commonly, sets of replica servers for load balancing, redundancy, or failover. Each upstream target is assigned a named designation.

Another part of the Nginx configuration defines server sockets (IP addresses and ports) that Nginx will listen to along with regular expressions that will match strings in the URL that is in the request. Each regular expression pattern can be associated with a different upstream target name.

The result of this is that Nginx will take incoming requests and direct them to the appropriate application server for that specific URL. It even means that within a given site, we can have multiple technologies used. The main [www.unh.edu](http://www.unh.edu) site, for example, is now in Drupal 7, but there are specific URLs within that site that are based in Drupal 6, ColdFusion, or just static HTML. These even can be nested inside each other. Another example is our IT site at [it.unh.edu](http://it.unh.edu). That site is based in a legacy content management system written in ColdFusion. Some new parts of the site have been developed in Drupal 7, such as <http://it.unh.edu/resources/>. So, we can direct those newly built parts to the Drupal systems while still leaving the old site intact.

We also use these routing capabilities to separate core UNH websites from ones hosted for outside groups or situations where we have outside web developers who need access. The diagram above references a “Client Stack”. This is identical to our normal Apache/PHP/Drupal 7 environment, but on separate virtual servers so any negative impacts from these sites that we do not fully control are isolated and cannot take down UNH’s main sites.

## Development Features

Having replica systems for development and testing work has proven to be a tremendous advantage for our web developers. They know that if something works properly on a dev system it will also work in production. We also use a naming scheme to make transitions from dev to test to prod simple. In most cases, no code changes are necessary. This is due to using just the host name such as “fe-php53-01” to identify systems. Then, it does not matter if the actual system is “fe-php53-01.unh.dev” for development, “fe-php53-01.unh.test” for testing, or “fe-php53-01.unh.edu” for production.

The local domains also allow us to control access to the dev and test stacks. On the UNH campus, users can enter a unh.test URL, and standard DNS will resolve it. No special host files or spoofing is required. That means it is simple for content owners in various departments to review their new sites in test and fully edit content prior to the site being accessible to the public. By using Subversion as a content repository, we can easily move sites between these environments, from development to test to production. At the same time, there is no limit on the number of sites we can keep in the dev or test systems. Previously, we had specific IP addresses associated with sites that were being built, meaning that only as many sites as there were IP addresses available could be worked on at a time. This old practice decreased the agility of our development process since some changes required development sites to be redeployed, replacing ones already in existence.

## We want to hear from you!

Part of the reason for this presentation at the NERCOMP Annual Conference is to reach out to other college and university IT web systems administrators. We would welcome the chance to share ideas, collaborate on new solutions, and discuss issues and concerns that are common to all of our campuses.

If you run web servers at your school, please drop us an e-mail and let us know who you are and your areas of interest. If you do not manage the servers yourself, we’d appreciate you sharing this document with the right people at your school so that they could get in touch.

Thanks!

David J. Blezard, Senior IT Manager david.blezard@unh.edu

Chris Schneider, Systems Administrator chris.schneider@unh.edu