

Information Technology Company

7389 Lee Highway, Suite 210, Falls Church, VA 22042 --- 703-237-0592

October 31, 2022

ISV zPDT Fixpack GA11.1

IBM recently released a fixpack containing a few changes that might be important to some ISV zPDT users. Briefly, these are as follows:

- Several IBM Z instructions dealing with UTF conversions have been updated. In very rare cases the zPDT emulated instructions could have incomplete operations with an erroneous result. This situation has been corrected by the fixpack. (And these were *very* rare situations!)
- The *acptool* command has been replaced with a new *ap_acptool* command. (The original *acptool* command remains available.)
- A new function has been added to the ISV zPDT product shipping “package” to provide a method of detecting unwanted/invalid changes to the zPDT code. This involves the inclusion of two small fixes with the product package and a new command named *openssl*.
- The “deflate” facility is now active by default.

Comments

The information technology business has become more concerned with the security and integrity of software products as well as many other aspects of computer usage. IBM and ITC are both very supportive of this trend. Starting with this fixpack, IBM will include two small files that enable the user to verify that the ISV zPDT product files have not been altered from their original shipping state. The two files are known as a *sig* file and a *pem* file. Sample names are:

- `z1090-1-11.57.07.x86_64.sig` *(an example of a signature file name)*
- `ga11-public.pem` *(an example of a public certificate file name)*

An example of using the command to verify the integrity of an ISV zPDT product shipment is:

```
openssl dgst -sha256 -verify ga11-public.pem -signature z1090-1-11.57.07.x86_64.sig  
z1090-1-11.57.07.x86_64
```

This command would be entered as a single line instead of the two lines shown here. You should verify the names of the sig and pem files that are included with your IBM zPDT package. The last operand (`z1090-1-11.57.07.x86_64` in the example) is the basic zPDT package name and will change with new releases. There is no requirement to verify the integrity of your zPDT package with this command; the usage is optional. However, if you obtain your ISV zPDT release through some intermediary route, both IBM and ITC suggest that checking the package with this command might be a good idea.

The *acptool* command that shipped with the initial ISV zPDT GA11 release was intended for very specialized usage by the few ISV zPDT users working with skills in this area. In retrospect, the command had some confusing aspects. IBM is furnishing a new ISV zPDT command with this fixpack that essentially does the same functions as the original *acptool* command but in a more convenient manner. The new command is *ap_acptool* and this command name is more compatible with other ISV zPDT commands related to the emulated cryptographic adapter. An outline of the operands of the new command is as follows:

```
ap_acptool -a n            (n is the number of the coprocessor)  
          -d m            (m is the domain number within the coprocessor)  
          --acps=0xXXXX   (a list of ACPS for certain function, separated by commas)  
          --show-role      (display role (domain) properties and each associated ACPS)  
          --disable-acps   (set specified ACPS to 0; must also specify --acps value)
```

```

--enable-acps    (set specified ACPS to 1; must also specify --acps value)
--query-acps     (also requires the --acps operand)
-h              (display help/man text)

```

As has been noted in the IBM documentation, the *acptool* and *ap_apcptool* commands provide a few of the functions that are normally provided by the Trusted Key Entry (TKE) function on a large IBM Z system. Both ITC and IBM strongly suggest not using these commands unless you have existing skills in this area. The IBM documentation for ISV zPDT does not contain any additional coverage for this area. The *help* function or *man* page might contain slightly more information. Also, please note that the older *acptool* command might be deleted in some future ISV zPDT release.

The original ISV zPDT GA11 release indicates that the “deflate” facility is not active. This is changed in fixpack #1. By default (in the fixpack) this facility is now active. The normal documentation for GA11 (IBM publication SG24-8205-06) mentions that the facility is inactive; this is no longer correct. It also mentions that the emulated zEDC feature (another IBM Z adapter that can provide data compression/decompression) is not available in ISV zPDT GA11; this is still correct.

The discussion in the book and on various forums has left some of our users a bit confused. We can try to briefly summarize the situation:

- Some standard z/OS products (and some customer-generated data files) are compressed. The general purpose is to save disk space and to improve effective data transfer rates. The startup of a java program is one example where such files (“some “jar” files) are decompressed.
- While there might be an endless list of methods to compress and decompress data, there are three methods involved in this discussion: (1) software code, (2) the zEDC adapter, and (3) the deflate facility (that involves the DFLTCC instruction).
- The zEDC adapter is not usable with z16-level systems and the current ISV zPDT release emulates the IBM z16 architecture. (IBM provided zEDC emulation with an older z15-level zPDT, even though it should not have been provided then.)
- “Hardware” compression/decompression is normally faster (or much faster) than software that provides the same actions. This means that zEDC (on older systems) and the “deflate” facility on z15 and z16 machines are important for performance reasons.
- If zEDC is not available and the “deflate” facility is not active then z/OS automatically uses its own software code for the compression/decompression being discussed here. In this sense, the missing zEDC and “deflate” facility do not remove any functionality --- they simply impact performance.
- Why was the “deflate” facility not provided in the initial GA11 release? As best we can determine, it is can be a very complex instruction and there were a few specialized situations when the emulated instruction (within ISV zPDT) did not work correctly. This has been resolved in the fixpack.
- ISV zPDT GA11 provided the *dflt* command that can be used to deactivate or activate the “deflate” facility. (As we best understand it, this command should be used after zPDT is started, but before IPLing z/OS.) You should no longer use this command.

