In this issue:
Face to face with business resiliency and problem determination

James Caffrey

Karla Arndt
Our cover subjects, Karla Arndt, a z/OS Predictive Failure Analysis Software Developer from IBM Storage and Technology Group (STG) in Rochester, Minnesota, and Jim Caffrey, Software Designer: System z Core Technologies from STG in Poughkeepsie, New York, have never met face to face. But that doesn’t stop them from collaborating along with Aspen Payton, STG Software Developer, to write our lead story “Predict to prevent: Let PFA change your destiny.”

Collaboration is what it’s all about as we present the themes of this issue: problem determination and business resiliency. Predictive Failure Analysis or PFA is the most recent part of IBM’s problem determination strategy for z/OS to help you predict those “soft failures” that can plague a system and allows you to make decisions about addressing them before they become major problems. It works with IBM Health Checker for z/OS checks to analyze and prevent problems with storage, LOGREC, and, starting with z/OS V1R12, SMF arrival rates.

IBM is committed to making your z/OS environment as healthy and robust as possible, which, after all, is the central focus of business resiliency. This issue also looks at Runtime Diagnostics to simplify the tasks you do when reviewing suspicious symptoms, sort of a super fast sysprog diagnostician that can quickly analyze WTO messages, global ENQs and locks, and CPU usage to head off potential software headaches.

And what would an issue on problem determination and business resiliency be without a discussion of disaster recovery? “It’s EZ to be prepared for anything,” informs you about the V1R12 enhancements to HCD and HMC so you can automatically generate disaster recovery OS configuration sites when you build your IODF.

With a focus on getting to z/OS V1R12 Marna Walle and Professor Migration, Shigeki Kimura, lead you through the migration process with great hints and tips to make the move smoothly. We also have a number of articles on enhancements to z/OSMF for z/OS V1R12 including updates to CEA and new WLM and RMF z/OSMF GUI functions that continue to simplify the management of z/OS systems.

There’s more: an update on IBM Tivoli Directory Server for z/OS and security, an article about the importance of cryptographic coprocessors and a crypto card retrospective, sorting out zFS file system ownership in a sysplex, and vsam reorgs simplified. There’s an article about GR5 and XCF critical member monitoring (another collaboration in z/OS V1R12 to help ward off critical failures), and one on assistance with sizing your log streams. You’ll find a helpful “how to” article on using an IBM Health Checker check to obtain reports on HFS file systems for migration to zFS, another great “Ask Mr. Catalog” column on DFSMS, what’s the latest with IBM InfoPrint Server, and a midnight ride with Paul Revere making his revolutionary announcement on ways of handling usermods for Language Environment installation default runtime options. Whew! This may be a trim issue, but we are really packing it in when it comes to the information you need to do your job.

Collaboration: it’s a good thing for you and your business, and with IBM and z/OS V1R12 working with you to keep your systems healthy and happy you can rest assured you’re in the best hands.

Oh! For those of you who are waiting with bated breath for answers to last issue’s crossword, check out page 51. As always, we welcome your comments, suggestions, critiques, and ideas for future issues of the newsletter.

Drop us a line at newsletter@us.ibm.com.

The Editors
Contents:

8 RedPaper for IBM Health Checker for z/OS

13 Professor Kimura explains z/OS migration to you

17 A new look for z/OSMF

18 Help manage your workloads and performance with enhancements to z/OSMF V112

21 Enhancements to the z/OSMF Incident Log

23 zFavorites

24 Ask Mr. Catalog

26 Mission impossible? ...not with PPRC Manager and FlashCopy Manager!

27 Kiss VSAM KSDS reorganizations good-bye!

29 InfoPrint Server: Still serving your printing needs

30 The height of “z” cuty!

33 Riding the crest of cryptographic history with Crypto Express3

34 Understanding the importance of cryptographic coprocessors

36 Spotting the Bad Apple: XCF and GRS team up to save the bunch

37 z/OS storage fabric discovery and auto-configuration

40 Coding the globe with Unicode

42 Sizing your log stream data sets

43 Sysplex-aware support on a file owning basis

46 obtain a report of mounted HFS file systems for migration to zFS

48 The Boston usermod party

49 Our contributors

51 Clarifications, corrections, and announcements

51 Answer key for z/OS Hot Topics, issue 22 crossword puzzle

Back cover: Got VSAM stripes?
the specialty of Predictive Failure Analysis® (PFA) is detecting a new class of problems—soft failures. A soft failure is a hard-to-detect problem caused by abnormal software behavior, software that is “sick, but not dead.” These soft failure events are serious because they are long lasting, generate different symptoms every time they occur, and are very hard to pin down. By detecting these events, PFA lets you take corrective action before it builds into something more serious—a system outage.

PFA detects soft failures using IBM® Health Checker for z/OS® checks to collect data about your installation. Using this data, PFA constructs a model of the expected or future behavior of the z/OS images and compares it to the expected behavior of a normal system. If the behavior is abnormal, PFA issues a health check exception.

Next, in z/OS V1R11, PFA expanded its focus to look at critical, persistent address spaces and at the entire z/OS image with checks for message arrival rates (PFA_MESSAGE_ARRIVAL_RATE) and frames and slots usage (PFA_FRAMES_AND_SLOTS_USAGE).

Beginning with z/OS V1R12, PFA delivers the system management facility (SMF) arrival rate check (PFA_SMF_ARRIVAL_RATE) adding yet another level to the analysis of damaged address spaces and systems. Also in z/OS V1R12, PFA allows you to further control the behavior of checks by using your personal knowledge of the behavior of your systems.

Checking in with z/OS V1R11
Beginning with z/OS V1R11, two new checks were added to PFA. Both of these checks examine persistent jobs, which are the jobs that begin within one hour after an IPL.

• The frames and slots usage check detects jobs that use an unexpectedly large number of frames and slots that can indicate a virtual storage leak.
• The message arrival rate check detects jobs that produce an unexpectedly large number of write-to-operator (WTO) or write-to-operator-with-reply (WTOR) messages that can indicate a damaged persistent address space or z/OS image.

Frames and slots usage check
Have you ever suspected that a job on your system was leaking virtual storage but had trouble diagnosing the cause? The frames and slots usage check is designed to help you in this tricky situation! Jobs frequently acquire a range of virtual addresses. When the job accesses memory at one of those virtual addresses, the location is backed with real storage frames. The operating system can later move the data from the real frames to a location in auxiliary storage. When a persistent job uses a larger-than-expected number of real frames and auxiliary storage slots, this check generates an exception message with an accompanying report listing the jobs whose usage exceeded the expectation.

Message arrival rate check
PFA can detect a damaged address space or system by comparing the current arrival rate of the WTO messages and WTOR messages that are normalized by CPU usage to the expected rate. This check can monitor and compare four different categories of address spaces. It tracks the ten persistent jobs with the highest arrival rates in a warm-up period that occurs the first time PFA starts. These jobs are referred to as “chatty” jobs and have predictions and comparisons done against them individually. A bonus feature is that if these same jobs are persistent after PFA restarts, they continue to be tracked. If they are no longer persistent, the warm-up will be rerun to find the new “chatty” jobs.

Another category is a group containing all the other persistent jobs. These jobs are not tracked individually but are combined such that a mathematical formula is applied to detect if any jobs in this group might be causing damage to a system. The address spaces that
are not persistent are also modeled and compared as a group as is the total system message arrival rate. All four of these categories can detect a damaged system or address space, and each category issues a report with additional details to help you find the address space causing the burst of messages.

### Checking in with z/OS V1R12

Beginning with V1R12, PFA extends its approach of detecting soft failures by adding the SMF arrival rate check to detect a damaged system or address space. The SMF arrival rate check works similarly to the message rate arrival check by learning which address spaces are interesting to track, normalizing the arrival rate by CPU usage, and allowing you to determine a minimum rate before an exception occurs. There are two types of SMF records: SMF records that are written when the interval has expired (for example, SMF type-30 records) and records that are written when an event occurs (for example, SMF type-61, 65, and 66 records generated by catalog). PFA uses the power of machine learning to exclude the SMF records that are written when an interval expires from the model; instead, it looks at the SMF records that are written when an event occurs.

The combination of the SMF arrival check with the LOGREC arrival rate and message arrival rate checks provides a multi-tiered approach for PFA to detect a damaged system or address space. This approach came from our review of outage reports from z/OS customers and by academic research on other platforms.

### But, wait! What if your system isn’t normal?

By providing “supervised learning” support beginning in z/OS V1R12, PFA allows you to use your domain knowledge to reduce false positives.

PFA autonomically collects, models, and compares data by using machine learning techniques to determine what is normal for your system. Supervised learning comes in handy when you are developing applications on your system, stopping and restarting test programs, or running workloads that have behavior such that PFA thinks there is a problem when there really isn’t one. The false positive exceptions caused by this activity can become annoying, but if you think that PFA is “crying wolf” when it really isn’t, you could have a bigger problem on your hands—a system outage!

That’s why, beginning with z/OS V1R12, PFA allows you to be in control by supervising the machine learning of PFA, and it does this by allowing you to specify which address spaces to ignore. The LOGREC arrival rate, frames and slots usage, message arrival rate, and SMF arrival rate checks all allow an EXCLUDED_JOBS file to be created in the configuration directory (config/) of the check. The file must contain the name of the job and the system to which it applies. The file allows wildcards for both the job and the system names so that one line can apply to multiple jobs and systems. You can copy the file from one LPAR to another with little to no additional edits. The MODIFY PFA command is enhanced to display the excluded jobs for each check and allow an updated file to be read and used immediately even when PFA is already running.

#### The enhancements keep coming

z/OS V1R12 contains many additional PFA enhancements. When installing PFA, you now only need one ini file that contains the Java™ configuration information for all checks. When an exception occurs, PFA copies the data you need to investigate the problem into a separate EXC_timestamp subdirectory, and stores the directories for the last 30 exceptions for each check.

Both the common storage and LOGREC arrival checks include performance improvements. Beginning in z/OS V1R12, the common storage check is enhanced to predict and compare four additional locations (SQA, CSA, ESQA, and ECSA) to determine if the storage in any of these locations is in danger of becoming exhausted. The LOGREC arrival check now uses pre-IPL data for modeling, and no longer waits 24 hours after IPL to model.

PFA enhanced its dynamic modeling for all checks to always keep predictions up-to-date. When the system is stable, predictions aren’t made as often, but when it is becoming unstable, modeling occurs more frequently.

Because receiving exceptions (and the possibility of paging activity!) for false positives are undesirable, the comparison algorithms have been further tuned to reduce exceptions that are not cause for alarm. But, what if you are still receiving exceptions that you don’t care about and have already implemented the supervised learning support? Each check has tuning parameters that you can adjust but the defaults should work for most installations. You can also configure these to improve the PFA results on your systems.

#### PFA fixes the future health of your systems

Because we know that your business cannot tolerate system outages, PFA now detects even more types of soft failures using a state-of-the-art layered approach and has been enhanced to give you more controls to reduce false positives. By using PFA, you can increase your system availability by receiving exceptions along with reports that allow you to correct a problem before it affects your business.

### Find out how

For more information, including details on the configuration parameters for each check, see z/OS Problem Management, G325-2564.
Runtime to the rescue!

Using Runtime Diagnostics to find out your problems fast

BY BOB ABRAMS, DONALD DURAND, AND DAVID ZINGARETTI

Runtime Diagnostics is a z/OS started task designed to help you analyze a z/OS system that is “sick, but not dead.” (See “Predict to prevent: Let PFA change your destiny” on page 4.) Runtime Diagnostics decreases the complexity of doing initial problem analysis of z/OS by greatly reducing the time and number of steps required to identify unusual problems or error symptoms, and it goes one better by identifying the next step to take in each case.

Use Runtime Diagnostics when getting ready for a call related to a system problem reported by the help desk or your operations staff. Runtime Diagnostics simplifies the tasks related to identifying suspicious system symptoms. You invoke Runtime Diagnostics using the START command, and it looks through many of the information sources that a skilled system programmer would ordinarily look through, but it does so in under a minute!

Keeping pace with your system

The basic premise behind Runtime Diagnostics is quick system analysis that helps you identify the culprit causing a system to be operating abnormally. Your system might not be performing well, it might appear hung, or maybe it’s not honoring some types of work. Based on this initial analysis, you are able to narrow down the scope of possible symptoms, so that you can then use your systems management tools, such as Resource Measurement Facility (RMF™) and IBM Tivoli® Omegamon, to identify a set of areas to investigate what might be causing these symptoms to occur. Using the output of Runtime Diagnostics as preparation for a call to IBM can help you save time in pinpointing the problem by providing you with some initial leads on possible z/OS component or resource involvement in the suspected problem.

Not a monitor but even better!

Another important consideration is that Runtime Diagnostics is not a monitor. It analyzes the system based on data obtained at the time of the problem and is not a continuous drain on the performance or capacity of your z/OS system. Further, Runtime Diagnostics is carefully designed to do no harm—its role is in the identification of important symptoms that can lead to abnormal z/OS behavior, and the output can suggest the next steps to take in your investigation. But remember that Runtime Diagnostics does not take any action itself.

The overall structure

Runtime Diagnostics follows a basic framework:

- Identifies system component-identified problems based on write-to-operator (WTO) messages captured from the last hour of operation, as contained in the OPERLOG message log.
- Identifies consumption of global resources that might be affecting a wide set of components. Examples of these types of resources include ENQs and locks.
- Identifies symptoms at the address space level, such as CPU usage and whether a function is suspected of looping.

The message analysis that Runtime Diagnostics is based on a set of about 120 z/OS message IDs, reflecting critical component issues that just might be contributing to your system’s abnormal
behavior. The messages are listed in the Runtime Diagnostics chapter of z/OS Problem Management, G325-2564, and include system (error) events in z/OS UNIX® System Services, consoles, dump management, processor spin detection, virtual and auxiliary storage management, XCF, and system logger. In some cases, data is extracted from specific messages and used for further evaluation, like the health of a job identified by a component’s error event. Other situations require the identification of some event, followed by the situation being resolved and therefore are not of real interest to the diagnostician.

Cross-component analysis and CPU usage
Cross-component serialization issues are detected through use of Global Resource Serialization (GRS) information related to long “waiters” and “holders” for ENQ resources. Filtering limits the focus to key system address spaces that are waiters for over five seconds, reporting the waiter and the blocker in each case.

At the address space level, Runtime Diagnostics takes two quick samples on CPU usage and identifies any address space using over 95% of a single CPU. Runtime Diagnostics also identifies address spaces suspended for local locks over 50% of the time. In addition, Runtime Diagnostics finds any task that is involved in a TCB loop through analysis of internal system traces.

Configuring Runtime Diagnostics
Runtime Diagnostics is a new started task beginning in z/OS V1R12. Its module prefix is HZR, so any references to HZR refer to the Runtime Diagnostic procedure in SYS1.PROCLIB.

HZR PROC and OPERLOG
If OPERLOG is running on your system, give the HZR PROC read access to the SYSPLEX.OPERLOG resource through z/OS Security Server Resource Access Control Facility (RACF®), or a similar security product. While it is best to have OPERLOG running for Runtime Diagnostics to perform message analysis, it is not mandatory. Runtime Diagnostics doesn’t read messages from SYSLOG. If you decide not to run with OPERLOG, Runtime Diagnostics simply skips the

message analysis. You can move the HZR procedure to another PROCLIB, but you cannot rename it.

Runtime Diagnostics logging
Runtime Diagnostics can also write its results to a sequential file as a log of all Runtime Diagnostics runs. If you’re interested in such logging, define and pre-allocate a sequential data set with RECFM=FB, and BLKSIZE=0. Specify the data set with DISP=(MOD,KEEP) in the HZROUT DD statement, replacing DD DUMMY. The data set cannot be a PDS, PDSE, or a generation data group (GDG). You also need to define HZR as a subsystem in the IEFSSNxx member of SYS1.PARMLIB.

