getaddrinfo() thread failed to start on IBM i PHP cURL calls in CL programs

Issue

Starting with version 2020 of Zend Server, interactive php-cli calls to curl_exec() can show this message in the curl_error() output:

```
getaddrinfo() thread failed to start
```

Environment

Any 64-bit version of PHP running on any supported version of IBM i. 64-bit PHP is provided in Zend Server version 2020 and higher, and in any version of ZendPHP running on IBM i.

Details

The 64-bit version of curl running on IBM i is multithreaded. This is a change from the 32-bit version that was used with versions 2019 and older of Zend Server. CLP calls to PHP scripts usually rely on QSHELL (QSH) calls, or QP2SHELL calls. Existing CL programs that run PHP scripts using curl_exec(), that ran fine in older versions of Zend Server, may suddenly start to fail in the newer 64-bit versions. Customers new to Zend Server that started on a 64-bit version, may successfully create some PHP-CLI scripts that run fine, then discover that their script running curl_exec() fails when run via a CLP call.

Resolution

The method used to run PHP with multithreading from a CLP depends on whether the CLP uses QP2SHELL or QSHELL (QSH) to run the PHP script.

Resolution for QP2SHELL

If your CLP calls QP2SHELL or QP2SHELL2 to call php-cli, you can create a very simple shell script to call the php-cli executable binary, and change your CLP to use this shell script. Please refer to the following article to see how to do it:

PHP-CLI launches a terminal from QP2SHELL on IBM i

Resolution for QSHELL (QSH)

Enable multithreading in QSHELL by adding an environment variable before your QSH command:

```
ADDENVVAR  ENVVAR(QIBM_MULTI_THREADED) VALUE(Y) REPLACE(*YES)
```

A good place for this command is right after your variable declarations (DCL) in your CL program. Just add the ADDENVVAR command to your CLP and compile it.

You can learn more about customizing the QSHELL environment here: Customizing the Qshell environment

You can learn more about the environment variables for QSHELL here: Shell Variables

Alternate resolution using ALWMLTTHD

If you are submitting your PHP script to run in a batch job, you can change the job to allow multi-threading.

In your SBMJOB, where you submit the job that runs your CL program, please add this parameter:

ALWMLTTHD(*YES)

For interactive jobs, the ALWMLTTHD parameter can be changed in the JOBD used by the interactive job. This will take effect on the next sign on.

> Note that by default, native mode jobs do not allow multi threading. It may be preferable to use one of the other methods, since native mode programs running in your job may expect multi threading to be off.

Test Script

This is a simple test script you can use to verify curl is working for you.

curl_test.php

```
<?php

// show any PHP errors
ini_set("display_errors", 1);

// create a new cURL resource
$ch = curl_init();

// set URL and other appropriate options
curl_setopt($ch, CURLOPT_URL, "http://www.zend.com/");
curl_setopt($ch, CURLOPT_HEADER, 0);

// grab URL and pass it to the browser
curl_exec($ch);

// Show any curl errors
if(curl_exec($ch) === false)
{
    echo 'Curl error: ' . curl_error($ch);
}
else
{
    echo 'Operation completed without any errors';
}

// close cURL resource, and free up system resources
curl_close($ch);
?>
```