Auto restart Apache on High server load (bash shell script) - Fixing Apache server temporal overload issues

Author : admin

I've written a tiny script to check and restart, Apache if the server encounters, extremely high load avarage like for instance more than (>25). Below is an example of a server reaching a very **high load avarage:**;

```
server~:# uptime
13:46:59 up 2 days, 18:54, 1 user, load average: 58.09, 59.08, 60.05
load average: 0.09, 0.08, 0.08
```

Sometimes high load avarage is not a problem, as the server might have a very powerful hardware. A high load numbers is not always an indicator for a serious problems. Some 16 CPU dual core (2.18 Ghz) machine with 16GB of ram could probably work normally with a high load avarage like in the example. Anyhow as most servers are not so powerful having such a high load avarage, makes the machine hardly do its job routine.

In my specific, case one of our **Debian Linux servers** is periodically reaching to a very high load level numbers. When this happens the Apache webserver is often incapable to serve its incoming requests and starts lagging for clients. The only work-around is to stop the Apache server for a couple of seconds (10 or 20 seconds) and then start it again once the load avarage has dropped to less than "3".

If this temporary fix is not applied on time, the server load gets increased exponentially until all the server services (ssh, ftp ... whatever) stop responding normally to requests and the server completely hangs ...

Often this server overloads, are occuring at night time so I'm not logged in on the server and one such unexpected overload makes the server unreachable for hours.
To get around the sudden **high periodic load avarage server increase**, I've written a tiny bash script to

monitor, the server load avarage and initiate an Apache server stop and start with a few seconds delay in between.

```
#!/bin/sh
# script to check server for extremely high load and restart Apache if the condition is matched
check=`cat /proc/loadavg | sed 's/\./ /' | awk '{print $1}'`
# define max load avarage when script is triggered
max_load='25'
# log file
high_load_log='/var/log/apache_high_load_restart.log';
# location of inidex.php to overwrite with temporary message
index_php_loc='/home/site/www/index.php';
# location to Apache init script
apache_init='/etc/init.d/apache2';
#
site_maintenance_msg="Site Maintenance in progress - We will be back online in a minute";
if [ $check -gt "$max_load" ]; then>
#25 is load average on 5 minutes
cp -rpf $index_php_loc $index_php_loc.bak_ap
echo "$site_maintenance_msg" > $index_php_loc
sleep 15;
if [ $check -gt "$max_load" ]; then
$apache_init stop
sleep 5;
$apache_init restart
echo "$(date) : Apache Restart due to excessive load | $check |" >> $high_load_log;
cp -rpf $index_php_loc.bak_ap $index_php_loc
fi
fi
```

The idea of the script is partially based on a forum thread - **Auto Restart Apache on High Load** - *http://www.webhostingtalk.com/showthread.php?t=971304*Here is a link to my restart_apache_on_high_load.sh script

The script is written in a way that it makes two "if" condition check ups, to assure 100% there is a constant high load avarage and not just a temporal 5 seconds load avarage jump. Once the first if is matched, the script first tries to reduce the server load by overwritting a the *index.php, index.html* script of the website with a one stating the server is ongoing a maintenance operations.
Temporary stopping the index page, often reduces the load in 10 seconds of time, so the second if case is not necessery at all. Sometimes, however this first "if" condition cannot decrease enough the load and the server load continues to stay too high, then the script second if comes to play and makes apache to be completely stopped via Apache init script do 2 secs delay and launch the apache server again.

The script also logs about, the load avarage encountered, while the server was overloaded and Apache webserver was restarted, so later I can check what time the server overload occured.
To make the script periodically run, I've scheduled the script to launch every 5 minutes as a cron job with

the following cron:

```
# restart Apache if load is higher than 25
*/5 * * * * /usr/sbin/restart_apache_on_high_load.sh >/dev/null 2>&1
```

I have also another system which is running FreeBSD 7_2, which is having the same overload server problems as with the Linux host.
Copying the **auto restart apache on high load script** on FreeBSD didn't work out of the box. So I rewrote a little chunk of the script to make it running on the FreeBSD host. Hence, if you would like to **auto restart Apache or any other service on FreeBSD server** - get /usr/sbin/restart_apache_on_high_load_freebsd.sh my script and set it on cron on your BSD.

This script is just a temporary work around, however as its obvious that the *frequency of the high overload will be rising with time* and we will need to buy new server hardware to solve permanently the issues, anyways, until this happens the script does a great job :)

I'm aware there is also alternative way to *auto restart Apache webserver on high server loads* through using **monit** - *utility for monitoring services on a Unix system*. However as I didn't wanted to bother to run extra services in the background I decided to rather use the up presented script.

Interesting info to know is Apache module mod_overload exists - *which can be used for checking load average*. Using this module once load avarage is over a certain number apache can stop in its preforked processes current serving request, I've never tested it myself so I don't know how usable it is. As of time of writting it is in early stage *version 0.2.2*
If someone, have tried it and is happy with it on a busy hosting servers, please share with me if it is stable enough?