

Check for locks on PHP files prior to upgrading Zend Server for IBM i

Issue

Sometimes the Zend Server upgrade process fails to upgrade PHP, because PHP is locked by some process.

Environment

Any version of Zend Server, running on any supported version of IBM i.

Resolution

The upgrade instructions for Zend Server include instructions to stop Zend Server before applying the upgrade. Usually, this is enough to prevent locks on components that need to be upgraded. However, sometimes PHP is in use by another process, or by a failed Apache process. This article tells how to verify that the php-cgi.bin and php.bin files are not in use, which is a strong indicator that PHP is not locked and can be updated.

There are two techniques to use. The first is a quick check of the QHTTSPVR subsystem to verify no Apache instances have a lock on php-cgi.bin. The second uses the QP0FPTOS API to verify that no process is using php-cgi.bin and that no process is using php.bin.

For Zend Server versions 2020.x and higher:

Part 1 - Verify no Apache instances have php-cgi.bin locked:

From the 5250 command line, with *SECOFR class authorities:

```
wrkactjob sbs(qhttpsvr)
```

On the Work with Active Jobs display, look under the Function column for every job. If the function is "PGM-php-cgi.bi", then the associated job is using PHP and must be stopped before continuing.

If the job name is ZENDPHP74, and there are no jobs named ZENDPHP74 running any other functions, then end the job. Use option(*immed) if needed, and endjobabn if the job will not end after 10 minutes.

If the entire Apache stack is there, you will see other functions like QZHBMAIN, QZSRLOG, QZSRHTTP, and zfcgi. This is how it looks if the ZENDPHP74 Apache instance is active:

ZENDPHP74	QTMHHTTP	BCH	.0	PGM-QZHBMAIN
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-QZSRLOG
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-QZSRLOG
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-QZSRHTTP
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-zfcgi
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-php-cgi.bi
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-php-cgi.bi
ZENDPHP74	QTMHHTTP	BCI	.0	PGM-php-cgi.bi

There will be more jobs running pgp-cgi.bin, we are just showing one page. If you see any Apache instance running php-cgi.bin like this, it must be stopped. The job name will typically be the instance name. So, to stop the ZENDPHP74 instance, you would use this command:

```
endtcpsvr *http httpsvr(ZENDPHP74)
```

If it is some other instance, then substitute that instance name for ZENDPHP74 in the above command. After you end this other instance, watch the subsystem to make sure any jobs running php-cgi.bin are ended. Use F5 until no more can be found. End any job that will not stop, using option(*immed) or endjobabn if the job will not end after 10 minutes, as needed.

Part 2 - Use the QP0FPTOS API to verify that no process is using a php binary

PHP can be run from other processes besides Apache. The most common case is to run php-cli, which uses the php.bin binary. However, it is also possible to run the php-cgi.bin binary, so you should check for both.

The way to check if an IFS file is in use is to run the QP0FPTOS API with the undocumented function type *LSTOBJREF to show IFS file locks. (This is a handy thing to know how to do, and seems to not be documented in the online manuals at ibm.com. We found it mentioned in a couple of different places on the web, and it does seem to work, so it might be good to make a note of this for future reference.)

The QP0FPTOS API outputs to a spool file. So, you run the command, then display the spool files for your job to review the output. As mentioned, we need to check for two files, so we will run two commands, again making sure to have *SECOFR class authorities so nothing is missed:

```
CALL PGM(QP0FPTOS) PARM(*LSTOBJREF '/usr/local/zendphp74/bin/php-cgi.bin' *FORMAT2)
```

```
CALL PGM(QP0FPTOS) PARM(*LSTOBJREF '/usr/local/zendphp74/bin/php.bin' *FORMAT2)
```

After each command, you may see these messages:

```
The dump request has completed. The latest spooled file QSYSPRT with
label QP0FDUMP has the dump.
```

Next, display spool files for your job:

```
WRKJOB OPTION(*SPLF)
```

You should have two QSYSPRT spool files with the QP0FDUMP label. Look at each of them using option 5. What you hope to see is that the object does not have any references, as in this example:

```
      List Object References (QP0FPTOS *LSTOBJREF *FORMAT2)
5761SS1 V6R1M0 080215                I74SUP4 Wed Jul 8 18:36:00 2023
Object . . . . . : /usr/local/zendphp74/bin/php.bin
Use Count . . . . . : 0
The object does not have any references.
Number of jobs . . . . . : 0
Number of jobs available . . : 0
```

If the object does have references, it will look like this example, which we ran while the ZENDPHP74 Apache is active:

```
      List Object References (QP0FPTOS *LSTOBJREF *FORMAT2)
5761SS1 V6R1M0 080215                I71SUP4 Wed Jul 8 12:54:29 2023
Object . . . . . : /usr/local/zendphp74/bin/php-cgi.bin
Use Count . . . . . : 1
The object does have references.
Number of jobs . . . . . : 11
Number of jobs available . . : 11
```

You can find each job on the listing, like this:

```
-----
Job. . . . . : 177729/QTMHHTTP/ZENDPHP74
```

For each one you find, go find the job and end it. Then, after all the jobs are ended, run the API again to verify there are no more references.

