WEB PRESENCE BY 5PM

By

Jason Pepin
November 20, 2002

Table of Contents

Introduction

My name is Jason Pepin and I am a Unix Administrator with CNA Life Insurance Company in Nashville, Tennessee. Part of my job responsibilities includes web server administration, routine maintenance, and integration with other technologies. Many administrators, myself included, are faced with various challenges everyday. Some include implementing various solutions within very limited time; many times Management expects the task to be completed by 5pm. **WEB PRESENCE BY 5PM** is a presentation aimed at entry-level web administrators and is based off the scenario discussed below.

Scenario

You arrive at work at 8am and find your boss waiting for you at your desk. Your boss says to you, "Johnson, before you leave today I would like for the company to have a presence on the Internet. Advertising has created some static html pages and our company name has been registered as msapgolf.com. Oh, and I also need to have the ability to get some kind of report that shows how many hits we get and who visits our site and so on. Well Johnson you better get moving!"

Let's assume you already have a Unix system built and ready for use. You will need to download the following software.

Apache	www.apache.org
Cronolog	www.cronolog.org
Webalizer	www.webalizer.com

For this presentation I will be using the following components. All components have been downloaded to /root/software.

Operating System:	Red Hat 7.3
Apache:	apache_1.3.26.tar.gz
Cronolog:	cronolog-1.6.2.tar.gz
Webalizer:	webalizer-2.01-10-linuxelf-x86-bin.tgz

NOTES:

Apache Installation

First we will look at setting up the Apache web server. Uncompress the Apache source code, *zcat apache_1.3.26.tar.gz* | *tar xf -*. Change to the apache_1.3.26 directory and follow the steps below to perform a simple installation.

```
#./configure --prefix=/usr/local/applications/apache1.3 \
#>--mandir=/usr/local/man
#make
#make install
#make clean
```

That's it! Let's examine what we just did. By using the --prefix option, we defined which directory Apache would be installed into. Also by using the mandir option we are able to tell the configuration to install the Apache man pages into /usr/local/man. If you would like to see a full listing of the available configuration options before you configure Apache use:

```
#./configure --help
```

Before we start our web server we need to change a couple of parameters in the main Apache configuration file, *httpd.conf*. Change to /usr/local/applications/apache1.3/conf and vi the *httpd.conf* file. Scroll down to the *ServerAdmin* parameter and change the value to the e-mail address of your Web Administrator. Next, find the *ServerName* parameter and change the value to the name of your domain, which in our case is www.msapgolf.com. Save the file and change to the bin directory, *cd ../bin*. Before we start the web server let's verify that the syntax in the *httpd.conf* file is correct. Issue the following command to complete that process.

```
#./apachectl configtest
```

NOTES:

You should get back a response of *Syntax OK* if all the syntax is correct. If something is incorrect you will receive an error telling you which line is incorrect. Now type the following command to start the web server:

```
#./apachectl start
```

Open your browser and enter http://www.msapgolf.com. You should see the Apache main page.

NOTE: Remember to copy your custom html files into Apache's *htdocs* directory before you go live.

When the Apache web server starts for the first time it creates an access_log file and an error_log file. The access_log file captures all the requests made to the web server. The error_log file logs errors the web server encounters. For our project we want a way to rotate the logs on a monthly basis without having to stop the web server, rename the log files, and then restart the web server so new log files appear. Apache has a log rotation program called *rotatelogs* but many people suggest using the *Cronolog* program.

NOTES:

Cronolog Installation

Change to the /root/software directory and uncompress the Cronolog source code, *zcat cronolog-1.6.2.tar.gz* | *tar xf* -. Change to the cronolog-1.6.2 directory and follow the steps below to build the software.

```
#./configure \
>--mandir=/usr/local/man
#make
#make install
#make clean
```

Done! The steps above built and installed the *cronolog* program into /usr/local/sbin and the man pages into /usr/local/man. Now let's look at integrating Apache and Cronolog so that the Apache log files will automatically rotate on a monthly basis without any interruption to the web server. Change to the /usr/local/applications/apache1.3/conf directory and use vi to edit the *httpd.conf* file. Find the line that reads:

```
ErrorLog /usr/local/applications/apache1.3/logs/error_log
```

Modify the line to read as follows:

```
ErrorLog "|/usr/local/sbin/cronolog
/usr/local/applications/apache1.3/logs/%m%Y.error_log"
```

This will pipe error messages through the cronolog program and into a file dated month.year.error_log, (Example: 102002.error_log). Now perform the same change for the access_log file. Find the line that reads:

```
CustomLog /usr/local/applications/apache1.3/logs/access_log common
```

Modify the line to read as follows:

```
CustomLog "|/usr/local/sbin/cronolog
/usr/local/applications/apache1.3/logs/%m%Y.access_log" common
```

NOTES:

In order for your changes in the *httpd.conf* file to take effect, you will need to restart the Apache web server. Before you restart your web server it is a good idea to verify that the changes you just made to the *httpd.*conf file are ok. Remember you can use the following command to verify the contents of the *httpd.*conf file.

```
#/usr/local/applications/apache1.3/bin/apachectl configtest
```

After restarting the web server you should see two new log files in the logs directory, 102002.access_log and 102002.error_log.