Running the command to initiate Runtime Diagnostics
Invoke the Runtime Diagnostics function through the START command as follows:

```
START HZR
```

Understanding the output
The START HZR request returns a multi-line WTO message on the z/OS console from which you entered the command. The multi-line WTO is displayed in an out-of-line area if one is set up on the console (for example, K A,20), allowing you to view the output without having it roll off the screen. There are two major portions of the output response: the SUMMARY section and the EVENTS section.

The SUMMARY section displays information about the status of the run including:

- Time of the run
- System name analyzed
- Number and types of system events found.

The example in Figure 1 shows the SUMMARY section indicating that Runtime Diagnostics found two high priority system events: HIGHCPU and LOOP.

The EVENTS section displays information about the system events Runtime Diagnostics detects. Runtime Diagnostics writes one system event record for each type of system event it finds. Currently, Runtime Diagnostics detects a variety of system events associated with component-detected errors. It can also detect:

- ENQ contention
- High CPU usage
- Loop
- Local lock contention.

In Figure 1, Runtime Diagnostics reports HIGHCPU and LOOP events.

The system event record also contains an ACTION statement, which describes the type of action you can take. The system event record can contain up to three ACTION lines within each system event record, documenting recommended actions that you can consider taking.

Figure 1. Example of the SUMMARY and EVENTS section
Keeping your system healthy
Check out Runtime Diagnostics in z/OS V1R12 and find out how quick and easy it is to use. Its primary goal is to allow you, the system programmer, to scope out the problem and recover as quickly as possible. You can view its output on a z/OS console, in the SDSF LOG display, or in a separate file specified in the associated member of SYS1.PROCLIB.

Find out how
The complete documentation for interacting with Runtime Diagnostics is in z/OS V1R12.0 Problem Management, G325-2564 including:

- Enabling Runtime Diagnostics (including defining HZR as a subsystem)
- Understanding Runtime Diagnostics output
- Interpreting Runtime Diagnostics symptoms (including a complete list of all the messages HZR examines)
- Using debug options in case you need assistance from IBM Service.

So there you have it—try it!

IBM Health Checker for z/OS is a tool that helps identify potential configuration problems before they impact availability or cause system outages. There are a bunch of checks provided by IBM, but you can also write your own check routines to verify optimal system configuration settings and issue alert messages for your own configuration or your customer’s.

And before you write your next check, take a look at the great new IBM Redbooks® RedPaper, “Exploiting the IBM Health Checker for z/OS Infrastructure.”

TOPICS INCLUDE:

- How to determine good check candidates unique to an installation. An example might be making sure that data set naming conventions are enforced.
- Security resources available to ensure checks do not violate system integrity.
- Samples demonstrating how to:
  - Write an unauthorized REXX™ check
  - Use message exit automation
  - Write a check routine that saves check data across an IPL
  - Write a check in z/OS Metal C
  - Use a generic message table.

SO TAKE A LOOK AT THE FOLLOWING SOURCES:

- “IBM Health Checker for z/OS Red paper” provides general information:
  

- Go to IBM SYS1.SAMPLIB to get sample health checks built and running as fully functional health checks on your system in 10 minutes!
Are you controlling availability and disaster recovery (D/R) of your IBM z/OS systems and applications through Geographically Dispersed Parallel Sysplex™ (GDPS™)? Do you mirror storage devices over Peer-to-Peer Remote Copy (PPRC) feature connections to have backup solutions available in case of outages, emergencies, or system failures?

If so, then you probably already experienced how time-consuming and susceptible to failure it is, not only to manage a primary site operating system (OS) configuration for every LPAR of your sysplex but also to keep it in sync with adequate definitions for a related D/R site OS configuration.

Previously, when you used Hardware Configuration Definition (HCD) or Hardware Configuration Manager (HCM) to define a D/R OS configuration in the I/O configuration definition file (IODF) in order to represent your sysplex you had to:

• Create a copy of the primary site OS configuration to have a base for the D/R site OS configuration
• Step through the entire device-to-operating system dialog for each DASD range used in PPRC connections, reversing the OFFLINE parameter value setting for each device manually within the copied D/R site OS configuration.

The second task was especially time-consuming and error-prone. You also had to apply all the necessary changes manually to the primary OS configuration and then later to the D/R site OS configuration. It was likely for the primary and secondary site OS configuration specifications to become out of sink.

Help is eZ

Beginning with z/OS V1R12, HCD and HCM introduce the concept of automatically generating D/R site OS configurations at “Build Production IODF” time. This function gives you the opportunity to avoid the disadvantages of earlier configurations. Now, step 1 and step 2 are your responsibility, while step 3 is done automatically by HCD or HCM. Although both HCD and HCM provide this new support in V1R12, HCD does not need HCM, but HCM must be using a z/OS V1R12 HCD level to have the full support.

BY MARTINA WEIDLER AND VIOLA THIERFELDER

Step 1: Configuring a D/R site OS

With the definition of each primary OS configuration of type MVS™, you can provide the unique name of a related disaster recovery site OS configuration that you want HCD or HCM to generate for you automatically. This generated D/R site OS configuration must be a duplicate of the primary OS configuration with the exception that, for duplex storage devices, the OFFLINE parameter setting for each device is reversed.

In Figure 1, a new column in the dialog called “Operating System Configurations” shows whether each OS configuration is generated.
Step 2: Changing the DASD value

A DASD or range of DASD used in a PPRC connection is now optionally classified according to its usage in the PPRC context. The classification as simplex, utility, or FlashCopy® device is optional and for description purposes only. It is, however, important that device ranges reachable from a primary OS configuration that specifies the name of a D/R site OS configuration be defined with PPRC usage type duplex. This specification causes the OFFLINE parameter value to be reversed when the D/R site OS configuration is generated. Figure 2 shows an example of how to define a device’s PPRC usage type.

Step 3: HCD or HCM does the work for you

The “Build Production IODF” and “Build validated work IODF” tasks are enhanced by an automatic step that, when triggered by the definitions in steps 1 and 2, generates the proper D/R site OS configuration. HCD or HCM either builds the OS configuration for the first time; if the OS configuration existed from a previous run, HCD or HCM deletes the OS configuration generated earlier and then re-creates it according to the actual definitions in the primary OS configuration.

Isn’t this the perfect way for you to generate your disaster recovery OS configuration?

Keep in mind:

A D/R site OS configuration loses its status of being generated as soon as you change it. For example, by assigning new devices or modifying its eligible device table. This is to ensure that user modifications do not get overwritten automatically with the next “Build Production IODF” or “Build validated work IODF” actions.

Therefore, after you modify an OS configuration that HCD or HCM creates for you, it is no longer handled by HCD or HCM as a generated OS configuration, which means it is left untouched with the next “Build Production IODF.” You have to decide explicitly whether to keep the changes or whether to delete the D/R site OS configuration and let it recreate according to the actual settings of the primary OS. Isn’t this the perfect way for you to generate your D/R OS configuration: to keep it updated and coordinated with the primary OS configuration?

Find out more

Check out the following publication to find out more:

- Hardware Configuration Definition User’s Guide, SC33-7988
It’s never too early to start thinking about what you’ll need to do to migrate to or through z/OS V1R12. As I’ve done in previous Hot Topic editions, I’d like to provide you with the highlights of some migration actions introduced in z/OS V1R12. Remember: if you are migrating from z/OS V1R9 to z/OS V1R11, you’ll also need to accommodate changes introduced in z/OS V1R10.

Reference
Of course, a complete list of migration actions to consider, along with more details on the ones mentioned below, can be found in z/OS V1R12 Migration, GA22-7499-16.

Action: Review the list of WTORs in parmlib member AUTOR00
z/OS V1R12 marks the start of a new parmlib member, AUTORxx, which is used to provide default write-to-operator-with-reply (WTOR) message specifications. IBM ships a default AUTOR00 in your PARMLIB data set (specified with DDDEF), which we suggest you have in your parmlib concatenation for the z/OS V1R12 IPL. You can concatenate your own AUTORxx members with the AUTOR00 member. We also suggest that you do this rather than editing the shipped AUTOR00 member. By using the shipped AUTOR00, you will receive recommended replies if automation doesn’t respond in a specified period of time or if automation isn’t set up for the WTOR.

What you can do now
Read in z/OS MVS Planning: Operations about AUTORxx and its contents. Compare the contents of AUTOR00 to your existing system automation to see if you would like to create your own AUTORxx parmlib member to be concatenated before AUTOR00. If AUTOR00 is found in your parmlib concatenation during an IPL of z/OS V1R12, it will be used, so it is important that your AUTORxx values reflect what you want.

Action: Accommodate the new REUSASID default in DIAGxx
REUSASID(YES|NO) in your DIAGxx parmlib member controls whether reusable ASIDs will be used when requested on the START command or the ASCRE macro. Before z/OS V1R12, NO was the default, meaning that an ordinary ASID would be assigned if requested. As of z/OS V1R12, YES is the default, meaning that a reusable ASID will be assigned if requested.

What you can do now
You can try out this new default on your current system by specifying REUSASID(YES) in your DIAGxx parmlib member. If the programs that you use cannot tolerate reusable ASIDs, they might issue a system 0D3 abend. If there are no problems, keep REUSASID(YES) on your pre-z/OS V1R12 systems, and either keep it or allow it to default in z/OS V1R12!

Action: Ensure your IP filter rules are compliant with RFC4301
As of z/OS V1R12, all IP security filters are required to be RFC4301-compliant. That is, you can no longer use the RFC4301Compliance parameter on the IpFilterPolicy statement. Any attempt to use that parameter will be ignored, and Policy Agent will enforce that all IP filters are compliant with RFC4301 in z/OS V1R12.

What you can do now
Use migration health check ZOSMIGV1R11_CS_RFC4301 (introduced in APARs OA28605 and PK84362) to determine if your IPSec filter rules are RFC4301-compliant. Change any non-compliant IPSec filters before migrating to z/OS V1R12.

Action: Accommodate Telnet server stopping when OMVS is shutdown
As of z/OS V1R12, Telnet will stop when you issue the F OMVS,SHUTDOWN command. Before z/OS V1R12, Telnet continued to be active but it would be in an unusable state.

What you can do now
When issuing F OMVS,SHUTDOWN, ensure that your procedures take into account that the Telnet server stops when the command is issued.

Action: Allow users to access SMP/E commands and programs
As of SMP/E APAR IO11698, users need proper authority to access SMP/E functions. This APAR applies to SMP/E V3R4 and V3R5. You might need to do this migration action before you migrate to z/OS V1R12 if you install this APAR with the current SMP/E level you are now using.
What you can do now

Define the FACILITY class profiles and permit users to access them for continued ability to execute SMP/E commands and programs. You can protect the SMP/E commands and programs generically (such as with GIM.*) or discretely (such as with GIM.PGM,GIMUNZIP or GIM.CMD.APPLY). READ access to these resources is required. Follow the instructions provided in the HOLD ACTION when installing the PTF for APAR IO11698.

Action: Accommodate the shell and utilities version of the tsocmd command

You might be using the nifty tsocmd that lets you run a TSO/E command from the shell. It is available from z/OS UNIX Tools and Tools Web site:

ibm.com/systems/z/os/zos/features/unix/bpax1toy.htm

As of z/OS V1R12, tsocmd has been incorporated and now has some differences from the earlier version:

• The environment variables tsain and tsaout are no longer supported.
• The environment variables stdin and stdout are supported (which is typical for other shell and utilities commands).
• The environment TSOPROFILE variable is supported.
• If the TSO/E command fails, exit values are consistently set.

What you can do now

If you want to continue to use your old tsocmd command, move it to another location than /bin (which is where the z/OS V1R12 tsocmd will be installed). If you want to use the z/OS V1R12 version of tsocmd, be aware of the differences you'll see, and remove the Tools and Toys version from the authorized load library where it was stored. Also, if you want to use the z/OS V1R12 version of tsocmd on your current system, it is now available from the Tools and Toys Web site. This lets you see the changes ahead of time; also, it is a good idea to use a consistent version of tsocmd across your z/OS releases!

Action: Modify the ICSF startup procedure program name from CSFMAIN to CSFINIT

The name of the ICSF initialization module is changing in FMID HCR7770 (which is in Web deliverable "Cryptographic Support for z/OS V1R9-R11" and integrated into z/OS V1R12). This name change was caused by a need for ICSF to be non-cancelable and non-swappable.

What you can do now

Decide how you will start ICSF. The options include having a new startup procedure or using the same startup procedure but with the program name as a system symbol that is uniquely resolved for different levels of ICSF. If you don't start ICSF with the correct new program name, ICSF will not initialize.

Action: Get ready for the Printer Inventory format changes

z/OS V1R12 requires a new format of the Printer Inventory for Infoprint Server. This new format is called v2, and the previous format is now called v1. Automatically, upon initialization of Infoprint Server on z/OS V1R12, if a V2 Printer Inventory is not found, a V2 Printer Inventory will be created from your V1 Printer Inventory. If the system finds a V2 Printer Inventory, It does not create a new one, and the system uses the V2 Printer Inventory that it finds. Upon fallback to z/OS V1R10 or z/OS V1R11, you must use a V1 Printer Inventory, as only z/OS V1R12 can understand a V2 Printer Inventory.

What you can do now

To accommodate this V2 requirement there are several items you can plan for now:

• Ensure that you have enough space in your Printer Inventory file system for a V2 Printer Inventory and your existing V1 Printer Inventory. Assume that a V2 Printer Inventory will be double the size of your V1 Printer Inventory. By having both the V2 and the V1 Printer Inventories available, you will be best positioned for fallback. Typically, your Printer Inventory is in the file system you have mounted at /var/Printsrv, but you might have specified a different location (base-directory) in your /etc/Printsrv/aopd.conf file. Migration health check ZOSMIGV1R12_INFOPRINT_INVSIZE added with APAR OA32093 can help with this issue.
• If you are falling back from z/OS V1R12 to z/OS V1R10 or z/OS V1R11 and will be staying on that lower release level for a while, consider removing your V2 Printer Inventory because any updates done to your V1 Printer Inventory will not be reflected in your V2 Printer Inventory. Health check INFOPRINT_V2DB_CHECK added with APAR OA32093 can help with this issue.
• Similarly, when you are confident that you will remain on z/OS V1R12, consider removing your V1 Printer Inventory file.

See “Infoprint Server: Still serving your printing needs” on page 29.

Action: Take advantage of SMP/E FIXCATs to do PTF verification

There are some important SMP/E FIXCATs associated with migrating to z/OS V1R12:

• IBM.Coexistence.z/OS.V1R12: Use this FIXCAT on the REPORT MISSINGFIX command to look at your z/OS V1R10 or z/OS V1R11 zones to see which coexistence PTFs are missing for z/OS V1R12 on these lower-level z/OS systems. If you are migrating from z/OS V1R10 or z/OS V1R11 you only need to specify this one FIXCAT to find all the required coexistence PTFs.
• IBM.TargetSystem-RequiredService.z/OS.V1R12: Use this FIXCAT on the REPORT MISSINGFIX command to look at zones where you have installed other products that run on z/OS, to see which target system PTFs for those other products that you need to run on z/OS V1R12.
• IBM.Function.HealthChecker: Use this FIXCAT on the REPORT MISSINGFIX command to look at your z/OS V1R10 and z/OS V1R11 zones to see if you have any missing IBM Health Checker for z/OS health checks for migration.

What you can do now

Periodically run the SMP/E REPORT MISSINGFIX command against the appropriate target zones, specifying the above FIXCATs. Install any PTFs that the reports identify as missing.
Professor Kimura explains z/OS migration to you!

BY SHIGEKI KIMURA

Professor Migration has expertise in and deep knowledge of z/OS release-to-release migration. Customers who need help in z/OS migration ask him for advice, and he writes a prescription to resolve the pain. This article contains a series of his practical advice in the following areas.

- Is there any migration assistance to help identify the impact of the change?
- Is there any enhancement or solution in a new release to help with the customer’s request?
- Is there any compatibility option in a new release to handle the old behavior?