Having done all this, there is of course no guarantee that someone will not come along and start something up that uses one of these files. However, this is about as thorough as you can get to make sure there are no problems with the PHP update.

For Zend Server versions 9.x and higher, including Zend Server 2018.0.x and 2019.0.x:

Part 1 - Verify no Apache instances have php-cgi.bin locked:

From the 5250 command line, with *SECOFR class authorities:

```
wrkactjob sbs(qhttpsvr)
```

On the Work with Active Jobs display, look under the Function column for every job. If the function is "PGM-php-cgi.bi", then the associated job is using PHP and must be stopped before continuing.

If the job name is ZENDPHP7, and there are no jobs named ZENDPHP7 running any other functions, then end the job. Use option(*immed) if needed, and endjobabn if the job will not end after 10 minutes.

If the entire Apache stack is there, you will see other functions like QZHBMAIN, QZSRLOG, QZSRHTTP, and zfcgi. This is how it looks if the ZENDPHP7 Apache instance is active:

```
ZENDPHP7 QTMHHTTP BCH .0 PGM-QZHBMAIN
ZENDPHP7 QTMHHTTP BCI .0 PGM-QZSRLOG
ZENDPHP7 QTMHHTTP BCI .0 PGM-QZSRLOG
ZENDPHP7 QTMHHTTP BCI .0 PGM-QZSRHTTP
ZENDPHP7 QTMHHTTP BCI .0 PGM-zfcgi
ZENDPHP7 QTMHHTTP BCI .0 PGM-php-cgi.bi
ZENDPHP7 QTMHHTTP BCI .0 PGM-php-cgi.bi
ZENDPHP7 QTMHHTTP BCI .0 PGM-php-cgi.bi
```

There will be more jobs running pgp-cgi.bin, we are just showing one page. If you see any Apache instance running php-cgi.bin like this, it must be stopped. The job name will typically be the instance name. So, to stop the ZENDPHP7 instance, you would use this command:

```
endtcpsvr *http httpsvr(ZENDPHP7)
```

If it is some other instance, then substitute that instance name for ZENDPHP7 in the above command. After you end this other instance, watch the subsystem to make sure any jobs running php-cgi.bin are ended. Use F5 until no more can be found. End any job that will not stop, using option(*immed) or endjobabn if the job will not end after 10 minutes, as needed.

Part 2 - Use the QP0FPTOS API to verify that no process is using a php binary

PHP can be run from other processes besides Apache. The most common case is to run php-cli, which uses the php.bin binary. However, it is also possible to run the php-cgi.bin binary, so you should check for both.

The way to check if an IFS file is in use is to run the QP0FPTOS API with the undocumented function type *LSTOBJREF to show IFS file locks. (This is a handy thing to know how to do, and seems to not be documented in the online manuals at ibm.com. We found it mentioned in a couple of different places on the web, and it does seem to work, so it might be good to make a note of this for future reference.)

The QP0FPTOS API outputs to a spool file. So, you run the command, then display the spool files for your job to review the output. As mentioned, we need to check for two files, so we will run two commands, again making sure to have *SECOFR class authorities so nothing is missed:

```
CALL PGM(QP0FPTOS) PARM(*LSTOBJREF '/usr/local/zendphp7/bin/php-cgi.bin' *FORMAT2)
```

```
CALL PGM(QP0FPTOS) PARM(*LSTOBJREF '/usr/local/zendphp7/bin/php.bin' *FORMAT2)
```

After each command, you may see these messages:

```
The dump request has completed. The latest spooled file QSYSPRT with
label QP0FDUMP has the dump.
```

Next, display spool files for your job:

```
WRKJOB OPTION(*SPLF)
```

You should have two QSYSPRT spool files with the QP0FDUMP label. Look at each of them using option 5. What you hope to see is that the object does not have any references, as in this example:

```
          List Object References (QP0FPTOS *LSTOBJREF *FORMAT2)
5761SS1 V6R1M0 080215                I61SUP4 Wed Jul  8 18:36:00 2015
Object . . . . . : /usr/local/zendphp7/bin/php.bin
Use Count . . . . . : 0
The object does not have any references.
Number of jobs . . . . . : 0
Number of jobs available . . : 0
```

If the object does have references, it will look like this example, which we ran while the ZENDPHP7 Apache is active:

```

List Object References (QP0FPTOS *LSTOBJREF *FORMAT2)
5761SS1 V6R1M0 080215 I61SUP4 Wed Jul 8 12:54:29 2015
Object . . . . . : /usr/local/zendphp7/bin/php-cgi.bin
Use Count . . . . . : 1
The object does have references.
Number of jobs . . . . . : 11
Number of jobs available . . : 11

```

You can find each job on the listing, like this:

```

-----
Job. . . . . : 177729/QTMHHTTP/ZENDPHP7

```

For each one you find, go find the job and end it. Then, after all the jobs are ended, run the API again to verify there are no more references.