Two tasks are now complete and it is not even lunchtime yet.

NOTES:

Webalizer Installation

Webalizer is a log analyzer program that will read the web server access log and create a graphical report with various web statistics. Change to the /root/software directory and uncompress the Webalizer binary package, *zcat webalizer-2.01-10-linuxelf-x86-bin.tgz* | *tar xf -*. Copy the *Webalizer* executable to /usr/local/bin and the *sample.conf* to /usr/local/applications/apache1.3/conf. I like to rename the *sample.conf* to the name of the web site (example: *msapgolf.conf*). Now we need to edit the *msapgolf.conf* file and set some parameters. Go to /usr/local/applications/apache1.3/conf and use your favorite editor to edit the file. The first parameter to modify is the *LogFile* parameter. Remove the comment mark and change the location of the log file to match the absolute path to your Apache access log file. Look at the example below.

```
LogFile        /usr/local/applications/apache1.3/logs/102002.access_log
```

The next parameter to modify is *LogType.* All you need to do is remove the comment mark from the line that reads *LogType clf.* Next is the *OutputDir* parameter. This is location where Webalizer will generate the output files. I like to have the output dumped to a directory called webreport located in Apache's *htdocs* directory. Don't forget to go to /usr/local/applications/apache1.3/htdocs and type *mkdir webreport* before running the Webalizer program.

```
OutputDir      /usr/local/applications/apache1.3/htdocs/webreport
```

Next, we have the *HistoryName* parameter. This parameter allows you to define a file that would keep track of twelve months worth of log file data. Remove the comment mark in front of *HistoryName.* The next two parameters are separate in the configuration file but are used in conjunction with each other so let's look at them together. The first one is *ReportTitle*. Remove the comment mark from the beginning of the line. The second parameter is *HostName*. Remove the comment from the beginning of the line and replace the word localhost with the name of your web site. Look at the example below.

```
ReportTitle    Usage Statistics For
HostName       www.msapgolf.com
```

NOTES:

Next, remove the comment mark from the *HTMLExtension* parameter so that the generated report files are in html format. The next parameter to modify is *DNSCache*. This parameter allows you to define a file that will be used for reserve DNS lookups. If you do not want the hostnames of that addresses visiting your site then just leave the comment mark in place otherwise remove it. For the purpose of this demonstration we will remove the comment and we will rename *dns_cache.db* to *msapgolf.db*. This keeps things consistent to the naming scheme of the web site. The next parameter to set is *DNSChildren*. Remove the comment and change the value from 0 to 5. This will spawn five children process used for performing the reverse DNS lookups.

```
DNSChildren	5
```

The remainder of the parameters can be left as is for the creation of a simple report. You can alter each parameter according to your requirements. Now, we need to configure a way to have the reports generated. Below is a simple script that will handle this function. Since root will own and run the script let's put the script under the /root/scripts directory. Here is the content of the simple script called *runWebalizer.sh*.

```
#!/bin/sh
#The name of this script is runWebalizer.sh.
#This script will kick off the Webalizer program.
#
/usr/local/bin/webalizer –c /usr/local/applications/apache1.3/conf/msapgolf.conf
exit 0
```

Next, create a cron job so that the reports are automatically generated daily. Edit root's cron tab and enter the line below. This will execute the webreport script one minute after midnight everyday.

```
01 0 * * * /root/scripts/runWebalizer.sh
```

To view your report, open a web browser and enter the URL, http://www.msapgolf.com/webreport. It should return a page titled, *Usage Statistics for www.msapgolf.com* and a chart showing the month and some basic web statistics. If you click on the month hyperlink it will expand the report to show you several different statistics. You can alter the output of the report by editing the msapgolf.conf file.

NOTES:

Technically at this point you have completed the tasks assigned to you by your boss but let's add one more thing to the mix. Let's look at how to restrict access to the webreport directory by requiring the input of a username and password. Change to the /usr/local/applications/apache1.3/bin directory and execute the following command.

```
#./htpasswd –c /usr/local/applications/apache1.3/.htpasswd webuser
password: password
re-enter password: password
```

This will create a user named *webuser* in a file called *.htpasswd*. Next, change to the /usr/local/applications/apache1.3/conf directory and edit the *httpd.conf* file. Find the section that reads *Controls who can get stuff from this server* and right below the closed directory tag, </Directory> enter the following information.

```
# This section will restrict access to the webreport directory
<Directory /usr/local/applications/apache1.3/htdocs/webreport>
  AuthType basic
  AuthName "Restricted Access"
  AuthUserFile /usr/local/applications/apache1.3/.htpasswd
  Require valid-user
</Directory>
```

When a user requests http://www.msapgolf.com/webreport they will see a pop up box that requires a username and password. Enter the username *webuser* and the password *password*. The webreport for www.msapgolf.com should then be displayed.

That's it, your done and before 5pm. Your boss will be excited and impressed by your quick ingenuity.

NOTES:
