IBM i Zend Website freezes or slows down, or PHP stops responding - how to research

Issue

Sometimes the web site running on Zend Server for IBM i will freeze or lock up, or run much slower than normal, or appear to crash. Restarting Apache seems to clear up the problem, at least until the next time it happens.

These problems can be difficult to analyze. This article tells how to gather some relevant material related to the IBM HTTP Server Apache instance for Zend Server to assist in the analysis of these types of issues.

Environment

Zend Server for IBM i version 6 or higher, running on any supported version of IBM i.

Resolution
Note: Try setting the Apache timeout to clean up locked jobs more quickly.
This article tells how: Set IBM i Apache Timeout to a shorter time to recover from lock ups faster

Naming convention changes

Starting with Zend Server 2020, we use ZENDPHP74 instead of ZENDPHP7 or ZENDSVR6 for the Apache instance name. This article has been updated to use ZENDPHP74, with reminders to use ZENDPHP7 for versions 9 through 2019, or ZENDSVR6 for versions 6 through 8.5.

While it is tempting to just restart Apache as quickly as possible to get the site back up and running, it is important to first take a look while Apache is active but not responding.

The place to start looking is the Work with Active Jobs display to see what the ZENDPHP74 (ZENDPHP7 or ZENDSVR6 for older versions) FastCGI child jobs are doing. Each FastCGI child job runs a copy of the PHP engine. Every request to process a PHP request via the Apache server is sent to one of these children for processing. If any of the children become permanently unavailable, that can slow the site, as requests may have to wait longer for available children when there are fewer of them. If enough of these children are locked up, the web site can become intolerably slow, as there are simply not enough active children left to process all the traffic in a timely manner.

The following steps won't tell the whole story as to why child jobs are locking up, but they do show the lock ups and can tell some important information that can be used for further research, as well as suggest some possible workarounds that can help the situation become less troublesome, even before the issue is fully resolved.

The following example assumes you are running the default ZENDPHP74 (ZENDPHP7 or ZENDSVR6 for older versions) Apache instance only. For customers running additional Apache instances using PHP, if some other instance is locking up, please substitute that instance name for ZENDPHP74 in the examples.

The first step is to wait for the lock up or freeze to occur. This is because we need to see the status of the Apache instance jobs when the problem is occurring. Before restarting Apache to get your web site going again, please take just a couple of minutes to review the status of your Apache jobs. From the 5250 command line:

For version 2020 and higher:

```
WRKACTJOB SBS(QHTTPSVR) JOB(ZENDPHP74)
```

For version 9 through 2019:

```
WRKACTJOB SBS(QHTTPSVR) JOB(ZENDPHP7)
```

For versions 6 through 8.5:

```
WRKACTJOB SBS(QHTTPSVR) JOB(ZENDSVR6)
```

You should see a list of jobs similar to this one:

```
Subsystem/Job  User        Type  CPU %  Function        Status
   ZENDPHP74    QTMHHTTP    BCH      .0  PGM-QZHBMAIN     SIGW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-QZSRLOG      SIGW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-QZSRLOG      SIGW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-QZSRHTTP     SIGW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-zfcgi        SELW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   THDW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
   ZENDPHP74    QTMHHTTP    BCI      .0  PGM-php-cgi.bi   TIMW
```

Again, these are the jobs for the ZENDPHP74 (ZENDPHP7 or ZENDSVR6 for older versions) instance of Apache, and the Apache configuration is set to the defaults as shipped by Zend. You may see some different jobs if your configuration has been modified.

The first six jobs are not child processes. If any of these are in any kind of error status, then you should try to get a job log for it. If there is a MSGW status, use option 7 to view the message and respond. If your configuration is changed, there may be more or less than six jobs that are not child processes. These will include any jobs running a program starting with 'Q', the job running program zfcgi, and the first job running php-cgi.bi. The normal status for these jobs is shown in the list above. Again, if any of these jobs is doing anything abnormal, try to get a job log for it.

The remaining jobs running php-cgi.bi are the child processes. Each of these processes runs an instance of FastCGI and PHP. A normal status for these jobs when idle is TIMW. If they are running a script, a normal status is TIMA. You may see them briefly in some other status, but wait a few seconds and then use F5 to refresh the display, and they should eventually settle back into a TIMW or TIMA status. If they seem stuck in some other status like CNDW or DEQW, or anything else, and stay that way even after refreshing the display a few times over two or three minutes, they are stuck and that is why your web site is not responsive. Try to get job logs for any child process that appears to be stuck.

Also, please take screen prints or copy the screen as text, to show all of the ZENDPHP74 (ZENDPHP7 or ZENDSVR6 for older versions) jobs and the status for them.

During this time while the web site is non-responsive, please also check your Apache Threads. It is important to see how many threads are in use during the incident. This article tells how to do it:

Adjust Apache threads on IBM i to improve performance

After collecting your job logs, taking your screen shots, and checking the Apache threads, please go ahead and restart Apache to get your web site going again:

Restart Apache on IBM i

ZENDPHP74 (ZENDPHP7 or ZENDSVR6 for older versions) jobs that will not end may be of interest

If any of the ZENDPHP74 (ZENDPHP7 or ZENDSVR6 for older versions) jobs stubbornly persist, these jobs will be of interest. First make a note of the status for the jobs, which you can do with a screen shot. Then, use option 5 to display a job, then option 10 on the Work with Job menu to display the joblog. Use F10 to show all messages and F18 to navigate to the bottom. You will often see the job is connected to a QSQSRVR job, and you can display that job to see if it is failed or if it is still running. This can be an important clue as to why your web site may be unresponsive.

As soon as you can after restarting the web site, please also obtain your Apache logs and run a Support Tool and get the output. If Zend Support is assisting you with the analysis, we will want to see all of these:

Job logs
Screen shots
Apache Logs
Support tool

We already discussed how to get the job logs and the screen shots. These articles can help you get the Apache Logs and Support Tool:

How to send your IBM i Apache logs to Support

How to run the Support Tool on IBM i

This probably seems like a lot of material to collect, and it is. The reason for this is because the only time any information can be collected is when an incident is happening, so it pays to be thorough. All of this may not be needed to analyze every issue, but it is difficult to predict which of these things might be needed until after the analysis is done. So, it is best to get everything.

Finally, chances are that if you have PHP scripts locking up, someone using your web site may notice the issue. If you have complaints of a page that fails to respond, that can be an important clue as to which script is locking up.

Possible fixes and workarounds

Analyzing why your web site stopped responding can be difficult because there are so many possible causes, some of which may not be obvious from looking at the logs. Another approach could be to consider some of the possible causes to see if any might apply to your site. You can also try some workarounds that can help make the outages less frequent. Here are some suggestions for you to explore.

Set Apache timeout to clean up locked jobs more quickly.

This workaround is also mentioned in the note at the start of the prior section. As distributed, Zend sets the Apache timeout very high so as not to conflict with the FastCGI and PHP timeouts. You can set this timeout to be much shorter, so that Apache can possibly end and respawn a frozen or zombie child process, when PHP is not able to do it.

This article tells how: Set IBM i Apache Timeout to a shorter time to recover from lock ups faster

Increase FastCGI child processes

This workaround can help if the problem is that PHP child processes are locking up over time, or if there is a need for more PHP children to handle the work load on your site.

This article tells how: Increase FastCGI child processes on IBM i

Verify valid DNS entries in the TCP configuration

This issue should not cause your web site to stop responding, but it can make it much slower, which can aggravate other conditions that might contribute to a freeze.

This article tells how make sure your DNS settings are valid: Invalid DNS slows Zend Server UI on IBM i

Check and adjust Apache threads

This issue is not very common, but it can cause your web site to slow down or stop. One possible symptom of this issue might be missing entries in the access log during the incident.

This article tells how check your threads and adjust as needed: Adjust Apache threads on IBM i to improve performance

Disable code optimization

Code optimization can very occasionally cause a problem with some script that can be very hard to trace. Disabling optimization is a quick way to test for optimization issues. For IBM i customers, we find that disabling optimization often has no noticeable effect on performance. Byte code caching remains in effect, and caching is where most of the performance benefit is found. It is also a good idea to set consistency checking on.

This article tells how to do it: Disable optimization for stability issues

Block search bots and other intrusive traffic (Denial of Service)

One reason your site can stop responding has nothing at all to do with your application. Your web site can be inundated with unwanted requests. When a malicious person does this on purpose to bring your site down, it is called a Denial of Service (DoS) attack. You can also be subjected to highly repetitive spidering by Search Bots hosted in foreign countries, like Baidu, Yandex, and many more. Outages caused by high traffic are usually observable in your Apache access log. To prevent these attacks, you could possibly use Apache rewrites to detect and deflect traffic from certain sites, but this job is probably best handled by your firewall firmware. Please contact your network equipment provider to learn how to do this.

Messages waiting on QSYSOPR message queue

The XMLSERVICE Toolkit provides a way to call RPG programs, run CL commands, and do other things in the IBM i native Library/Object environment. If one of these processes halts waiting for a reply to a message, the child process will sit forever in a CNDW status, until the message is answered and the job is allowed to complete. If that job happens to be running stateful, then no other stateful jobs will process while that one message waits for a response. If that happens, and if the majority of the pages on the site are using the XMLSERVICE Toolkit, it can seem like the whole site has stopped responding. Of course, with error monitoring there is no real reason why any program needs to halt like this, and if it does, there is no real reason that the message has to sit unanswered in the queue. IBM i provides tools to manage all of this. But it happens sometimes anyway, so this is something you can check for: No XML Toolkit jobs will run if one program is stuck in MSGW status

Another way this can cause an apparent site freeze is that some web browsers will resend a request if no response in a certain amount of time, so the user might just be waiting patiently, or may turn away from the computer to do something else, and their browser can sit and repetitively call the same script with the same post variables, that will cause the same RPG program to be called with the same parameters and crash again at the same statement. This can create a freeze even when using stateless jobs.

User submits form multiple times, locking their own session

The submit button on an HTML form stays active after it is clicked. So the user can click it again and send another identical request. Even just a double click that any Windows user will habitually use, can submit the form action twice. An impatient user might click it six or ten or even more times. This can be bad if each click requests a PHP script, especially if the script is long running, which is exactly the kind of script that invites extra clicks. The way most people notice this is happening is if the script opens a session. All the additional requests will wait for the original session lock to clear, and the FastCGI child processes can get used up pretty quick. The php log may show timeouts on session_start() and the Apache error log can show errors like:

```
[error] Can not connect() to /usr/local/zendsvr6/bin/php-cgi.bin (0.0) - [3425] A remote host refused an attempted connect operation.
```

This can get pretty ugly as many users get frustrated by the long waits and start hitting their submit buttons over and over, trying to get their web page to respond. This article offers some discussion and an idea about how to find ways to disallow multiple submissions from the form: IBM i Apache error logs show: [3425] A remote host refused an attempted connect operation

When your site stops responding, you can almost always get it started back up by restarting Apache (but please not before you collect the information mentioned in the first section of this article!) However, you have a menu option available to do a Reset. This will stop Apache, rebuild your Zend IPCs, and then start Apache back up. If you do happen to try restarting Apache, and your site will not come back up, try the Reset. If the Reset works when a simple restart will not, that will be important information to know. It could point to a problem in caching or other Zend components.

Here is how to do the Reset: Reset Zend Server for IBM i

Upgrade Zend Server to the latest release

Upgrading to the very latest release for your version of Zend Server insures that any older known bugs that might contribute to your issue will be fixed. It also will allow Zend Support to refer your issue to Zend R&D for research if that becomes necessary, as end R&D can only research issues on the currently supported releases.

Make sure your IBM i PTFs are current

Bringing your IBM i PTFs up to date insures that any older known bugs that might contribute to your issue will be fixed.

Here is how to make sure you are up to date: Verify your IBM i PTF groups are current

Related articles

- Set IBM i Apache Timeout to a shorter time to recover from lock ups faster
- IBM i Apache HTTP - Server Authentication using IBM i user profiles
- Restart Apache on IBM i
- Increase FastCGI child processes on IBM i
- After updating, Apache start up fails on IBM i, PHP keeps restarting