Migration assistance for the change to DISP=UNCATLG processing

From the customer:
Dear Professor Migration:
After the transition to z/OS V1R10, we experienced an error situation in DISP=UNCATLG processing as follows. How can we easily identify the affected JCL statements to be updated?

```plaintext
//UNCAT DD DSN=BEANS.ZOS.HOT.TOPICS.SAMPLE1,
//             UNIT=3390,DISP=(OLD,UNCATLG),VOL=SER=WRKI02
//DEL DD DSN=BEANS.ZOS.HOT.TOPICS.SAMPLE1,
//             UNIT=3390,DISP=(OLD,DELETE),VOL=SER=WRKI02
```

IEF142I BEANSZZ STEP1 - STEP WAS EXECUTED - COND CODE 0000
IEF287I BEANS.ZOS.HOT.TOPICS.SAMPLE1 NOT UNCTLDG 13
IEF287I VOL SER NOS= WRKI02.
IEF285I BEANS.ZOS.HOT.TOPICS.SAMPLE1 DELETED
IEF285I VOL SER NOS= WRKI02.

Professor’s advice:
Thank you very much for your inquiry. Your observation is correct because the behavior in z/OS V1R10 now only allows a data set to be uncataloged when the data set information is retrieved from the catalog at allocation time to prevent accidental data loss.

I suggest that you make use of the new function in APAR OA27917. This APAR allows you to avoid a migration action if the changes required for it are too disruptive (for example, if you need to change a lot of production jobs).

It introduced a new option for the ALLOCxx parmlib member to provide migration assistance for existing JCL that might be incorrectly coded and it takes advantage of the pre-V1R10 behavior. You can request the pre-V1R10 behavior by tracking the job and program information with the Tracking Facility (using the SYSTEM VERIFY_UNCAT(TRACK) option) and optionally through message IEF384I in the job log; the message indicates that the incorrect data set might have been uncataloged (using the SYSTEM VERIFY_UNCAT(MSGTRACK) option).

This is something that is valuable even after you have migrated to z/OS V1R10, because you can still use the tracker to indicate what is out there that you might want to identify or update. This item should also be helpful for other customers who migrate to z/OS V1R11 from z/OS V1R9.

You can find more information about the Tracking Facility at the following Web site:

```
publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/IEA2G3A0/A.0?SHELF=EZ2ZBK0H&DT=2009063221244
```
Solution for the duplicate temporary data set name

From the customer:

Dear Professor Migration:

When a temporary data set is specified as DSN=&&tempname, the system-generated qualified name for the temporary data set is not unique under two conditions. The first is if the system created data sets for multiple tasks or APPC transactions having the same job names within the same system clock second. The second is if the tasks or transactions contain DD statements that specify the same temporary data set names. To help avoid the problem and suppress the JCL error, we need to remove the DSN=&&tempname specification and update the rest of the JCL to use refer-back notation for the data set. Do you have any solution to avoid this time-consuming work?

Professor’s advice:

Yes, a new function in z/OS V1R12 will help you. Beginning with z/OS V1R12, you can use a new ALLOCxx parmlib option, SYSTEM TEMPDSFORMAT(UNIQUE), to specify that the system uses the data set naming convention for the unnamed temporary data set instead. This substantially reduces the probability of this JCL error without the need to change JCL. This solution involves generating a unique name for a temporary data set even when the DSN=&&tempname is specified; consequently, the resulting data set name looks like the one generated when DSN=&&tempname is not specified.

Solution for the overflow of SMF30SRV field

From the customer:

Dear Professor Migration:

There are 4-byte fields (SMF30xxx: xxx=SRV,CSU,SRB,IO,MSO,ESU) in the Performance Section of the SMF type 30 record. We always need to get the correct values because the service units (SMF30SRV) are used as a base for customer billings. However, this field grows to ‘FFFFFFFF’ and then wraps back to zero and continues growing, so it results in erroneous resource consumption of data, especially for the long running jobs. Our shop is now running z/OS V1R10; How can we resolve this issue?

Professor’s advice:

You are so lucky because this issue can be resolved by new expanded sized service unit fields in the SMF-type 30 records. When you have migrated to z/OS V1R11, or if you have applied PTF UA42968 for APAR OA25540 (SRM) and PTF UA47865 for APAR OA26832 (SMF) in z/OS V1R10, you can use the following new functions.

- New 8-byte fields (SMF30xxx_L: xxx=SRV,CSU,SRB,IO,MSO,ESU) are added to the Performance Section of the SMF-type 30 record. These fields are equivalent to the corresponding 4-byte fields and continue to be valid after the 4-byte fields wrap.
- A new flag byte SMF30INV is also added to the Performance Section. It provides individual bits that are set when each 4-byte field exceeds its maximum capacity. For example, when wrapping occurs for SMF30SRV, SMF30SRV_INV will be set to on.

Enhancement of writing an EOF mark for non-SMS managed sequential data set

From the customer:

A guy who is new to z/OS in our shop asked me why we run the in-house procedure to open output and close the newly allocated non-SMS managed sequential data set. I know the simple answer for him, but it makes me ask you another question. How can we resolve this issue without our own effort?
Professor’s advice:

Now, I have good news for you. Beginning with z/OS V1R11, when the system allocates a new data set with DSORG=PS or no DSORG, regardless of whether it is SMS-managed or not, it writes a file mark at the beginning of any data set for which space is allocated. With this enhancement, a program can safely read the data set before data has been written in it.

Here is a simple scenario to demonstrate this new feature. Note that ISPF OPT3.2 Allocation writes an end-of-file mark for any new non-SMS managed sequential data set even before z/OS V1R11.

CURRENT OF new STORAGENSWDP

specifies that the system should also select non-swappable address spaces to resolve the storage shortage except for the address spaces in service class SYSTEM. To avoid your problem situation and expect the same behavior as previous releases, setting the STORAGENSWDP=NO parameter can be an answer for you. Good luck!

Consideration of HIBFREXT and LOBFREXT options in TSOKEYxx parmlib member

From the customer:

Dear Professor Migration:

After migrating to z/OS V1R10, some TSO/E users reported that the Forward (PF8) operation in ISPF/PDF (OPT1 VIEW or OPT2 EDIT) hung. When they pressed the ESC key, the next ISPF screen appeared. Is this a known problem?

Professor’s advice:

I would suggest that you check the HIBFREXT and LOBFREXT values in your TSOKEYxx parmlib member. I do not recommend using any lower values than the defaults, HIBFREXT=48000 and LOBFREXT=24000, and this is very important when you migrate to z/OS V1R10. There have been quite a few instances of customers hanging at the z/OS V1R10 level because of low settings for these parameters. To avoid this situation, both HIBFREXT and LOBFREXT should be set to the default values or higher.

While you might have used the TSOKEY00 parmlib member shipped by z/OS V1R10 ServerPac that contained HIBFREXT=6600 and LOBFREXT=3300, the good news is that z/OS V1R11 ServerPac has already been enhanced to specify the default values. This should help.

Moving along

I hope these questions and answers can help you when you start to migrate to z/OS V1R12. This is Professor Kimura signing off for now!
The z/OS Management Facility (z/OSMF) is a modern Web-based management console for z/OS. First introduced in z/OS V1R11, z/OSMF has been enhanced for z/OSMF V1 12. But we’ll discuss those enhancements in other articles in this issue of the newsletter. For now, we want to brag about how z/OSMF for z/OS V1R12 is easier than ever to use.

**Configuration: easier and more flexible**

First, we simplified the configuration process by using updated installation scripts requiring fewer steps to complete. And we provide more flexibility. You can now choose which z/OSMF tasks to configure immediately and add others later, at your convenience. That way, you don’t have to configure it all at once. We also enhanced the z/OSMF Incident Log task that uses z/OS components and features adhering to best practices for data management for problem determination. Starting with z/OSMF V1 12, the z/OSMF tasks are delivered across multiple FMIDs, which simplifies serviceability by allowing you to update a single FMID!

**A more welcoming welcome**

After z/OSMF V1 12 is installed and running, the first thing you’ll notice is the Welcome page. You can customize this page with your own company-approved text and logo.

**Better links**

As in z/OSMF V1 11, you can still add links to other sites for information and systems management tools. However, in z/OSMF V1 12 you can also add performance-related links to the Performance category, problem-determination-related links to the Problem Determination category, or any links to any category that makes sense to your users. This makes it easy for you to organize your links and for your users to locate the right link at the right time. While we are talking about links, let me add that it is now easy to distinguish z/OSMF tasks from links to non-z/OSMF resources because we use different icons for them.

**Attach, encrypt, and personalize: easy problem determination**

Now take a look at the Problem Determination category and select the Incident Log task. In response to customer requests, we’ve made several enhancements to make Incident Log tasks even easier to use:

- You can attach additional documents to an incident and send those documents either alone or with other data associated with that incident. This allows you to use the simplified send wizard of the incident log to send the additional diagnostic data using FTP to IBM or anywhere else you need to send it.
- z/OSMF can encrypt diagnostic data that you send to IBM. With a one-time configuration of an IBM FTP destination, you simply use that destination when sending data to IBM, and it will be encrypted for you using your user-supplied cipher.
- You can add a personalized note to each FTP destination so you can easily identify which destination to select when sending diagnostic data.
- You can add personalized notes to each incident in the Incident Log itself. By adding your personal notes to an incident, you can easily identify selected incidents that have been assigned to a particular person or indicate where you want to send the data, or indicate whatever else you want to use the data for.

**New performance management tasks!**

In the Performance category, the new z/OSMF Workload Management task provides a simpler way to create and edit service definitions using best practices and integrated IBM Health Checker for z/OS checks to avoid common errors. This new task also simplifies the maintenance of service definitions for experienced z/OS users by providing functions to analyze service definitions and integrated information on how service definitions can be optimized.

The Monitoring Desktops and Sysplex Status tasks, also under the new Performance category, provide integrated monitoring from a single point of control for the z/OS sysplexes or Linux® images (System z™ and Intel®) in your environment. The Sysplex Status task gives a quick assessment of sysplex health. With the predefined monitoring desktops task, you can drill down into resource attributes and metrics for a quick start. At the same time, the task provides flexibility through customizable user-defined desktops.

**For more information**

See the following articles in this issue:

- “A new look for z/OSMF” on page 17
- “Set the cruise control to z/OSMF: Help manage your workloads and performance with enhancements to z/OSMF V1 12” on page 18
- “Incidentally, z/OSMF is moving forward: Enhancements to the z/OSMF Incident Log” on page 21.
With z/OS V1R11, IBM announced IBM z/OS Management Facility (z/OSMF), a modern, easy-to-use GUI interface for managing your z/OS system. In z/OSMF V1 11, z/OSMF started with problem data management and TCP/IP policy-based configuration tasks to help you manage your systems. In z/OS V1R12, z/OSMF includes enhancements for existing tasks as well as additional new functions and tasks.

What’s new and changed with the z/OSMF Welcome Page

The z/OSMF Welcome page has been updated as follows:

Configuration category
Enhancements to and support for the Configuration Assistant from the Communication Server application include:

- Support for configuring IKEv2
- New cryptographic algorithms for IPSec and Internet Key Exchange (IKE), and configuration of Federal Information Processing Standards (FIPS) 140 cryptographic mode for IKE
- Configuration of certificate trust chains and certificate revocation lists
- Enforcement of RFC4301 compliance for IP security filters
- Usability enhancements.

Links category
A new script interface now allows users to programmatically add a link to z/OSMF. In z/OSMF V1 11, adding a link places the items as a task only in the Links category. With this support, you can add a link to any existing category.

Performance category
A new category in z/OSMF V1 12 introduces three tasks for the z/OS Workload Manager (WLM). You can use the WLM policy editor to perform the following functions for the Workload Management task:

- Creating and editing of WLM service definitions
- Installing WLM service definitions
- Activating WLM service policies.

In addition, z/OSMF is providing the Resource Monitoring task with the Sysplex Status and Monitoring Desktops tasks. These tasks provide integrated performance monitoring of z/OS sysplexes in your environment.

Problem Determination category
The Incident Log task has been enhanced with new functions, many of which are in direct response to customer requirements:

- Encrypting incident files, including memory dumps
- Sending incident files to IBM through FTP
- Sending additional documentation with an incident to an FTP destination
- Providing free-form notes for each incident
- Including the new EMAIL variable in incident log FTP profiles.

In addition, with the Incident Log task, you can create diagnostic log snapshots that are based on the SYSLOG and LOGREC data sets, as well as the OPERLOG and LOGREC sysplex log streams.

Read more!
Want to find out more? See the following articles in this issue:

- “Easy as z/OSMF” on page 16
- “Set the cruise control to z/OSMF: Help manage your workloads and performance with enhancements to z/OSMF V1 12” on page 18
- “Incidentally,” z/OSMF is moving forward: Enhancements to the z/OSMF Incident Log” on page 21.
Set the cruise control to z/OSMF

Help manage your workloads and performance with enhancements to z/OSMF V1 12

BY MARIANNA WENSKE, VLADISLAV GORDON, AND HARALD BENDER

You’ve no doubt been hearing a lot about the IBM z/OS Management Facility (z/OSMF). z/OSMF is designed to help you manage various aspects of a z/OS environment through a Web browser interface. In V1R12, two new tasks are introduced in the z/OSMF Performance category: the Workload Management task and the Resource Monitoring task to help get you and your workloads to where they need to go fast!

Part 1: Easing into things, the workload management task

The z/OSMF Workload Management task simplifies the work of the z/OS Workload Manager (WLM) administrator and addresses pain points identified in customer studies. It provides the functions required for administering and operating z/OS WLM as well as features simplifying the handling and maintenance of WLM service definitions and policies.

The z/OSMF Workload Management task offers different GUI views (table-based or HTML-based) to quickly analyze, review, or edit service definitions. You’re able to customize the views by sorting and filtering functions to get the optimal view for a particular edit or analysis task. Let’s compare table-based and HTML-based views.

Table-based viewing

In the table-based view, you can directly navigate between the different parts of the service definition in one click. Therefore, the editor minimizes the time required for investigating and analyzing a service definition. The service definition parts consist of logically related elements in a service definition, and each of them is presented in a table with relevant information that might interest you when editing it. The tables can be directly edited through intuitive and easy-to-use navigation operations. Reasonable and self-explanatory table actions help you edit and manipulate the elements in a table. Supplementary features like sorting and filtering simplify the analysis tasks of a WLM administrator. With the WLM editor, you can search through the displayed table of elements so that you can view all matches at one time.

Another outstanding improvement when working with a WLM service definition is checking feature that avoids common specification errors. You can edit a service definition in real time because the checking mechanism provides guidance on how to improve a service definition. Recommendations and errors are indicated visually through corresponding icons. To get an overview you can use the messages table showing all recommendations for a service definition.

Serving up service definitions

In Figure 1, we can see the service classes of a specific service definition. A service class can have up to eight periods. The service-class periods relationship is presented as a parent-child relationship. Dependent on the element type of a row, context sensitive menus are available. Recommendations, like warnings, are displayed in the affected cell and the parent service class row, so that you can see the recommendations even when the periods are collapsed. To reach another part of the service definition, you click the Switch-to button in the right corner.

Figure 1. HCM Operating System Configuration dialog
You can also open multiple service definitions simultaneously to compare them. A comprehensive cut, copy, and paste function that is designed to let you compose a new service definition by copying pieces of existing service definitions provides for a simplified migration or consolidation of service definitions.

**HTML-based viewing**

The HTML-based view is a read-only view and shows all parts of a service definition at once or only a subset of them; it’s up to you. You can arrange your own combination by setting a display filter. The display filter enables you to select the parts of a service definition that are of interest, including the messages table with all best-practice recommendations. Besides providing a good outline of the whole service definition, the HTML-based view is well suited for printing service definitions or service definition parts.

**Storing service definitions**

In addition to the policy editing and review functions, the z/OSMF workload management task integrates a repository to store the service definitions of an installation. You can import and export service definitions into and from the repository to host data sets or workstation files in XML or CVS format. Thus, you are able to print service definitions as well as the best-practice recommendations for a service definition for further study in a relatively easy manner.

