Designing a PHP Diagnostics Page for a Tier-3 Open Science Grid Computer Cluster

Kim Day

Overview

The High Energy Physics A group at Florida Tech currently maintains a Tier-3 Open Science Grid cluster for running simulations. The diagnostic information on the cluster is stored on two different websites. The first page is located at the CMS dashboard website and shows the result of SAM, or Service Availability Monitoring tests. These tests are run at regular intervals on the cluster to ensure that it is up to standard. The second page is located on the cluster itself and shows the results of Ganglia diagnostics. These contain graphical data for each computing node as well as Load, CPU, Memory, and Network data that can be displayed for the past hour, day, month, and year.

Because it was necessary to check through both websites in order to obtain a status update, a third website was needed that could display the data from both. This website would also need to refresh automatically every five minutes so that it could be left open on a computer monitor for quick viewing of the cluster's status. Ideally, it would also provide the option of which time intervals of Ganglia graphs to load.

Features

The current diagnostics page uses the PHP function file_get_contents() to fetch the diagnostic data from the SAM and Ganglia websites. Data on the diagnostics page retains the basic formatting and hyperlinks from the original page. The graphs and SAM tests displayed on the page will lead to another page when clicked, much like they did on their original pages. Clicking on the graphs will show a zoomed-in version with more detail. Clicking on a SAM test result will lead to a page containing information on that SAM test if the user has the appropriate certificate. The page will also refresh every five minutes to keep the data as up-to-date as possible.

Description of Files

The diagnostics page uses a combination of HTML, PHP, and CSS. The necessary files are:

diagnostics.php	- Contains the php and html code that generates the html
style.css	- Contains the css for formatting the html

The diagnostics.php file manages the page's content, which includes the HTML for the SAM tests, Ganglia graphs, toolbar, and general information (hosts up/hosts down, etc.). The style.css file uses CSS to manage the way the content is formatted, such as the font colors, background images, image sizes, and navigation bar colors. These files are located at /var/www/html. Most of the coding for these files was done in the vi editor.

Functions and Methods Used for the Diagnostics Page

Obtaining Diagnostic Results from SAM and Ganglia

The PHP function file_get_contents() was found to be best for fetching the diagnostic data. Using the function with a webpage URL as an argument returns a string containing the html source code of the webpage. This method was used to create two variables, $sam and $ganglia, which contain the HTML code for the SAM and Ganglia diagnostic tests.

Deciding which Ganglia page to load

The page uses the special variable GET to get the arguments in the url. These are the words following '*.php?'* in the URL.

This statement is used to obtain the URL argument:

```
isset($_GET[time]) && $_GET[time] != ""
```

Here the program is checking to see if the variable "time," set by the URL, has a value and if it is not a blank string. The function 'isset' first ensures that _GET has a value. For example, adding "?time=string" to the end of the URL will declare the variable "time" and set it to "string." The rest is a series of 'if' statements checking if 'time' contains day, month, or year. Then the program decides the appropriate Ganglia URL to read. If 'time' is not equal to any of those strings, then the program uses the default ganglia page, which is organized by load rather than node.

Formatting HTML with str_replace

The HTML for the SAM tests could not be printed directly to the diagnostics page because the table was originally formatted for a white background instead of a black background. The PHP function "str_replace" was used to edit the data in the $sam string variable. This was used to replace all instances of "FFFFFF" (white) with "000000" (black) for the backgrounds. CSS was used to do the rest of the formatting for the table.

The str_replace function also allowed some of the words in the table to be abbreviated to allow the table to better fit onto the page. For instance, "uscms1.fltech-grid3.fit.edu" was replaced by "CE" to stand for the Computing Element.

Printing An Error Statement

The program "detects" SAM test errors by using the "strpos" function. The strpos function returns the position of a certain string inside a larger one if the smaller string can be found or a negative number if it was not. The SAM tests will change to "error" if and only if a test fails. Therefore, running strpos("error", $sam) will return a negative number if no errors are listed in the tests and a positive number if there is one or more errors. The PHP file checks this value, and if it is indeed positive, can echo an error message onto the screen.

Grabbing Text From the Ganglia Webpage

The text sequence from ganglia containing the number of CPUS, hosts, etc. is always exactly 381 characters long in the source code. Because of this, the “substr” method combined with strpos can be used to extract only this portion of the HTML.

Since the desired text from $ganglia starts with a unique series of characters, strpos can be used to find the start position of the HTML string. Then the function substr is used to read in a string, start position, and length to return only the portion of the Ganglia webpage which is needed for the diagnostics page. This text was also formatted with str_replace to add spaces and breaks.

Overview graph information on Ganglia Page

There are also four overview graphs on the ganglia page. These were also taken from the $ganglia variable with a combination of strpos and substr.

First, strpos was used to find the start and end position of the image url, knowing that the url begins and ends with a certain unique string of characters. These values are used to calculate the length of the string to extract Then, substr is used to extract the url of that image and it is stored into a PHP variable and echoed to the page. This is repeated four times.

This method works because strpos has an optional integer argument that tells the function where to start searching from. Therefore, if the end of the previous URL is set as the optional argument, strpos will find the position of the next graph.

So in summary, the steps are:

1. Find the start position of the graph image.
2. Find the end position of the graph image.
3. Use the start and end positions to calculate a string length.
4. Get the substring with that start position and length.
5. Echo the substring.
6. Set the new "start" index to start searching at to the end of the previous substring.
7. Repeat four times to get all four graphs.

Node graph information on Ganglia Page

A similar method to extract the graph information was also used for the graphs of the nodes. First, the number of working nodes was grabbed from the general information substring. This is because if a node goes down, it is not displayed on Ganglia. The searching algorithm described above was repeated for each node. However, while looping there is an additional statement that adds a new line for every five graphs.

Important PHP Functions Used

file_get_contents

string file_get_contents (string $filename [optional args])

When given a url or filename, returns the contents of that file as a string variable. Used to get the contents of the ganglia and sam pages.

strpos

int strpos(string $haystack, mixed $needle [, int $offset = 0])

When given a string and a substring, returns the position of that substring within the string or negative one if it does not contain the substring. Used to find the positions of certain strings so that they may be extracted with substr.

An optional argument 'offset' will tell the function where to begin looking in the string. The default value is 0 (start from the beginning).

str_replace

mixed str_replace(mixed $search, mixed $replace, mixed $subject [optional args])

When given a string and two substrings, replaces all instances of the first substring with the second substring. Used in this program to edit the contents of the text from using file get contents.

substr

string substr(string $string, int $start [, int $length])

When given a string and two integers, returns the substring from the first integer as a starting index to the second integer as the length.