Having done all this, there is of course no guarantee that someone will not come along and start something up that uses one of these files. However, this is about as thorough as you can get to make sure there are no problems with the PHP update.

For Zend Server versions 6 - 8.5.x:

Part 1 - Verify no Apache instances have php-cgi.bin locked:


From the 5250 command line, with *SECOFR class authorities:

```
wrkactjob sbs(qhttpsvr)
```

On the Work with Active Jobs display, look under the Function column for every job. If the function is "[PGM-php-cgi.bi](#)", then the associated job is using PHP and must be stopped before continuing.

If the job name is ZENDSVR6, and there are no jobs named ZENDSVR6 running any other functions, then end the job. Use option(*immed) if needed, and endjobabn if the job will not end after 10 minutes.

If the entire Apache stack is there, you will see other functions like QZHBMAIN, QZSRLOG, QZSRHTTP, and zfcgi. This is how it looks if the ZENDSVR6 Apache instance is active:

	ZENDSVR6	QTMHHTTP	BCH	.0	PGM-QZHBMAIN
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-QZSRLOG
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-QZSRLOG
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-QZSRHTTP
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-zfcgi
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-php-cgi.bi
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-php-cgi.bi
	ZENDSVR6	QTMHHTTP	BCI	.0	PGM-php-cgi.bi

There will be more jobs running pgp-cgi.bin, we are just showing one page. If you see any Apache instance running php-cgi.bin like this, it must be stopped. The job name will typically be the instance name. So, to stop the ZENDSVR6 instance, you would use this command:

```
endtcpsvr *http httpsvr(ZENDSVR6)
```

If it is some other instance, then substitute that instance name for ZENDSVR6 in the above command. After you end this other instance, watch the subsystem to make sure any jobs running php-cgi.bin are ended. Use F5 until no more can be found. End any job that will not stop, using option(*immed) or endjobabn if the job will not end after 10 minutes, as needed.

Part 2 - Use the QP0FPTOS API to verify that no process is using a php binary

PHP can be run from other processes besides Apache. The most common case is to run php-cli, which uses the php.bin binary. However, it is also possible to run the php-cgi.bin binary, so you should check for both.

The way to check if an IFS file is in use is to run the QP0FPTOS API with the undocumented function type *LSTOBJREF to show IFS file locks. (This is a handy thing to know how to do, and seems to not be documented in the online manuals at [ibm.com](#). We found it mentioned in a couple of different places on the web, and it does seem to work, so it might be good to make a note of this for future reference.)

The QP0FPTOS API outputs to a spool file. So, you run the command, then display the spool files for your job to review the output. As mentioned, we need to check for two files, so we will run two commands, again making sure to have *SECOFR class authorities so nothing is missed:

```
CALL PGM(QP0FPTOS) PARM(*LSTOBJREF '/usr/local/zendsvr6/bin/php-cgi.bin' *FORMAT2)
```

```
CALL PGM(QP0FPTOS) PARM(*LSTOBJREF '/usr/local/zendsvr6/bin/php.bin' *FORMAT2)
```

After each command, you may see these messages:

```
The dump request has completed. The latest spooled file QSYSPRT with
label QP0FDUMP has the dump.
```

Next, display spool files for your job:

```
WRKJOB OPTION(*SPLF)
```

You should have two QSYSPRT spool files with the QP0FDUMP label. Look at each of them using option 5. What you hope to see is that the object does not have any references, as in this example:

```
          List Object References (QP0FPTOS *LSTOBJREF *FORMAT2)
5761SS1 V6R1M0 080215                I61SUP4 Wed Jul 8 18:36:00 2015
Object . . . . . : /usr/local/zendsvr6/bin/php.bin
Use Count . . . . . : 0
The object does not have any references.
Number of jobs . . . . . : 0
Number of jobs available . . : 0
```

If the object does have references, it will look like this example, which we ran while the ZENDSVR6 Apache is active:

```
          List Object References (QP0FPTOS *LSTOBJREF *FORMAT2)
5761SS1 V6R1M0 080215                I61SUP4 Wed Jul 8 12:54:29 2015
Object . . . . . : /usr/local/zendsvr6/bin/php-cgi.bin
Use Count . . . . . : 1
The object does have references.
Number of jobs . . . . . : 11
Number of jobs available . . : 11
```

You can find each job on the listing, like this:

```
-----
Job. . . . . : 177729/QTMHHTTP/ZENDSVR6
```

For each one you find, go find the job and end it. Then, after all the jobs are ended, run the API again to verify there are no more references.

Having done all this, there is of course no guarantee that someone will not come along and start something up that uses one of these files. However, this is about as thorough as you can get to make sure there are no problems with the PHP update.