**Synchronizing changes**

The z/OSMF Workload Management task automatically synchronizes with z/OS WLM. The installed service definition can be directly accessed and edited. When you edit a service definition, the change is serialized to ensure that policy changes are not overwritten accidentally. A wizard helps you install the service definitions from the repository and activate service policies, and you can monitor the WLM status of a sysplex.

**Logging and administrative tasks made easy**

z/OSMF also allows operations on service definitions (modifications, installations, activations, and so forth) to be automatically logged and displayed in a history report so that you don’t have to rely on manual tracking.

The z/OSMF Workload Management task substitutes its own functions for those of the ISPF WLM administrative task, the WLM service definition editor tool, and the WLM service definition formatter tool in a typical environment; thus, z/OSMF helps improve the efficiency and ease of use for WLM administrators who need to perform all of these tasks.

**Part 2: Easing into things, the Resource Monitoring task**

Now that you’ve created and installed the WLM service definition, it’s time to verify the active policy with the new z/OSMF Resource Monitoring task. Resource monitoring provides two tasks that enable integrated performance monitoring from a single point of control: the Sysplex Status task and the Monitoring Desktop task.

The Resource Monitoring task gets its data from the IBM z/OS Resource Measurement Facility (RMF) Distributed Data Server (DDS). Ensure that you have at least one z/OS or Linux DDS running in your installation.

**Task 1: Get on board with the Sysplex Status task!**

The starting point for the monitoring is the Sysplex Status task. This task provides a comfortable way to assess the health status of all systems in your installation at a glance. The table contains the list of z/OS sysplexes and Linux images. You can add, modify, and delete items in the table using the Actions menu. When you start the task for the first time, one default entry is provided as LOCALPLEX that points to the DDS in the sysplex in which z/OSMF is running. If you have a running DDS in this sysplex, the Sysplex Status task detects it automatically without requiring an explicit declaration of its host name or IP address. In addition, the Sysplex Status task can automatically detect the DDS movement within the sysplex during the z/OSMF session.

In addition to the LOCALPLEX you can add all target sysplexes you want to monitor to the table. Now let’s explore the column output as shown in Figure 2.

![Figure 2. Sysplex Status task example](image-url)
The PI-Status column has a red-yellow-green indicator for the sysplex health. This indicator is based on the WLM service-class period goals and actuals. If all service class periods on the system are meeting the goals (that is, they have a performance index (PI) of less than or equal to 1), the PI Status is green. If service classes with importance of 3, 4, and 5 with the PI greater than 1 are detected, the indicator is yellow. If at least one important service class (that is, importance of 1 or 2) misses the WLM goal, the indicator is red.

In this case, it becomes essential to figure out the reasons by drilling down into the details on the respective sysplex. You can analyze the data within the Monitoring Desktops task.

Task 2: Discover it with the Monitoring Desktop task!

A monitoring desktop is a customizable view containing different performance metrics that you can group and arrange flexibly. You can create and save your own desktops or open and modify the predefined desktops that are included with the task in z/OSMF.

When you start a monitoring desktop, it begins retrieving the online data from the DDS periodically. By default, the most current snapshot is displayed in a new tab as shown in Figure 3.

Filtering capabilities

Resource monitoring provides sophisticated filtering capabilities. Suppose you select the metric “% CSA utilization by job,” which lists all the jobs active on the system and their common storage area (CSA) consumption in percent. Because the number of jobs in the system can be high, you can specify filters and work scopes: that is, display only jobs that match a certain name pattern, belong to a specific WLM service class, or meet a threshold value.

z/OSMF: Ease of use for managing workloads and monitoring performance

The main objectives of the new z/OSMF Workload Performance and Resource Monitoring tasks are simplification and ease of use. Comprehensive online help is provided for each panel. Tool tips, descriptive texts as well as error, warning, and information indications guide you through the panels. Try it and stay tuned! There’s more to follow in the future!

A good place to start with understanding z/OSMF and its functions and tasks is to review the IBM z/OS Management Facility V1R12 Configuration Guide, SA38-0652.

Special thanks

This article was written in part by Stefan Wirag whose expertise on designing the interface contributed greatly to this article. Thanks also to Cheryl D. Loughlin and Toshiba Burns-Johnson for the great GUI design contributions.
“Incidentally,” z/OSMF is moving forward

Enhancements to the z/OSMF Incident Log

BY SUSAN Z. DEMKOWICZ AND WILLIAM A. RUBY

The new release of z/OSMF Incident Log has more functions to assist you in your problem data management tasks. Enhancements in the z/OSMF Incident Log user interface and its corresponding z/OS component, the common event adapter (CEA), provide additional configuration flexibility, in support of diagnostic data logging and recording, for the Incident Log. This addresses a number of customer and service requirements, by making the Incident Log viable in basic sysplex environments (with no coupling facility).

**Bringing diagnostic snapshots into sharper focus**

In z/OS V1R11, CEA supports the creation of “diagnostic snapshots” only when you use the OPERLOG and LOGREC log streams. In z/OS V1R12, CEA supports diagnostic data collection using SYSLOG and LOGREC data sets when OPERLOG and LOGREC log streams are not in use. The same CEA configuration policy continues to work, with one restriction: the LOGREC summary is forced to a maximum of 4 hours instead of the previous default of 24 hours. But, don’t worry! If you have Incident Log already configured, it will continue to work.

**Changes to the Incident Log summary panel and LOGREC summary report**

The z/OSMF Incident Log summary panel now supports a Notes column, and the data is persisted by CEA in the sysplex dump directory, along with other data related to each incident. The Notes value is a free form area (up to 255 bytes) that can be set and modified. This popular request allows you to record a variety of information for each incident.

Based on suggestions from IBM Level 2 support, improvements were made in some of the data that CEA can capture. The LOGREC summary report is now expressed in tabular form. Also, the reports presented for abend-related incidents in I/O related components contain additional hardware-type LOGREC information.

**Recording problem incident information in a mixed-release sysplex**

The Incident Log function runs on one system in the sysplex and reflects incidents about SVC dumps that originate on all systems in the sysplex, provided that the sysplex dump directory (SYS1.DDIR) is set up on a shared DASD volume. With z/OSMF V1 12 and the CEA enhancements in z/OS V1R12, information related to problem incidents continue to be recorded, even in a mixed-release sysplex; these new features will be available only on the new z/OS V1R12 release.

For example, the SYSLOG diagnostic snapshot will only be captured if the system on which incident occurs is at the V1R12 level. The snapshot will be listed on any release of the z/OSMF panels, as long as the CEA is at the V1R12 level. Similarly, the new Notes value is supported only on z/OSMF V1 12.
In summary, the rules are simple:

- z/OSMF V1 12 can only run on z/OS V1R12
- z/OSMF V1 11 can run on z/OS V1R10, V1R11 or V1R12.

Figure 1 shows CEA Incident Log data in a mixed-release sysplex:

When you delete an incident using the z/OSMF Incident Log, CEA deletes the dump and all diagnostic snapshots that were created by CEA for that incident on any z/OS release level. In z/OS V1R12, the delete function has been adjusted to accommodate a new diagnostic data set naming convention. To ensure that the older releases can completely perform this function, you must ensure that toleration APAR OA32285 is installed on z/OS V1R10 and V1R11 systems.

z/OSMF Incident Log just keeps getting better and better

So if you haven’t done so yet, give the z/OSMF Incident Log a try! You can now benefit from this z/OS simplification feature by using SYSLOG and LOGREC data sets, OPERLOG, and LOGREC log streams, in a sysplex or not, in any combination!

To learn more about z/OSMF and Common Event Adapter, see the articles “An Introduction to z/OSMF” and “New kid on the block: Common Event Adapter” in z/OS Hot Topics Newsletter Issue 21, August 2009, GA22-7501-17.

Acknowledgments

We’d like to thank Bob Abrams and Anuja Deedwaniya for their contributions to this article.
It’s the zFavorites for System z credit card CD! You’re gonna love this! It has all sorts of helpful Web links, like those for:

- Hardcopy
- Operating systems
- Software
- Language and tools
- ISV development and applications
- Product documentation
- Marketing information
- Education
- Support
- Links to FREE downloads
- IBM Redbooks sampler
- WebSphere® Application Server
- XML.

To use the CD insert it in any standard CD device, and it should start automatically. If it does not, click Start > Run, and then type x:\index.htm (where x is your CD drive letter) and press Enter.

Additional copies of zFavorites CD (GK3T-4331-17) are separately orderable.

zFavorites:
www.ibm.com/servers/eserver/zseries/zos/zfavorites/
Here are more Q and As on catalog-related issues from Mr. Catalog.

(Q) I’ve heard that with z/OS V1R12, we will be able to catalog a zSeries™ File System (zFS) version root using an indirect volume serial number. I assume this would work for any zFS data set on the SYSRES pack?

(A) Yes, the support for cataloging a zFS version root with an indirect volume-serial number is in z/OS V1R12.

I’ll let my learned colleague Rohan Kurane, who developed the code, explain the procedure for using this new feature:

Rohan Kurane: Thanks, Mr. Catalog! Cloning non-VSAM data sets like a hierarchical file system (HFS) uses the extended, indirect volume serial (VOLSER) support, which uses system symbols for the VOLSER. In a shared catalog environment, the data set name entry in a catalog is only defined once with one indirect VOLSER.

But zFSs are VSAM linear data sets (LDSs) that, up until now, could not have an indirect VOLSER in its catalog entry.

Well, here’s some news that will put a smile on your face! Starting with z/OS V1R12, you can make a clone, or copy, of a zFS and use indirect VOLSER numbers to access it. You can define a catalog entry for a zFS to specify different VOLSERs for different systems, and use existing processes for cloning systems.

This support is limited to single-volume zFS data sets.

To take advantage of this new function, just do the following steps:

1. Define a system symbol in a SYS1.PARMLIB member (IEASYMxx).
   For example:
   ```
   SYSDEF SYMDEF(&VOL01='1P0301').
   ```
   The period at the end of the symbol is optional.

2. Clone a zFS by making copies of the existing zFS data sets using the COPY command with the PHYSINDYNAM (PIDY) parameter.

   Using PHYSINDYNAM is important because it does not create a catalog entry, and you can use the same name for your original and copied zFS. PHYSINDYNAM does create a VSAM volume data set (VVDS) entry for the zFS.

   The following example JCL shows a COPY command to clone a zFS:
   ```
   DEFINE CLUSTER -
   (NAME(ZFS.LDS) -
   LINEAR -
   VOLUMES(&VOL01) -
   RECATALOG)
   ```

3. Issue the DEFINE RECATALOG command using the system symbol (indirect VOLSER) that you defined in step 1. The data set has a device type of x’00000000’.

When you issue the LISTCAT command later, the data set has the following attributes for the VOLUMES section of the data component:

   ```
   CLUSTER ------- ZFS.LDS
   DATA ------- ZFS.LDS.DATA
   VOLUME
   VOLSER----------&VOL01
   DEVTYPE-------x'00000000'
   ```

Using this process, a single catalog entry (for zFS VSAM linear data sets) can represent different volumes on various systems, or different volumes at different times on the same system.
Figure 1 outlines the steps:

**Figure 1. Cloning and cataloging a zFS data set with an indirect VOLSER**

**Mr Catalog:** Thanks Rohan, now on to the other questions...

(Q) Why can’t I partially release space when one of the volumes in a multivolume VSAM data set is empty?

(A) This problem came about because of the Storage Management Subsystem (SMS) space-constraint relief processing. When DFSMS™ first introduced partial release processing, the only way a VSAM data set could be multivolume was through extend processing. Because extend processing is triggered by the addition of a new record, data was always present on the newly acquired volume. In other words, an empty primary volume simply could not occur. Therefore, partial release processing only released space on the last volume of the data set.

When DFSMS introduced space-constraint relief processing (also known as the best fit method), the data set could be spread over multiple volumes on initial allocation. As a result, this introduced the partial release problem because primary volumes could be empty. A function to remove empty volumes did not exist.

The good news is that this problem is solved in z/OS V1R12. When a partial release is invoked, empty volumes are removed and assigned candidate status. That leaves only primary volumes with used space. Unused space from the last primary volume is then partially released from the high-used relative byte address to the high-allocated byte address as it was in the past.

(Q) Is there a way to allocate all of the GDSs within a GDG?

(A) Yes, by specifying the generation data group (GDG) name on the JCL DD statement, all generation data sets (GDSs) within the GDG are allocated and are processed as if they were on a concatenated DD statement.

For example, consider B.GDG, which contains three GDSs: B.GDG.G0002V00, B.GDG.G0003V00, and B.GDG.G0005V00. And let’s say that the GDG limit is 3. The GDG relative generation 0 is B.GDG.G0005V00. Adding B.GDG.G0001V00 rolls off B.GDG.G0002V00 because the GDG limit is met, and the result is B.GDG.G0001V00, B.GDG.G0003V00, and B.GDG.G0005V00. (Roll off specifies that the oldest copy of the data set

Instead of:

```plaintext
//DD1 DD DSN=A.GDG.G0001V00,DISP=SHR
//   DD DSN=A.GDG.G0002V00,DISP=SHR
//   DD DSN=A.GDG.G0003V00,DISP=SHR
```

you can simply specify:

```plaintext
//DD1 DD DSN=A.GDG,DISP=SHR
```

and get the same result from your application program.

(Q) What does Catalog do when you specify an absolute GDS name with DISP=(NEW,CATLG)?

(A) The GDS is added in the GDG where it belongs, as follows:

- If it is less than the least generation number, the GDS is added as the oldest.
- If it is higher than the greatest, the GDS is added as the newest.
- If it is in between and not a duplicate, the GDS is added in sequence.

For example, consider B.GDG, which contains three GDSs: B.GDG.G0001V00, B.GDG.G0002V00, and B.GDG.G0003V00, and B.GDG.G0005V00. And let’s say that the GDG limit is 3. The GDG relative generation 0 is B.GDG.G0005V00. Adding B.GDG.G0001V00 rolls off B.GDG.G0002V00 because the GDG limit is met, and the result is B.GDG.G0001V00, B.GDG.G0003V00, and B.GDG.G0005V00. (Roll off specifies that the oldest copy of the data set
is a candidate for deletion in order to maintain the customer-specified limit.) The GDG relative generation 0 is still B.GDG.G0005V00.

Adding B.GDG.G0021V00 to that result rolls off B.GDG.G0001V00 because the GDG limit is met, and the result is B.GDG.G0003V00, B.GDG.G0005V00, and B.GDG.G0021V00. The GDG relative generation 0 is still B.GDG.G0005V00.

Adding B.GDG.G0019V00 to that results in B.GDG.G0003V00 rolling off because the GDG limit is met, and the result is B.GDG.G0005V00, B.GDG.G0019V00, and B.GDG.G0021V00. The GDG relative generation 0 is still B.GDG.G0021V00.

(Q) Why doesn’t the DELETE GDG FORCE command delete migrated HSM GDSs instead of recalling them?

(A) Again, good news! In z/OS V1R12, DELETE GDG FORCE deletes all HSM-migrated GDSs by using HSM HDELETE. An allocation is required to delete data sets that are not migrated. In earlier releases, migrated GDSs were recalled during allocation processing, and then they were allocated for DELETE processing. The recall process could take a long time, especially if there were a lot of migrated GDSs in the GDG. In z/OS V1R12, the code detects that the VOLSER for the GDS is “MIGRAT” and issues the HSM HDELETE command instead of recalling the data set.

Thanks, everyone! Keep the questions coming! And special thanks to Rohan Kurane!

See you next time,
Mr Catalog

MISSION IMPOSSIBLE?
...not with PPRC Manager and FlashCopy Manager!

Your mission is to move a sysplex with seven systems from your current location to a city 2,500 miles away. Not only do you need to move the sysplex, you also need a point-in-time copy of the data. And if this isn’t tough enough, there are 7,000+ volumes in the sysplex and you’re under time constraints. How are you ever going to do this?