October 31

A few tips about current ISV zPDT and ADCD z/OS usage

Our zPDT users have a variable range of z/OS skills. Some of the tips we receive are useless details for some users but can be helpful to others. Please use your own judgement about each of the following “tips.”

Stopping the ADCD z/OS system

Stopping the current ADCD z/OS system is intended to be rather simple:

```

(logoff any TSO sessions, let batch jobs complete, etc)
S SHUT00                                     (enter this z/OS console command)

```

...	(many other z/OS messages appear)
\$HASP99 ALL AVAILABLE FUNCTIONS COMPLETE	(wait for this message)
\$PJES2	(give this command a few seconds to complete)
quiesce	(this puts z/OS in a wait state)

At this point you can do an **awsstop** command in a Linux window to terminate zPDT operation. However, there are at least two details not covered in this example:

- There are many other shutdown procedures in the ADCD z/OS system in addition to SHUT00, For example, SHUTALL, SHUTDOWN, and so forth. These are procedures in ADCD.Z25B.PROCLIB (in our current ADCD version at the time of writing) and the procedures reference the same member names in ADCD.Z25B.PARMLIB. Which procedure you use depends a little on which subsystems (such as CICS, Db2, etc) you use, although the SHUTDOWN version is the most general. The SHUT00 version is the most compact and is a tiny bit faster if the most simple IPL startup was used.
- Sometimes the ALL AVAILABLE FUNCTIONS COMPLETE message does not automatically appear and you are then unable to terminate JES2 with the **\$PJES2** command.

For someone with z/OS systems programming skills this is a minor issue. For an applications programmer this “hang” might be confusing. If you issue the command **\$PJES2** (without waiting for the ALL AVAILABLE FUNCTIONS COMPLETE message) you might see HASP messages like this:

```

ACTIVE  ADDRESS  SPACES
ASID    JOBNAME  JOBID
-----
0040    AXR03    STC2028

```

Your details (ASID, JOBID, JOBNAME) will differ, but the key detail is that a started task named AXR03 is still running and JES2 cannot be ended while this task is still running. A simple solution is to issue a z/OS command such as **C AXR03** and this should then allow the ALL AVAILABLE FUNCTIONS COMPLETE message to appear. This small operational detail sometimes appears with several recent ADCD z/OS releases, although the jobname varies a little (AXR01, AXR02, AXR03, etc). If you do not want to go through this manual step to help with JES2 shutdown, you can simply edit the appropriate members in ADCD.Z25B.PARMLIB and insert **C AXR01**, **C AXR02**, **C AXR03**, etc) commands near the end of the members. It does not matter if you include a few extra AXRxx names.

Secure TSO logons

The recent IBM zPDT documentation describes a method for obtaining a secure TSO logon using the x3270 program.

x3270 L:10.1.1.2:2023 (your IP address is probably different!)

Unfortunately, this does not work with the latest ADCD systems --- at least with a simple environment based on ISV zPDT GA11 and the ADCD z/OS2.5B releases. It results in a message such as *TLS: Host certificate verification failed; EE certificate key too weak (66)*. We suspect this is due to security upgrades in recent z/OS systems and not due to the x3270 program or ISV zPDT problems. (The same problem occurs if you omit the “L:” prepend in the command; it just takes a bit longer to see the failure message.)

We (ITC) suggest that you skip the *Secure (TLS) connections to z/OS* section in the formal ISV zPDT documentation and determine how to define your own setup for a secure connection in the latest ISV zPDT and ADCD z/OS environment.

Messages about CF structure sizes

If you start a parallel sysplex operation with ISV zPDT GA11 and the current ADCD z/OS system you might see z/OS messages about Coupling Facility structures that are too small. The new CFCC version (corresponding to the z16 level of ISV zPDT GA11) has changes and among the changes are increased minimum sizes for many structures.

This can be a complex topic. Recent “guidelines” from IBM suggest a minimum size of 50 MB for a structure, and they can be much larger depending on the particular usage involved. With current memory sizes provided with major IBM Z systems this is considered a most basic minimum value. However, with the PC memory available while running z/VM (needed for the z/VM emulation of a Coupling Facility) plus several z/OS guest systems then careful memory planning might be needed. We suggest you see IBM publication SG24-8386 (page 140) for a practical discussion about CFCC structure sizes.

Using basic CF functions within a zPDT parallel sysplex environment generally works well. However, if (virtual) memory utilization tends to stress the defined zPDT memory allocation or the actual real memory available on the PC, then there could be significant performance impacts.

Split-lock messages

In a few rare cases when IPLing z/OS, messages such as the following have appeared:

```
kernel: x86/split lock detection: #AC: cpu-1/27793 took a split_lock trap at address: 0x6327cd
```

We have seen one of these messages for each emulated CPU. The messages appear only once. zPDT and z/OS continue to run with no visible side effects. Thus far, this situation has appeared with the combination of ISV zPDT GA11, OpenSUSE 15.4, and a Dell server, but it might occur with other combinations. IBM is aware of the issue and is looking into it.