Well… with Peer-to-Peer Remote Copy (PPRC) Migration Manager and IBM Tivoli Storage FlashCopy Manager your worries are over. Read all about these products at the following IBM Redbooks Web site:

Are you tired of having to shut your applications down to tame an unruly, ever-growing, I/O thrashing VSAM key-sequenced data set (KSDS)? Now you can say goodbye to most of your KSDS reorganizations (reorgs) with the new VSAM control area (CA) reclaim enhancement in z/OS V1R12.

The problem

KSDS, one of the most popular file structures in VSAM, consists of a B+ tree-type index, which is responsible for tracking the location of individual data records. It is a structure that allows for efficient direct and sequential insertion, retrieval, and removal of records as you can see in Figure 1:

In Figure 1, the CA pointed to by index CI ‘C’ has become empty and remains empty, becoming part of the index structure indefinitely. It can only be reused if the same deleted data-record key range is added back into the data set. When the data set is read sequentially, the level 1 index record for the empty CA and the high key (empty) data control interval (CI) in the CA must be read in order to detect if CA 3 is in fact empty. When many empty CAs exist, the system can waste much time processing them.

What's an empty CA anyway?

VSAM, like other access methods, stores records of fixed or variable length. In a KSDS, records are logically grouped into CIs. A CI is in use when it contains one or more records. CIs are then logically grouped into CAs, each CA containing a fixed number of available CIs. One or more CAs make up an extent on a direct access volume. The very first CA is created when the first record is written. More CAs are created as needed to contain

Figure 1. VSAM KSDS tree-type structure

While VSAM KSDS can provide a powerful, high-performance access method for the critical business data of z/OS customers worldwide, currently there is one limitation that can interfere with the overall success of VSAM. When records are added to the KSDS, and then erased, VSAM does not reclaim empty CAs. By not reclaiming the CAs, the data set becomes fragmented. The fragmentation causes two problems: (1) wasted disk space and, (2) performance problems when the KSDS is accessed.

In Figure 1, the CA pointed to by index CI ‘C’ has become empty and remains empty, becoming part of the index structure indefinitely. It can only be reused if the same deleted data-record key range is added back into the data set. When the data set is read sequentially, the level 1 index record for the empty CA and the high key (empty) data control interval (CI) in the CA must be read in order to detect if CA 3 is in fact empty. When many empty CAs exist, the system can waste much time processing them.

What's an empty CA anyway?

VSAM, like other access methods, stores records of fixed or variable length. In a KSDS, records are logically grouped into CIs. A CI is in use when it contains one or more records. CIs are then logically grouped into CAs, each CA containing a fixed number of available CIs. One or more CAs make up an extent on a direct access volume. The very first CA is created when the first record is written. More CAs are created as needed to contain

In Figure 1, the CA pointed to by index CI ‘C’ has become empty and remains empty, becoming part of the index structure indefinitely. It can only be reused if the same deleted data-record key range is added back into the data set. When the data set is read sequentially, the level 1 index record for the empty CA and the high key (empty) data control interval (CI) in the CA must be read in order to detect if CA 3 is in fact empty. When many empty CAs exist, the system can waste much time processing them.

What's an empty CA anyway?

VSAM, like other access methods, stores records of fixed or variable length. In a KSDS, records are logically grouped into CIs. A CI is in use when it contains one or more records. CIs are then logically grouped into CAs, each CA containing a fixed number of available CIs. One or more CAs make up an extent on a direct access volume. The very first CA is created when the first record is written. More CAs are created as needed to contain
additional CIs. When records are deleted, they are deleted from CIs. When the last record in a CI is deleted, the CI is placed on a free list within the CA and can be reused for new records anywhere in the CA.

However, until z/OS V1R12, when the last record was removed from a CA, the empty CA always stayed put. This could lead to KSDS fragmentation, which in turn wastes disk space and can degrade performance, particularly for sequential processing but also for random processing.

Reorganizing a KSDS (a reorg) is currently the only way to reclaim the unused space. The process requires the data set to be taken offline to the application, copied, deleted, reallocated, and reloaded from the copy. The larger the data set, the longer the reorg process takes. This KSDS reorg is a major obstacle for your 24x7 requirement in today’s business world.

**The solution: VSAM CA reclaim!**

Beginning with z/OS V1R12, the CA reclaim function in VSAM helps you avoid the need to reorganize almost any KSDS by automatically reusing the empty CAs.

With CA reclaim, every KSDS that is accessed by VSAM NSR/LSR/RLS methods (and that includes SMS and non-SMS managed, catalogs, and AIXs) now can reclaim CAs when all records are erased from the CA. Empty CAs are now placed on a “freechain” so they can be reused for new CAs, and that means a KSDS can reuse this space for new CA splits anywhere in the data set!

**Enabling CA reclaim**

VSAM and VSAM RLS can reclaim the empty CAs in a data set if all of the following conditions are true:

- The data set is a VSAM KSDS.
- SYS1.PARMLIB(IGDSMxx) specifies CA_RECLAIM(DATACLASS).
- The DATACLAS for the KSDS specifies or defaults to CA Reclaim=Y. This can be set during data set allocation or by using the IDCAMS ALTER command.

In z/OS V1R12, the system level default is CA reclaim not enabled (that is, the PARMLIB default is CA_RECLAIM(NONE)).

The SETSMS CA_RECLAIM (NONE|DATACLASS) command is also provided to override the CA_RECLAIM option in SYS1.PARMLIB(IGDSMxx).

CA reclaim does not reclaim empty CAs created before CA reclaim is enabled, so your key-sequenced data sets might need one last reorg. It also does not reclaim CAs that are not completely empty. To help avoid new KSDS fragmentation, all systems sharing the data set must be at z/OS V1R12 with CA reclaim enabled.

When CA reclaim is enabled, as CAs become empty the associated index CIs (both level 1 and above) are placed on two free chains, one for the level-1 index CIs and one for higher-level index CIs. Both chains are pointed to by the first second-level index CI (CI #2). Index CI #2 is never reclaimed. Figure 2 shows the resulting structure after the empty CA pointed to by index CI ‘C’ has been reclaimed:

Figure 2. VSAM KSDS structure after reclaim

Reusing empty CAs helps most data sets grow only in accordance with the amount of data they contain. Fragmentation caused by empty CA space is eliminated for all KSDSs, never to cause wasted processing time spent moving through the data set. This helps conserve your valuable DASD space and can help improve performance when the index expands and contracts as CAs are added and deleted.

**Your ticket to the future**

CA reclaim is your ticket to keeping VSAM the high performance and high availability access method it was intended to be. So kiss those pesky reorgs caused by empty CAs good-bye, and say hello to 24x7 availability with z/OS V1R12!
By Mario Verna

You remember Infoprint Server. However, you might not know that the last enhancement made to Infoprint Server as part of a z/OS release was back in z/OS V1R8! Much has changed since then. For one thing, Ricoh bought the Printing Systems Division. Nevertheless, Infoprint Server remains an IBM product as an optionally priced feature of the base operating system. That’s right; Infoprint Server is shipped as part of the base operating system. Beginning with z/OS V1R12, Infoprint Server has been significantly enhanced to relieve constraints and reduce spool occupancy. All enhancements to Infoprint Server now ship through a z/OS release instead of through the service stream.

Stay in your pajamas

Infoprint Server is an optionally priced feature of z/OS that uses z/OS UNIX System Services. This feature lets you consolidate your print workload from many servers onto a central z/OS print server. One of the many components of Infoprint Server is Infoprint Central, which is designed to provide a secure Web-based print management system. That means you can manage print jobs, printers, check Infoprint Server system status, and many other things from home—in your pajamas if you want.

Keeping up with the JESes

In z/OS V1R12, Infoprint Server provides new capabilities for greater flexibility in managing output. Some of the changes include:

- The IP PrintWay™ extended mode component now manages a greater number of SYSOUT data sets with similar output attributes. This means you can submit jobs that create a large number of output data sets (approximately 32,640) on the JES spool for Infoprint Server to process. To support the large number of output data sets in the same print job, you must reformat the Infoprint Server printer inventory when you first migrate to V1R12. The first time you start the Infoprint Server on V1R12, the existing printer inventory automatically uses the new format.
- IP PrintWay extended mode now supports the same response notification exit as IP PrintWay basic mode. The response notification exit lets you take an action based on the status of the transmission of an output data set to a printer. For example, the exit can send a message to the operator’s console, the IP PrintWay message-log data set, or both when a transmission to a printer fails.
- Infoprint Server gives higher priority to spooling and printing of existing print jobs before receiving new print jobs.
- Infoprint Server has capacity for a greater number of active jobs in the system at one time, up to the maximum that JES currently allows.
- Infoprint Server line printer daemon (LPD) now receives files larger than 2 GB. For example, this means that you can use TCP/IP line printer requester (LPR) command or the Infoprint Server Port Monitor for Windows® client to manage large files through Infoprint Server.
- The Internet Printing Protocol (IPP) Server component of Infoprint now requires Java V6.0.
- The Infoprint Central component now requires IBM XML Toolkit for z/OS V1.10 edition. One enhancement delivered in V1R12 provides authorized users the ability to limit how many print jobs and documents Infoprint Central displays.

Considerations

APAR OA32093 (an Infoprint Server co-existence APAR to apply before migration to z/OS V1R12) is available to assist you in migrating to z/OS V1R12 and, if required, fallback to z/OS V1R11 or z/OS V1R10. For some additional migration information, see “z/OS strikes 12! Migrating to z/OS V1R12” on page 11.
BY JONATHAN COTTRELL, SAHEEM GRANADOS, AND RAINTER HIMMELSBACH

By choosing IBM System z, many customers have moved their core enterprise systems to the highest security levels. The IBM Tivoli Directory Server for z/OS ships as part of the base operating system, and supports many z/OS security and high availability functions like the following:

- Automatic Restart Management (ARM)
- z/OS Security Server Resource Access Control Facility (RACF)
- Workload Management (WLM)
- HiperSockets™ (wireless connection)
- File-based and DB2® based sysplex clustering.

This article discusses two new security-related features new to the IBM Tivoli Directory Server for z/OS in V1R12. It also provides general information about how to configure the SAP NetWeaver application with the Tivoli Directory Server for z/OS.

Enhancements: extended mode only
As has been the case since z/OS V1R5, all enhancements to Infoprint Server for z/OS for printing in a TCP/IP and SNA network are made only in IP PrintWay extended mode and not in IP PrintWay basic mode. (IP PrintWay extended mode provides improved usability and more function than IP PrintWay basic mode.) With z/OS V1R8 and above, APAR OA26583 introduces you to the INFOPRINT_PRINTWAY_MODE health check, which issues an informational message indicating extended mode is preferred when it detects basic mode is in use. IP PrintWay extended mode uses the SYSOUT application programming interface to obtain output data sets from the JES spool.

More information
For a complete understanding of the steps necessary to use Infoprint Server in z/OS V1R12, see z/OS V1R12 Migration, GA22-7499-17, and the following Infoprint Server publications:

- z/OS Infoprint Server Customization, S544-5744-11
- z/OS Infoprint Server Operation and Administration, S544-5745-11
- z/OS Infoprint Server Introduction, S544-5742-10
- z/OS Infoprint Server User’s Guide, S544-5746-10
- z/OS Infoprint Server Messages and Diagnosis, G544-5747-10
- z/OS Infoprint Server Printer Inventory for PSF, S510-7703-02.

The height of “z” curity!

Tightening the screws
In z/OS V1R12, the Tivoli Directory Server for z/OS introduces features for password policy and access control filter support to give LDAP administrators two powerful tools for even more data security control in the LDAP server.

Password policy support
What is password policy? Password policy is a set of rules to control how the LDAP server administers and uses passwords. Password policy ensures that users have strong passwords that cannot easily be compromised and are changed periodically. These rules:

- Restrict the reuse of old passwords
- Lock users out after a defined number of failed bind attempts
- Automatically expire passwords after a specified period of time.
Configuring password policy

Now, doesn’t having automatic password policy in your LDAP server sound like a great idea? We’re certain it does. To configure a password policy, we’ve summarized a few simple updates needed in the LDAP server configuration file:

1. Add the following configuration option to the CDBM backend:

   ```
   database cdbm GLDBCD31/GLDBCD64
   ```

2. Ensure that the serverCompatLevel option is set to 6.

After these two steps are complete, restart the LDAP server. The cn=pwdPolicy,cn=ibmpolicies entry is automatically created in the CDBM backend. The cn=pwdPolicy,cn=ibmpolicies entry is known as the global password policy entry and applies to all LDBM and TDBM entries that have userPassword attribute values. The global password policy is not activated until you change the ibm-pwdPolicy attribute from false to true using an LDAP modify command. Use the ldapmodify utility to update the ibm-pwdPolicy attribute. For example:

```
ldapmodify -D adminDN -w adminPw -f modPolicy.ldif
```

where modPolicy.ldif has the following contents:

```
dn: cn=pwdPolicy,cn=ibmpolicies
changetype: modify
replace: ibm-pwdPolicy
ibm-pwdPolicy: true
```

Here is a summary of changes for password enhancements:

- **Global password policy entry**
The global password policy entry has many attributes that control various aspects of allowed userPassword attribute values in LDBM backend and TDBM backend entries. The password policy attributes give the LDAP administrator many knobs for fine-tuning the password policy for the organization. These controls include automatic password expiration, requiring a minimum password length, and locking users after too many failed authentication attempts.

- **Additional password policy entries**
You can define additional password policy entries under the cn=configuration entry in the CDBM backend. These entries can apply to certain individual users or groups that must have a policy differing from the global policy for security reasons to provide even more control.

- **Password policy operational attributes**
For TDBM and LDBM user entries that are subject to password policy, the LDAP administrator can query several operational attributes in the user entries in order to obtain password policy state information such as when the password was last changed and a history of previous password values.

- **PasswordPolicy control**
You can also update LDAP client applications to send the PasswordPolicy control on requests to solicit additional warning and error information from the LDAP server related to password policy enforcement, for example, to find out if the length of the new password is not long enough.

**Access control filter support**

Access control filter support is an extension to the standard access control list (ACL). These controls allow more granular permissions to be set for users in the LDAP server and allow permissions to be augmented, reduced, or replaced based on logical combinations of the following attributes:

- Bind distinguished name (DN)
- Alternate DN
- Groups to which the bind DN or an alternate DN belong
- IP address of the client connection
- Time and day of week the entry was accessed
- Authentication mechanism
- SSL connection status.

You might ask, “Why would anyone want this granularity? Isn’t this extra level of granularity too complicated to implement?”

Now consider a mobile workforce. Many times legal and regulatory requirements can require access control to be different depending on the location where a user attempts to access a resource. Alternatively, IT security officers might like to enhance security with additional attributes that the LDAP directory doesn’t represent, for instance, time or day of access.

This new extension to the LDAP access control model provides the means and flexibility to security officers so that they have dynamic access control using common LDAP constructs. Enterprises that require different access control given a user’s location, time, or even connection type would simply add new aclEntry or entryOwner attribute values to protected LDAP objects.

**An aclEntry example**

For example, the supported aclEntry and entryOwner attribute values have been extended in V1R12 to include new syntaxes. The following is an example of an aclEntry value that uses the new filtered aclEntry syntax:

```
aclexentry :aclFilter: 
(& (ibm-filterDayOfWeek=1) 
 (ibm-filterDayOfWeek=5)) :union:critical:rwsc: restricted:rwsc
```

You can use this aclEntry to augment any matching aclEntry values for a user who attempts access on Monday through Friday. The aclFilter keyword (ownerFilter for entryOwner values) is required for these new filtered values. The string (ibm-filterDayOfWeek=1) (ibm-filterDayOfWeek=5), is a standard LDAP filter that uses a predefined set of attributes to represent the user’s new dynamic attributes (in this case, allow, access Monday through Friday). The keyword :union: indicates that Tivoli Directory Server for z/OS is to augment the aclEntry values. Finally, the string critical:rwsc:restricted:rwsc represents the permissions to augment. The entryOwner attribute values have a similar syntax that allows entry ownership to be granted or denied dynamically.

**Password policy management and access control filter support = more flexible security**

For z/OS V1R12, these new powerful security features give IT security officials the flexibility to really strengthen the security of an IT enterprise. To help address the many legal and regulatory requirements needed these days to secure the LDAP.
your organization, Tivoli Directory Server for z/OS V1R12 provides both password policy management and access control filter support with extended operations to allow an administrator to test deployed aclEntry and entryOwner values.

The SAP application case
Tivoli Directory Server for z/OS is part of the base operating system, so it is security that is provided virtually for free to SAP customers on IBM System z. You can zecure—pardon, secure—your SAP applications with IBM System z, deploying tools that the acclaimed, highly secure System z platform can offer. Of course, the Tivoli Directory Server for z/OS and the SAP application require some set up and configuration steps.

In our IBM SAP test environment, we successfully completed steps to achieve these application configurations:

• Setting up and configuring SAP NetWeaver Application Server Java and SAP NetWeaver Enterprise Portal with the User Management Engine (UME) together with Tivoli Directory Server for z/OS
• Setting up and configuring SAP NetWeaver application server ABAP user repository to synchronize with Tivoli Directory Server for z/OS.

Our IBM SAP test environment includes the following hardware and software (See Figure 1).

• IBM System z9® Enterprise Class and two Linux on System z guests that were hosted by z/VM®
• Two LPARs defined in a sysplex with one that operated as LDAP server and the other as data base server
• DB2 9 for z/OS as the data base backend.
• Network connection that was over a public LAN and included fast HiperSockets for the connection from Linux on System z to z/OS.

Additional information
For a detailed description of the implementation of these configurations, see "SAP and IBM Tivoli Directory Server for z/OS" available on the following SAP Web site:


For information about password policy and filtered ACL support, see IBM Tivoli Directory Server Administration and Use for z/OS V1R12.0, SC23-5191-05.
The Crypto Express3 feature is the most recent IBM cryptographic feature and complements the cryptographic functions of the CP Assist for Cryptographic Function (CPACF). The state-of-the-art Crypto Express3 feature rides the crest of cryptographic history, providing the tools necessary to implement the latest advances in cryptographic technology. To understand how the Crypto Express3 can meet today’s cryptographic requirements, let’s take a look back at the history of cryptography and then look around at today’s challenges.

Some history
For thousands of years, people have wanted to keep information private. One such person was Julius Caesar. To support his military operations, Caesar used a cryptosystem to encipher and decipher messages. The substitution method he used is now called the Caesar Cipher. With his cipher, one letter of the alphabet is substituted for another to convert a message from plaintext to ciphertext. Caesar used this cryptosystem to exchange secret messages with his generals.

During World War II, the German military used an electro-mechanical machine called Enigma to encrypt and decrypt messages. Over the years, Enigma gave birth to many variants, but the advent of the digital age advanced a new era of cryptography.

As the years pass, cryptographic algorithms and protocols become more and more sophisticated. However, some of the fundamental algorithms used today in symmetric and asymmetric cryptography were introduced in the 1970s. Some algorithms were rendered useless by cryptanalysts and researchers who found flaws or weaknesses in them. Increases in computing power can also leave previously effective algorithms susceptible to brute force attacks. This trend continues. As long as there are cryptographers developing cryptosystems, there are cryptanalysts trying to break these systems.

Today’s challenges
System owners, cryptographic module vendors, protocol designers and application developers must work together to develop secure systems, and they all must keep up with advances in cryptographic technology. Organizations like the International Organization for Standardization (ISO), National Institute of Standards and Technology (NIST), and National Security Agency (NSA) publish recommendations and best practices for developing secure systems and outline effective use of cryptographic algorithms. In addition, business industries have their own standards as do national and international governments. The recommendations indicate what to use an algorithm for, and when to retire an algorithm.

In addition to understanding the changes in technology, system owners need to understand which algorithms and protocols their businesses use. Not every system owner can afford to have a crypto specialist in their ranks. Sometimes, system owners purchase security solutions rather than develop an in-house solution.

There is more to keeping your system secure than just switching to a different set of algorithms or protocols as the old ones reach the ends of their lives. A solid transition plan must be built with the help of highly trained security professionals. After identifying the right solutions, the system owner does not simply switch to the latest algorithms all at once. Transitioning takes time. To provide a smooth migration path from the prior technology, the products must support both the old and the new mechanisms during the conversion period.
Crypto Express3 meets the challenges
The Crypto Express3 feature is suited to applications that require Rivest-Shamir-Adleman (RSA) algorithm acceleration, cryptographic operations for data encryption and digital signing, secure management and usage control for cryptographic keys, specialized banking cryptographic functions, or custom cryptographic applications. Common uses include financial applications such as PIN generation and verification in automated teller and point-of-sale (POS) transaction servers, remote key loading of ATMs and POS terminals, Web-serving applications, public key infrastructure applications, smart card applications, and custom proprietary solutions.

Crypto Express3 provides the tools you need to implement an algorithm and key usage transition plan. Crypto Express3 maintains support for old algorithms while expanding the cryptographic algorithms that are supported. In addition to supporting Data Encryption Standard (DES and TDES), Secure Hash Algorithm (SHA), and RSA, Crypto Express3 also supports:

- Advanced Encryption Standard (AES) key lengths 128, 192, and 256
- SHA-256, SHA-384, SHA-512
- Hash-based Message Authentication Code (HMAC)
- Elliptic curve cryptography (ECC)
- Elliptic Curve Digital Signature Algorithm (ECDSA).

Let’s clear up this crypto confusion!

Understanding the importance of cryptographic coprocessors

BY KATHLEEN FADDEN, STEVEN HART, AND BOB PETTI

The z/OS Cryptographic Services Integrated Cryptographic Services Facility (ICSF) uses hardware cryptographic features in a way designed to provide secure cryptographic services in the z/OS environment. The cryptographic hardware features that are available to you depend on your server:

- On z800 and z900 servers, available cryptographic hardware consists of the PCI Cryptographic Coprocessor (PCICC), which works in conjunction with the Cryptographic Coprocessor Feature (CCF).
- Starting with the z890 and z990 servers, and on all later server models, CCF is replaced by a combination of CP Assist for Cryptographic Function (CPACF) and attached cryptographic coprocessors. With CPACF-based hardware, much of the cryptographic function is consolidated on the cryptographic coprocessors, with only a limited set of functions available in the native CPACF hardware.

The shift from CCF-based hardware to CPACF-based hardware has caused some confusion. A common mistake is...
stemming from this confusion is trying to use a cryptographic function that requires a cryptographic coprocessor when no cryptographic coprocessor is present. A similar mistake is trying to use a cryptographic function that requires master keys to be set on a cryptographic coprocessor when master keys have not been set. So when do you need a cryptographic coprocessor? And when do you need master keys set? Let’s see if we can clear up this crypto confusion on CPACF-based System z hardware.

**When do I need an active cryptographic coprocessor?**

In general, the need for cryptographic coprocessors depends on whether you’re using encrypted key cryptography or clear key cryptography:

- Encrypted key operations provide a high level of protection that prevents key material from ever appearing in the clear within application storage or the keystore. All encrypted-key operations in ICSF require an active cryptographic coprocessor with its master key registers initialized.
- Clear key operations do not provide the high level of protection that encrypted key operations provide. For clear key operations, the base key value is stored within the keystore and might appear within application storage when used.

So when do you need an active cryptographic coprocessor on CPACF-based System z hardware? If you want to use any encrypted key operation, you need an active cryptographic coprocessor with master keys set. In fact, most ICSF operations and utilities require an active cryptographic coprocessor with master keys set.

Furthermore, in order to perform master key entry using the ICSF panels, utilities, or the PPINIT utility, you must have an active cryptographic coprocessor. (The master key entry does not require the master key registers to be initialized in advance because that is the purpose of the master key entry.)

**An active coprocessor with master keys set is required!**

An active coprocessor is required to set master keys, and an active coprocessor with master keys set is required to perform encrypted key operations and to use the majority of other services and utilities. Among these other services and utilities are:

- PKA services
- PIN services
- Digital signature services, except for clear ECC keys
- PKDS Key Management services
- Message Authentication Code (MAC) services (except for the One-Way Hash Generate service)
- Key Generator Utility Program (KGUP)
- ENCODE utility
- DECODE utility.

The Random Number Generate (CSNBRNG) service requires an active coprocessor and (depending on the version of ICSF your system is running) might require master keys to be set. Starting in ICSF FMID HCR7770, Random Number Generate only requires an online coprocessor. In earlier releases of ICSF, Random Number Generate required an active cryptographic coprocessor with master keys set.

**Without an active coprocessor, expect limited function**

Given that the vast majority of ICSF operations use the cryptographic coprocessors, z/OS environments without them are limited to only a subset of cryptographic functions. The following ICSF callable services and utilities do not require a cryptographic coprocessor or master keys to perform clear-key operations and other miscellaneous operations:

- Key record create (CSNBKRC)
- Key record write (CSNBKRW)
- Key record delete (CSNBKRD)
- Key record read (CSNBKRR)
- Symmetric key decipher (CSNBSYD)
- Symmetric key encipher (CSNBSYE)
- Symmetric MAC generate (CSNBSMG)
- Symmetric MAC verify (CSNBSMV)
- Key token build (CSNBKTB)
- One way hash (CSNBOWH)
- MDC generate (CSNBMGD)
- Character/Nibble conversion (CSNBXBC and CSNBXCB)
- Code conversion (CSNBXEA and CSNBXAE)
- ICSF query algorithm (CSFIQA and CSFIQA6)
- ICSF query facility (CSFIQF and CSFIQF6)
- X.9.9 data editing (CSNB9ED).

For more information

Hopefully this article has cleared up some of the crypto confusion by showing that ICSF functionality is largely based on the use of cryptographic coprocessors with master keys set.

- For information on cryptographic coprocessors, see z/OS Cryptographic Services ICSF Administrator’s Guide, SA22-7521.
- For information on ICSF callable services, see z/OS Cryptographic Services ICSF Application Programmer’s Guide, SA22-7522.
Spotting the bad apple

XCF and GRS team up and save the bunch

BY CHRIS BROOKER, JOSEPH GENTILE, AND TAMMY GARREN

The Global Resource Serialization (GRS) star environment built on Parallel Sysplex has been heralded for its scalability and unparalleled availability (pun intended). However, even in this robust environment, “one bad apple” can still spoil the bunch. Although rare, a System z outage is extremely costly for a business, especially when multiple systems are affected. In an effort to avoid these situations, z/OS development examines every multi-system outage to determine if IBM can take any action.

Meet the critical component

In order to address these issues, XCF is delivering function in z/OS V1R12 that extends XCF system status monitoring to other critical components using XCF services. Now, a component can register itself as critical to XCF by specifying CRITICAL=YES when joining an XCF group. XCF then monitors the component’s XCF group related routines. The component can also optionally register a status field, interval, and exit to be monitored by XCF. As long as the status field changes during the interval, XCF considers the component to be healthy. If the status field is unchanged, XCF drives the status exit to determine the health of the component.

If XCF determines that the component is unhealthy, it starts to monitor it more closely, and can eventually terminate that component’s task, address space, or system if the component does not recover. The action that XCF takes and the timing of that action depends on how the component registers for monitoring as well as the system failure detection interval (FDI) and MEMSTALLTIME parameter, both of which are customizable by the installation. See Figure 1.

GRS is the first component to use XCF critical member monitoring in z/OS V1R12. At regular intervals, GRS will check on the global ENQ server task and the global GQSCAN driver task. If either of these tasks is not progressing, it can affect the other members of the sysplex. Using this function, GRS will alert XCF whenever it notices a problem and record diagnostic data, including the name of the task causing the problem, and the time the problem occurs. Because GRS services are critical to z/OS, the system terminates when GRS is unhealthy for too long. Although taking a system down is not desirable, terminating one system is better than letting one task take down the entire sysplex!

Special thanks

Special thanks for this article go to the rest of the GRS Development and Test Team: Bryan Childs, Nick Matsakis, Steven Partlow, and Thomas Rankin.

Find out more

To find out more about XCF APIs for critical members, see the following publications:

• z/OS MVS Sysplex Services Guide, SA22-7617
• z/OS MVS Initialization and Tuning Reference, SA22-7592.
Defining a DASD or tape subsystem to z/OS requires careful planning on the number and selection of paths to provide for sufficient bandwidth and availability, and to avoid single points of failure. Chosen device numbers have to follow the installation conventions, and data that has already been specified when configuring the subsystem needs to be duplicated for the I/O definition, like LCU numbers, unit address ranges, or device types. Any mistakes or transcription errors can lead to a sub-optimal or nasty, failing configuration.

A kinder, simpler z/OS
Are you dreaming of a kinder, simpler z/OS that could search all of the storage “fabrics” (see First Phase: fabric discovery), determine all of the attached hardware and automatically configure new devices? z/OS V1R12 provides a new function in Hardware Configuration Definition (HCD) and Hardware Configuration Manager (HCM) that allows you to discover new and changed DASD and tape controllers that are connected to FICON® directors. You can then select one or more controllers and let HCD perform an automatic definition of the new control units and devices that have been configured in the controller.

HCD and the HCD policy
HCD works with the I/O Supervisor (IOS) to perform the actual discoveries and definition proposals. This process works for all of the systems in the active sysplex. A user specified policy, stored in the HCD profile, controls the discovery process. The policy information includes things such as:
- Control unit and device numbering scheme
- Number of paths to define for a control unit
- Accessibility of the devices by specific systems.

A user can also define logical partition (LP) groups to restrict the definition to a subset of the systems in the sysplex, and OS configuration groups to control the connection of discovered devices to specific OS configurations. A dialog is available to set and store this policy information in the HCD profile.

You can invoke the I/O auto-configuration function in HCD using option 1.6 “Define, Modify, or View Configuration Data – Discovered new and changed control units and I/O devices.” On the first panel you can set the following processing options:
- Target IODF name
- Scope of discovery
- Automatic or expert mode.

The discovery process
To ensure that the discovery process results in a reasonable definition of the new items, the target IODF must be the active production IODF, or an IODF created from the production IODF. You can choose the scope of discovery to find only new (that is, one not already defined) controllers, or both new and changed controllers. You can also restrict discovery to a single controller based on a control unit that is already defined in the I/O configuration.

First phase: fabric discovery
The first phase of the discovery and autoconfiguration process is fabric discovery. In this phase, z/OS determines which channels to explore for each target system. A device to use for fabric discovery is configured on each system and connected to the first channel to be...
explored. z/OS communicates with each fabric name server to determine reachable destinations, and subsequently identifies the channels and storage controllers at each one. This process continues, connecting the fabric discovery device to the next channel to explore on the target systems. After HCD explores all channels, fabric discovery is complete and a list of controllers is identified. For each discovered controller, all possible channels and destination ports that can reach each controller are also identified.

HCD presents the newly detected or changed controller nodes in a list where you can select those entries to define in the IODF. See Figure 1.

Second phase: controller discovery
The second phase of the discovery and autoconfiguration process is controller discovery. Controller discovery is responsible for determining whether any new control units and devices exist on the storage controller, and proposing changes to the I/O configuration. If possible, the system performs controller discovery by using devices on the controller that are currently defined to z/OS or by configuring temporary devices for each control unit that exists on that controller.

The system uses controller-specific I/O commands to determine the proper device configuration for each control unit on the controller.

If the system finds new control units, it selects paths to connect each target system to the new control units. The discovery and autoconfiguration policy, described earlier, contains the number of paths to configure and the number of static and managed paths. The static paths assigned to a control unit are chosen from the available channels and destination ports that can reach the control unit, minimizing single points of failure. For example, if you are configuring two static paths, you can use channels that are on different channel cards and connected to different switches, if possible.

Dynamic channel path management
Because paths are chosen for availability, it is a good idea to use the dynamic channel path management (DCM) to manage I/O performance. For example, choosing four static paths and four managed paths allows the static paths to be chosen to avoid single points of failure, and allows paths to be added or removed to manage for performance.

For control units that already existed in the target IODF, additional devices might be found that are not currently configured. For this case, information about the missing devices is returned. You do not need to select any paths because the control unit is already defined in the target IODF.

The IODF definition
The policies stored in the HCD profile determine the IODF. When the controller discovery phase completes, HCD checks the proposed configuration for existing definitions in the target IODF:

- For each discovered control unit that is already defined with the same control unit address (CUADD) value, the existing control unit definition is checked for the same serial number. If the serial numbers match or the IODF definition does not contain a serial number, the control unit number of the existing control unit is used. If the serial numbers do not match, the system issues a warning message that proposes a new number for the discovered control unit.
- For each discovered control unit that is not already defined in the IODF, a new number is proposed.
- When a new control unit number is proposed, the system selects the value from the preferred range specified by policy AUTO_SUG_CU_RANGE.
- If a free control unit number does not exist in the IODF within that range, the system issues a warning message that indicates the policy could not be followed and instead proposes an unused control unit number outside of the range.
- For each discovered device that is already defined with the same unit address on an existing control unit, the existing device number is proposed. For non-existing devices on the control unit, HCD uses the existing device numbering scheme, if possible.
- For new devices on new control units or on existing control units when the system cannot follow the existing device number scheme, the device numbers are determined based on policies AUTO_SUG_DEV_RANGE and AUTO_MATCH_CU_DEVNUM for PAV alias devices additionally based on policies AUTO_SS_ALTERNATE...
Proposed existing control units are updated with the discovered serial number. If the type of a discovered control unit differs from its definition in the IODF, the system updates the definition.

**Proposed Control Unit List**

If you previously selected automatic mode, HCD defines the control units and devices without requesting your intervention; however, in expert mode, you are presented the definition proposals, first, for the control units, and then for the devices. You can accept the proposed information, exclude some of them from being defined, or change some of the proposed values. See Figure 2.

Hardware Configuration Manager (HCM) also provides an interface to this function as a wizard that interfaces with HCD for the discovered items and defined data. You can specify the same options and policies as in HCD, and the processing is similar. At the end of the wizard, HCD presents a summary page of the discovered items and their definition proposal before you accept or reject the data to be included in the IODF.

I/O dreaming

Using the z/OS discovery and auto-configuration function with HCD or HCM greatly simplifies the I/O definition of new or changed storage devices. It’s the stuff of your I/O dreams and brings you closer to a kinder, simpler I/O configuration task.

---

**z/OS V1R12 provides a new function in Hardware Configuration Definition (HCD) and Hardware Configuration Manager (HCM) that allows you to discover new and changed DASD and tape controllers ... connected to FICON directors.**

---

**Figure 2. Proposed control unit list**
Coding the globe with Unicode

BY THUY CHRISTENSON, PAT GLENSKI, AND RITA BEISEL

In today’s global economy, more companies than ever are doing business internationally and the need to process text data from across the globe has grown tremendously. With Unicode and IBM z/OS Unicode Services, your applications can support a global audience. Unicode is the fundamental building block for language encoding on the Internet. With Unicode, you can handle text in any language efficiently so that a single application to work for a worldwide audience.

It might not be necessary for your company to create Web applications in different languages today. However, when you do need to expand from your local language, Unicode Services are here to help!

What is Unicode and why is it important?
Unicode provides a unique number for every text character, regardless of platform or language. It enables application source code to be written to process data in any language, and makes it easy to add new language support to an application. Unicode is an industry standard that is maintained by the Unicode Consortium, of which IBM is a member.

Before Unicode was invented, there were hundreds of different encoding systems for representing the text required for different languages and different systems. Today, Unicode allows data from many different languages to be stored in one repository. The latest version of Unicode contains a listing of more than 100,000 text characters.

Unicode is important because it is the basic text encoding used by Web standards and applications such as XML, Java, Lightweight Directory Access Protocol (LDAP), Common Object Request Broker Architecture (CORBA), Wireless Markup Language (WML), Google Docs, Open Office, and others. Using Unicode, you can develop an application that works with various languages, yet maintains text data integrity. This makes text data portable.

What services do Unicode Services provide?
On z/OS, Unicode Services enable your applications to use Unicode. These services do the basic work to support Unicode so only a small update to your application might be needed. Unicode Services consist of three main services:

• Character conversion service
• Case conversion service
• Collation service

Character conversion service
You might be wondering, “If my Internet data is in Unicode and my application data is in EBCDIC, how do I convert the Internet data to be usable to my application?” The answer is to use the character conversion service.

The character conversion service converts data from one coded character set identifier (CCSID) to another, for example, from EBCDIC to Unicode or vice versa.

Unicode Services support the UTF-8, UTF-16, and UTF-32 Unicode encodings. In addition, the character conversion service supports over 500 different encodings other than Unicode and can convert to and from them all.

If you are a C or shell programmer, you might already be using Unicode Services and not know it. Since z/OS V1R9, the Language Environment® iconv interface uses Unicode Services to do character conversions.

Case conversion service
You might be thinking, “EBCDIC was simple. It only had a few characters and to convert them to uppercase, you only had to add X’40’. Unicode contains over 100,000 characters, and there are many rules on converting them to uppercase or lowercase. How can I convert text characters easily?” The answer is to use the case conversion service.

The case conversion service converts Unicode data to uppercase or lowercase using the rules provided by Unicode. Unicode Services also support locale-sensitive casing for more complex cases.

Collation service
You might ask yourself, “How do I sort data that contains all of the Unicode characters?” The answer is to use the collation service.
The collation service allows for culturally correct comparisons between two Unicode input strings. The collation service takes a Unicode string and provides you with a collation key that your code can use to do simple hex sorting. You can also use this service to generate a sort key for one or two Unicode strings.

After a sort key is generated, it can be kept and used later to do comparisons between other sort keys.

**Programming languages**

Unicode Services work with C, C++ and HLASM programming languages. These programming interfaces share the following characteristics:

- Both 31-bit and 64-bit addressing mode versions are provided
- Callable from any key
- Callable from problem or supervisor state
- Callable in task or SRB mode
- Callable in cross-memory mode.

The header files and sample code to invoke Unicode Services are in the SYS1.SAMPLIB data set.

**Unicode on-demand**

Starting in z/OS V1R7, Unicode Services dynamically loads conversion tables into storage if the appropriate conversion table is not already there. This is known as Unicode on-demand or dynamic loading of conversion tables. All conversion tables needed for character conversion, case conversion, normalization, and collation services are loaded into storage when they are required, and are added to the other conversion tables already in storage.

To see the Unicode conversions that are loaded, issue the MVS command on the system console:

```plaintext
DISPLAY UNI,ALL
```
Log size matters
Sizing your log stream data sets

BY ANDREW M. SICA AND NICHOLAS R. JONES

When you configure system logger, sizing your log stream data sets can be a puzzle. The system logger default log stream sizes might be too small, but there's a 2 GB limit to log stream data set size to work within also. You know you need to look at sizing the data sets, but what's a good starting point?

First, let's define our terms. Data in a log stream spans two kinds of storage:

- Interim storage, where data can be accessed quickly without incurring DASD I/O. Interim storage is either in coupling facility list structures or in a DASD-only log stream, in local storage buffers.
- DASD log data set storage, where data is "hardened" for longer term access. These are the data sets we'll look at in this article.

System logger uses two types of DASD log stream data sets:

- Offload data sets contain log data no longer managed in interim storage.
- Staging data sets contain a backup copy of interim storage for recovery.

The really good news is that z/OS V1R12 along with APAR OA30548 introduces support for up to 4 GB offload and staging data sets, doubling the size limit. This should help anyone constrained by the old 2 GB limitation. With this enhancement available, it's a great time to review some pitfalls involved in sizing log stream data sets.

The big cost of too small
Sizing your offload data sets too small can drive frequent data set allocations and delay the offload process. In addition, these frequent allocations can cause data set extents to run out, eventually leading to write failures.

Sizing your staging data sets too small can cause the staging data set to be perpetually full, driving constant offload activity and making the log stream frequently unavailable for writes.

Don't use the defaults without making some calculations
If you don't specify data set size parameters in the LOGR policy, allocation will use your system's default, which could be as small as 2 tracks. The following actions can help you right size your data sets:

- To display your data set sizes currently in use, use the LISTCAT option on the IXCMIAPU LOGR report.
- To change your data set sizes, update IXCMIAPU parameters LS_SIZE, STG_SIZE, LS_DATACLAS or STG_DATACLAS.

The documentation for STG_SIZE and LS_SIZE in MVS Setting up a Sysplex defines how system logger calculates a default for your system if you do not specify a value.

Can't extend?
Having enough data set extents can help ensure that you don't run short of log data set space. If you run out of extents, system logger might not be able to offload log stream data and interim storage will fill up, resulting in applications being unable to write to the log stream.

You can manage the number of data set directory extents using the DSEXTENT parameter in the LOGR couple data set format utility IXCL1DSU. If you have log streams that require more than 168 DASD log data sets, increase the value of DSEXTENT. Each additional directory extent specified goes into a common pool available to any log stream in the sysplex. System logger allocates these directory extents as needed when a log stream runs out of DASD log data set directory space. Each directory extent allows a log stream to extend by 168 additional log data sets.

To monitor your sysplex for an extent shortage, keep an eye out for system messages IXG257I, IXG261E, IXG262A, and IXG301I. The IXCMIAPU report with the REPORT(YES) option also displays the number of DSEXTENTs in use.

We also suggest that you define your alternate LOGR couple data set with extra DSEXTENT records. That way, if a shortage occurs, a quick PSWITCH may avoid an outage.

Offload threshold: Too high or too low?
The frequency of offloads can influence the efficiency of DASD-only log streams, especially if you define large log data sets. You can manage the frequency of offloads by defining offload thresholds on the log stream definition in the LOGR policy using the HIGHOFFLOAD and LOWOFFLOAD parameters. For a DASD-only log stream, if the differences between the HIGHOFFLOAD and LOWOFFLOAD values are too great, delays and write failures might occur during offloads. Having large staging data sets can make delays worse because DASD-only log streams do not delete log data until the offload completes.
z/OS V1R12 along with APAR OA30548 introduces support for up to 4 GB offload and staging data sets, doubling the size limit. This should help anyone constrained by the old 2 GB limit.

Here’s an example. If a log stream defaults to HIGHOFFLOAD(80) LOWOFFLOAD(0), then 80% of the staging data set size has to be moved during the offload and only 20% is available for use. If this 20% is exhausted, the application will receive a staging data set full condition until the offload completes.

To avoid this problem, ensure that DASD-only log stream HIGHOFFLOAD and LOWOFFLOAD values allow for adequate space to buffer incoming write activity during offloads. To view log stream parameters like HIGHOFFLOAD and LOWOFFLOAD, look at the IXCMIAPU LIST LOGSTREAM output.

**References**

For more system logger tuning information see the following publications:

- IBM Redbooks *System Programmers Guide To: System Logger*, SG24-6898
- *MVS Setting up a Sysplex*, SA22-7625.

**Acknowledgments**

Thanks to Doug Zobre and Donny Chan for their contributions to this article.

---

**zFS takes ownership when and where you need it**

*Sysplex-aware support on a file owning basis*

**BY JIM SHOWALTER, MARY NG, AND SCOTT MARCOTTE**

K eeping track of the owner of file systems in a multisystem sysplex can be a bit tricky. When you run in a shared file system environment, you need to understand the implications about which system is the owner of a read-write mounted file system. If you don’t ensure that the file system owner is the system that is doing more of the file requests to the file system, you can pay a performance penalty. That’s because file requests from a system that is not the owning system go through cross-system coupling facility (XCF) communications to the owning system for each request, which can increase processor usage. Figure 1 shows what this situation looks like.

In Figure 1, SY2 is the owning system. File requests from a different system (say, SY1), are sent from the application layer through the sysplex owner to SY2.

**Figure 1. File requests in a shared file system environment before zFS sysplex-aware support**

<table>
<thead>
<tr>
<th>SY1</th>
<th>SY2</th>
<th>SY3</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS</td>
<td>z/OS</td>
<td>z/OS</td>
</tr>
<tr>
<td>UNIX</td>
<td>UNIX</td>
<td>UNIX</td>
</tr>
<tr>
<td>appl</td>
<td>appl</td>
<td>appl</td>
</tr>
<tr>
<td>z/OS UNIX</td>
<td>z/OS UNIX</td>
<td>z/OS UNIX</td>
</tr>
<tr>
<td>owner</td>
<td>owner</td>
<td>owner</td>
</tr>
<tr>
<td>z/FS</td>
<td>z/FS</td>
<td>z/FS</td>
</tr>
</tbody>
</table>

Read/Write
to z/OS UNIX on the requesting system (SY1), and then sent to z/OS UNIX on the owning system (SY2) through XCF communications and then finally to zFS on the owning system (SY2). This extra pathlength makes file requests from SY1 take longer than file requests from SY2 and thus the importance of choosing the right owning system. The system where the file system is initially mounted normally owns the file system, but can move when the owning system is shut down (or is the result of an explicit movement).

**zFS sysplex-aware support in z/OS V1R11**
Beginning with z/OS V1R11, zSeries File System (zFS) provides sysplex-aware support and additional enhancements with APAR OA29619. This support can make ownership of the file system much less important because zFS read-write file systems can now be sysplex-aware.

**What it really means**
When a zFS read-write file system is sysplex-aware, file requests go to z/OS UNIX on the local system, and then directly to zFS on the local system. When in zFS on the local system, a decision is made whether the request needs to be sent to zFS on the owning system. zFS maintains cache consistency through a token management mechanism by using a distributed locking scheme as shown in Figure 2.

**Who owns what**
In addition, designed to further improve performance, zFS can dynamically move ownership to another system (say, SY3) if substantially more requests are arriving from that system. This means movement of the zFS owner. It’s important to understand that the z/OS UNIX ownership is not moved. The I/O happens on the zFS owning system and can move independently of the z/OS UNIX owner for sysplex-aware file system. The z/OS UNIX owning system has no effect on performance for a sysplex-aware file system. To display the current z/OS UNIX owner, use the `df -v` command (as shown in Figure 3). To display the current zFS owner, use the `zfsadm lsaggr` command.
Flexible file systems

The zFS file system support in z/OS V1R11 allows you to specify that all zFS read-write file systems run sysplex-aware. The enhanced support that APAR OA29619 provides gives you the ability to individually choose which zFS read-write file systems are sysplex-aware (FS2 in Figure 4) and which are not (FS1 in Figure 4). This gives you the flexibility to determine which file systems are best to run sysplex-aware.

To be or not to be sysplex-aware

With zFS APAR OA29619, you can give zFS the capability to mount zFS read-write sysplex-aware file systems, but not every file system needs to be sysplex-aware. You enable this function using the sysplex=filesys option in the zFS IOEFSPRM configuration options file. When all systems in your shared file system environment are running z/OS V1R11 and are running zFS with sysplex=filesys, you can specify which zFS read-write file systems should be sysplex-aware by mounting the file systems using the RWSHARE MOUNT PARM.

Summing up: flexibility and control for your file systems

Running zFS with sysplex=filesys on all systems in your shared file system environment is the recommended way to run zFS (even if you don’t want to mount any zFS read-write file systems as sysplex-aware).

For complete details explaining how to use these new options, see the April 2010 version of z/OS V1R11 Distributed File Service zSeries File System Administration, SG24-5989-11.
F
or several releases, zFS is considered the strategic file system over HFS.
To make migration from HFS to zFS easier, we’ve provided a new IBM Health Checker for z/OS check called USS_HFS_DETECTED that identifies the mounted HFS file systems and generates a report. After you receive the report, you can plan how to proceed with the migration. This check is planned to be available with z/OS V1R12 and with V1R11 through the PTF for APAR OA29947.

How it works
The USS_HFS_DETECTED check looks for all mounted HFS file systems and when it finds one, it generates an exception message. The check examines only HFS file systems that are owned on the system running the check. For example, when one or more HFS file systems are found active on the current system, the following exception message displays on the console and the system posts a detailed report in the SDSF.CK log:

You want the check to run every six hours to get a report of every mounted HFS file system. (The default is 24 hours). At the same time, you want to ignore the HFS file system named MY.OWN.HFS. To do this:

1. Create a new POLICY statement in the HZPRMxx member of SYS1.PARMLIB.
2. Specify the member either when first starting the IBM Health Checker for z/OS on the START hzsproc,HZSPRM=xx console command. Or, if the IBM Health Checker for z/OS is already running, use the MODIFY hzsproc,REPLACE,PARMLIB=(xx) console command to update the policy like in this example:

   ADD POLICY STATEMENT(USS_HFS_DETECTED) UPDATE CHECK(IBMUSS,USS_HFS_DETECTED) CHECK(IBMUSS,USS_HFS_DETECTED) SEVERITY(LOW) INTERVAL(06:00) PARM('RUN_ON_MOUNT=YES,HFS_LIST=MY.OWN.HFS') DATE(20100407) REASON('HFS file systems are no longer the strategic file ' 'system. All HFS file systems should be migrated ' 'to zFS')
You can also make the changes directly using the MODIFY console command:

```plaintext
MODIFY hzsproc,UPDATE,CHECK=(IBMUSS,USS_HFS_DETECTED),
   INTERVAL=(06:00),PARM=('RUN_ON_MOUNT=YES,HFS_LIST=MY.OWN.HFS')
```

Every six hours, the following actions take place:

1. The following exception message is displayed on the console screen:

   `BPXH068E One or more HFS file systems mounted.`

2. The SDSF.CK log generates a detailed report as in this example:

   `BPXH069I The following HFS file systems were found:`

   `----------------------------------------`
   `ZOS112.ETC.HFS`
   `ZOS112.MAN.HFS`
   `POSIX.MY.HFS`

   `END TIME: 04/07/2010 14:09:06.347517  STATUS: EXCEPTION-LOW`

Notice that mounted HFS file system MY.OWN.HFS didn’t show up in the BPXH069I report. That’s because you specified HFS_LIST=MY.OWN.HFS.

Also, because you specified RUN_ON_MOUNT=YES, if an HFS file system was mounted at any moment during the six-hour time interval, the check runs and issues the BPXH068E message.

After you do the migration of all HFS file systems to zFS, the above example policy results in the following message in the SDSF.CK log:

`BPXH067I No HFS file systems are mounted.`

`END TIME: 04/07/2010 18:12:36.667641  STATUS: SUCCESSFUL`

**Migrate now; here’s how**

You can use the ISPF-based BPXWH2Z tool to migrate HFS file systems to zFS file systems. For complete details, see z/OS UNIX System Services Planning, GA22-7800.

For more information about the new USS_HFS_DETECTED health check, see z/OS IBM Health Checker for z/OS User’s Guide, SA22-7994.

**Did you know...**

You can access any of the documentation for z/OS products mentioned in this newsletter by visiting the following Web site? You can view PDF or book formats for most publications as well as z/OS LibraryCenters and z/OS platform publications!

[ibm.com/systems/z/os/zos/bkserv](http://ibm.com/systems/z/os/zos/bkserv)
Listen, my sysprogs, and you shall hear
Of the support you’ve wanted for many a year...

It was a fateful night, in Boston 1773, when the colonists could no longer accept the taxing z/OS Language Environment assembler usermods forced upon them by the king of z/OS and the other red ties.

GA of z/OS V1R12 was almost here, and the minutemen sysprogs did not yet understand that they no longer needed to code CEEDOPT, CEECOPT, or CELQDOPT assembly language CSECTs, all used to set Language Environment installation default runtime options. Good news if they but heard it! For this, in turn, meant they would not have to assemble those CSECTs into installable SMP/E usermods.

The code heard ‘round the Language Environment world

We take you now to a small tavern in Concord, as Samuel Adams, a noted programmer and devoted z/OS user, arose, beer in hand, to speak to the gathered minutemen sysprogs. He said, “For many years we suffered with no recourse against programming tyranny and were forced without representation to use the Language Environment usermods to set our installation default runtime options. Five years ago, in z/OS V1R7, to placate our demands, z/OS granted a less taxing method of setting runtime options with a CEEPRMxx parmlib member and the SETCEE operator command.”

“Wait!” said a minuteman from the back of the room, “Two years ago the z/OS crown graciously provided CEEROPT for batch.” This brought a few more cheers. Samuel continued, “But that does not grant us the freedom to fully control CEEPRMxx. Were we not forbidden to indicate that options could not later be overridden by Paul Revere or other minutemen sysprogs, thus prolonging the use of the Language Environment usermods?”

“So tonight I stand before you to announce the king of z/OS has heeded our demands in z/OS V1R12. Now, my friends, we sysprogs can use CEEPRMxx and the SETCEE command to enforce that runtime options may not later be overridden. A new syntax is being provided, which looks similar to the existing assembler syntax.”

At this a distinguished sysprog from Hartford immediately arose and interrupted yelling, “I will not change my CEEPRMxx syntax.” A roar of agreement came from the crowded tavern. “Calm yourself, my good man!” cried Samuel, “Your existing CEEPRMxx will continue to work unchanged. Only those of us needing the new capability will need to use the enhanced syntax, although everyone can if they have a mind to!” Another sysprog shouted, “Can you show us a sample of the new syntax?” Samuel handed out printed parchments of the new syntax that appeared as follows in Figure 1:

Figure 1. Ye new runtime option syntax
Samuel assured the now excited crowd, “The CEEPRM00 sample parmlib member shipped in the CEE.SCEESAMP data set was also updated to support this new syntax. And, if you are unsure that you are already using CEEPRMxx use the Language Environment check of the IBM Health Checker for z/OS: CHECK(IBMCEE,CEE_USING_LE_PARMLIB). It can verify that at least one parmlib member is defined and at least one option group is in use. More I say! It also reports on which option groups are in use.” At that the crowd began to cheer and raised a toast to celebrate this happy news from z/OS.

One clever minuteman urged the crowd to quiet down. “Haven’t you all forgotten that there is a ship in Boston Harbor full of Language Environment usermods that are no longer needed?” Samuel asked this bold man, “How do you suggest we handle these usermods?” The man responded quickly, “Let’s go dump them in the harbor!” The crowd applauded. Try as he might Samuel was unable to regain control of the meeting, and the minutemen rushed out of the tavern, boarded the ship, and dumped every last Language Environment runtime option usermod overboard.

z/OS enthusiast Paul Revere rode through the countryside shouting, “The usermods are gone! The usermods are gone! Why wait to migrate to CEEPRMxx? Take the horse by the reins! Full support is available with z/OS V1R12!”

Proclamation from z/OS!

Afterward, the king of z/OS heard of this and decreed that in the future the capability to change installation runtime option defaults with SMP/E installable usermods was to be removed. Then, CEEPRMxx and SETCEE will be the only usermods to be removed. Then, option defaults with SMP/E installable capability to change installation runtime this and decreed that in the future the option defaults with SMP/E installable capability to change installation runtime this and decreed that in the future the option defaults with SMP/E installable capability to change installation runtime...
Anne Dames is a Senior Software Engineer in Charlotte, NC. She holds degrees in Mathematics and Computer Science. Since 1993, she has worked in development of high-security cryptographic products and is the team leader for the IBM Common Cryptographic Architecture (CCA) team, which develops software, drivers and firmware for the Crypto Express products.

Anuja Deedwaniya is a Senior Technical Staff Member in the z/OS Development lab at IBM Poughkeepsie, NY. She has been with IBM for 26 years and leads the design and delivery of z/OS Simplification and Systems Management initiative.

Susan Z. Demkowicz is an Advisory Software Engineer for z/OS Enabling Technologies at IBM Poughkeepsie, NY. Her expertise in projects on behalf of the simplification initiative includes common event adapter (CEA) and jobs instrumentation. Susan holds an M.S. degree from Boston University.

Donald Durand is a Senior Software Engineer. He has been with IBM for 24 years. He was the lead system programmer at a major insurance company and has worked in various z/OS areas. Currently, he is the team leader for the Runtime Diagnostics project.

Kathleen Fadden is the Integrated Cryptographic Service Facility (ICSF) Level 3 Team Leader. She has been working in ICSF for over 15 years. Previously, she worked on the Resource Access Control Facility (RACF) service team and in various VM test groups.

Tammy Garren is a Staff Software Engineer in z/OS Global Resource Serialization (GRS) development and the GRS Test Team Lead. She holds a B.S. degree in Biology from Russell Sage College and has been with IBM for 28 years.

Joseph Gentile has worked at IBM since 2008, contributing to z/OS development as a Global Resource Serialization (GRS) Function Tester. He holds a B.S. degree in Computer Science from Binghamton University.

Pat Glenski is a Software Designer and Programmer for IBM. He has worked in the area of Unicode and software globalization for over 15 years, has several patents, and has presented at several conferences on the topic of Unicode.

Vladislav Gordon is a Staff Software Engineer at the IBM Lab in Boeblingen. His focus is the z/OS Management Facility (z/OSMF). His previous experience includes various Tivoli System Automation projects.

Saheem Granados CISSP* is an Advisory Software Engineer in IBM Poughkeepsie, NY, and currently a member of the IBM Tivoli Directory Server for z/OS development team. His expertise includes Encryption and Java Security. Saheem is with IBM for 11 years and holds a Masters degree from Binghamton University.

Steven Hart CISSP* is a Staff Software Engineer for z/OS Cryptographic Services in Poughkeepsie, NY. With IBM for seven years, he has worked on the Integrated Cryptographic Services Facility (ICSF), Trusted Key Entry (TKE), Encryption Facility, and Tivoli Key Lifecycle Manager. Currently, he works on ICSF design, development, and service.

Rainer W. Himmelbach graduated with a degree in IT and Electronics Engineering in Berlin, 1988. He joined IBM in 1991 and currently serves as CVT/SVT test team lead in Resource Measurement Facility (RMF). He is the customer-oriented validation team leader since 2007 and has responsibilities for z/OS and zLinux certifications and IBM System z Business Continuity Solution for SAP.

Nicholas Jones has been a Developer, Function Tester, and Service Leader for system logger over the last seven years. Nick holds a B.S. in Computer Science from Union College.

Shigeki Kimura is the Technical Lead for z/OS migration at IBM Japan. He has participated in key reviews of z/OS Migration, GA22-7499, since 2006. Shigeki has been with IBM for 24 years and was approved as an International Council on Networks (ICN) author this year.

Rohan Kurane is a Software Engineer at Silicon Valley Lab with over two years of experience in the field of DFSMS. Rohan specializes in Integrated Catalog Facility (ICF). Rohan has a Master of Computer Science Degree from Northern Illinois University.

Larry Law is a Senior Software Developer with expertise in VSAM and VSAM RLS. Larry has worked for IBM for 28 years and is the lead developer for the CA Reclaim enhancement discussed in this issue.

Scott Marcotte is the zFS Designer and Lead Developer for the zFS team and works in DFS/MVS Development in Poughkeepsie, NY.

Terri A. Menendez is a Senior Technical Staff Member (STSM) and chief architect for DFSMS with expertise in VSAM, VSAM RLS, and Catalog components. She has been with IBM for 27 years most of which was dedicated to the development of VSAM RLS and is the chief architect for the CA Reclaim enhancement discussed in this issue.

John Monti is a Senior Software Engineer in the z/OS Language Environment development group and is a frequent speaker at SHARE.

Barbara Neumann is an Advisory Software Engineer and a Managing Editor of z/OS Hot Topics. She is the team leader for z/OS Language Environment information development. She holds an M.B.A. degree from the University of Rochester.

Mary Ng is a member of the zFS development team and works in z/OS Distributed File Service Development in San Francisco, CA.

Al J Noll is a Software Engineer for zSHIP (System House Installation and Packaging) at IBM Poughkeepsie, NY. He graduated from Northeastern University in 1984 and has held various positions in security, information development, tools development, and most recently, software function test.

Aspen Payton is a Staff Software Engineer for Predictive Failure Analysis in Rochester, MN, with a degree from the University of Alaska at Anchorage. With IBM for more than ten years, she has worked on performance-related software including System i Collection Services, Job Watcher, and Disk Watcher. She has published numerous articles and holds patents on autonomic processing and enhanced flight data recording currently serving on the IBM Invention Review Board.

Bob Petti is the Test Team Leader in the z/OS Security area. He has worked on numerous security and middleware components, including Integrated Cryptographic Services Facility (ICSF), Lightweight Directory Access Protocol (LDAP), Secure Socket Layer (SSL), and zFS.
William A. Ruby is an Advisory Software Engineer in the Common Event Adapter (CEA) development team in IBM Poughkeepsie, NY.

Jim Showalter is the zFS Development Team Leader and works in z/OS Distributed File Service Development in Endicott, NY.

Andrew Sica has worked on z/OS system logger in various roles for over ten years at IBM. Andrew has co-authored an IBM Redbooks document on system logger and holds a B.S. in Computer Science from Western New England College.

Viola Thierfelder joined the z/OS HCD/HCM development team in 2008. Before that she worked for 13 years with IBM performance management for the Resource Measurement Facility (RMF) development team.

Mario Verna is the z/OS Print Software Product Manager responsible for defining the strategy for PSF, Infoprint Server, and Transforms. He has been involved in IBM JES3 Design and Development, Release Management, People Management, and divestiture of the IBM Printing Systems

Marna Walle is a Senior Technical Staff Member in the z/OS Build and Installation group at IBM Poughkeepsie, NY. Her current responsibilities include z/OS release migration and new installation technologies. Marna is a frequent speaker at SHARE.

Martina Weidler is member of the z/OS hardware configuration definition/hardware configuration management (HCD/HCM) development team for almost 16 years and works in the Boeblingen Lab.

Clarifications, corrections, and announcements

Correction to our contributors for z/OS Hot Topics, Issue 22 (March 2010):

Holger Scheller is a Senior Software Developer at IBM Boeblingen Lab since 1996. His expertise includes object-oriented (OO) design and development (C/C++,Java), application development for DB2 on z/OS and z/OS UNIX System Services. He is a global expert for SAP on System z.

Visit the Hot Topics Web site for clarifications and corrections to articles from our past issues.

ibm.com/systems/z/OS/zos/bkserv/hot_topics.html

Greg Walters is a User Experience professional in the z/OS Development lab at IBM Poughkeepsie, NY. He has been with IBM for 33 years and performed a variety of roles. His current assignment includes the simplification of tasks performed by z/OS system programmers.

Marianna Wenske is a Software Engineer at the IBM Boeblingen Lab. Since joining IBM in 2006, she has worked on Dynamic Infrastructure* and Workload Management (WLM).

Dave Zingaretti is a Software Developer in z/OS development. He has been with IBM for 29 years. His previous focus areas were z/OS UNIX System Services and the common event adapter (CEA) components.

* Certified Information Systems Security Professional
RLS striping support provides compatibility between VSAM and RLS applications using striped data sets. In z/OS V1R12, RLS removes all VSAM striping restrictions for VSAM stripe data sets. Note that, as always, VSAM striping rules still apply to RLS support.

Some definitions:

*Striping data* means that the tracks, for the sequential access method (SAM), or the control intervals (CIs) for VSAM, are spread across multiple devices. This format allows a single application request for records on multiple tracks or CIs to be satisfied by concurrent I/O requests to multiple volumes.

*A striped VSAM data set* is defined as having a stripe count greater than one. Any data set listed with a stripe count equal to one is considered to be in the extended format but is also considered as striped.

For more info on VSAM striping see:

“Chapter 6. Organizing VSAM Data Sets” in DFSMS Using Data Sets, SC24-7410