Are you ready?

z/OS V2R1, zBC12, zAware, zEDC, z/OSMF, Flash, Security, and much, much more!

Jeff Magdall
z/OS Project-Development Team (PDT) Leader
On our cover, Jeff Magdall, who presents the z/OS® trends and directions during the MVS project kickoff at SHARE, stands ready to roll out lots of new z/OS goodies. In this issue, we'll look at all the new features and functions brought to you by z/OS V2R1 and the IBM® zEnterprise® BC12 (zBC12) hardware, along with a sprinkling of the great things we did between the z/OS V1R13 and z/OS V2R1 releases. Look for articles about IBM z/OS Management Facility (z/OSMF), IBM System z® Advanced Workload Analysis Reporter (IBM zAware), z/OS Distributed File Service zSeries® File System (zFS), System z Test Report, and all of the other Zs. It is amaZing to see how much we’ve stuffed into this one issue of Hot Topics! For Jeff's overview, see our cover article “Are you ready? z/OS V2R1”.

But you can read the issue to hear about all that. Right now, we want to say goodbye to our fearless Hot Topics leader, Jim Guilianelli, who’s retiring. This is his last issue, and we'll miss him tremendously. Jim has an amazing, breezy style as executive editor, humorous yet with an iron grip on a huge array of details, from our suspenseful procurement and delivery process and ever-shifting art budget, to his staff of very quirky editors (present company excluded, naturally). All of our Wittiest headlines and most alluring graphics are Jim's idea, and he has wheeled many a cover model through the tricky process of being photographed. (Our technical and executive superstars, deeply preoccupied with ensuring that z/OS is at its very best, haven't been hardened to the attentions of the paparazzi.)

We're going to miss Jim very much, but say hello to our new executive editor, Jodi Everdon, whose no-nonsense alpha leadership style will send us flying into the new era. You'll never find a more enthusiastic booster of the z mainframe business than Jodi and she won't steer you wrong. Welcome, Jodi!

As for the rest of us, we're hoping you find this issue of Hot Topics entertaining and informative. We always love your comments and look forward to hearing from you.

The Editors
newsletr@us.ibm.com
## Contents: Featured articles

<table>
<thead>
<tr>
<th>Page</th>
<th>Title</th>
<th>Summary</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Are you ready?</td>
<td>z/OS V2R1</td>
<td>Jeff Magdall</td>
</tr>
<tr>
<td>7</td>
<td>It’s a bird. It’s a plane. It’s the zBC12!</td>
<td></td>
<td>Bargav Balakrishnan</td>
</tr>
<tr>
<td>9</td>
<td>A great new look and so much more</td>
<td>The new and improved z/OSMF V2R1!</td>
<td>Anuja Deedwania</td>
</tr>
<tr>
<td>12</td>
<td>Perfectly integrated</td>
<td>z/OSMF Performance Monitoring and Management</td>
<td>Jürgen Baumann and Stefan Wirag</td>
</tr>
<tr>
<td>14</td>
<td>Up in a Flash</td>
<td>Your system bounces back faster with Flash Express</td>
<td>Elpida Tzortzatos, Charles Mari, and Steve Partlow</td>
</tr>
<tr>
<td>17</td>
<td>z Enterprise Data Compression Express</td>
<td></td>
<td>Anthony Sofia, Ceci Carranza Lewis, and Nicholas Jones</td>
</tr>
<tr>
<td>20</td>
<td>THINK: Smart Card Support Element for zFlash</td>
<td></td>
<td>Ed Chencinski, Jim Coon, and Dick Gusefski</td>
</tr>
<tr>
<td>21</td>
<td>Master and coprocessor Enhanced secure key support</td>
<td></td>
<td>Steven Hart, Bob Petti, and James Sweeney</td>
</tr>
<tr>
<td>24</td>
<td>Get hip to the latest cryptography for SSL</td>
<td></td>
<td>Jonathan Cottrell and Mary Rezeski</td>
</tr>
<tr>
<td>28</td>
<td>Are you aware how IBM zAware can improve z/OS availability?</td>
<td></td>
<td>Garth Godfrey, Jim Labarge, and Chris Broker</td>
</tr>
<tr>
<td>27</td>
<td>Use zFS extended directory support</td>
<td></td>
<td>Ann Totten</td>
</tr>
<tr>
<td>30</td>
<td>Managing storage tiers</td>
<td></td>
<td>Glenn Wilcock and Sean McMillen</td>
</tr>
<tr>
<td>33</td>
<td>Unclog your systems with record level sharing for catalogs</td>
<td></td>
<td>Terri Menendez and Jason Lee</td>
</tr>
<tr>
<td>35</td>
<td>Ask Mr. Catalog</td>
<td></td>
<td>Stephen Branch, Eric Harris, and Ryan Wisnewski</td>
</tr>
<tr>
<td>39</td>
<td>Getting a performance kick with large pages</td>
<td></td>
<td>Alfred F. Foster, David Hom, and Scott Tuttle</td>
</tr>
<tr>
<td>46</td>
<td>Are you ready to migrate to z/OS V2R1?</td>
<td></td>
<td>Shigeaki Kihara</td>
</tr>
<tr>
<td>49</td>
<td>Get ready for your next move: Migrating to z/OS V2R1!</td>
<td></td>
<td>Marna Walle</td>
</tr>
<tr>
<td>52</td>
<td>Set up high availability when failure is not an option</td>
<td></td>
<td>Mark Hodsman, Peter Mailand, and Daniel Martin</td>
</tr>
<tr>
<td>54</td>
<td>A point of contention</td>
<td></td>
<td>Nat Stevenson, Joseph Gentile, and Bryan Childs</td>
</tr>
<tr>
<td>56</td>
<td>The clock is ticking!</td>
<td></td>
<td>David Whitney and Rita Beisel</td>
</tr>
<tr>
<td>57</td>
<td>One format fits all</td>
<td></td>
<td>Harold Bender, Karin Genther, and Peter Mailand</td>
</tr>
<tr>
<td>59</td>
<td>Upgrade your batch processing with these JCL enhancements in z/OS V2R1</td>
<td></td>
<td>Girija Varanasi, W. Kevin Kelley, and Steven Jones</td>
</tr>
<tr>
<td>63</td>
<td>Quick and Easy: BCPii!</td>
<td></td>
<td>Steve Warren and Rita Beisel</td>
</tr>
<tr>
<td>66</td>
<td>Your next stop: the HEAPZONE!</td>
<td></td>
<td>Marie Laser and John Monti</td>
</tr>
<tr>
<td>68</td>
<td>Create your own mainframe application development sandbox using RD&amp;T</td>
<td></td>
<td>Reginaldo Barosa</td>
</tr>
<tr>
<td>71</td>
<td>Training budgets got you down? Need a skills plan?</td>
<td></td>
<td>Don Resnik, Kathy Pfieffer, and Howard Baker</td>
</tr>
<tr>
<td>72</td>
<td>Making the switch</td>
<td></td>
<td>Stephen M. Kocik and Dale Riedy</td>
</tr>
<tr>
<td>73</td>
<td>Our contributors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>Glossary for network administrators</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In April 2012, we began telling you about z/OS Version 2 Release 1 (V2R1) and a new two-year release cycle with five years of support. We shifted to a two-year cycle to address an important issue we were hearing directly from you. Many of you said that you had trouble with the pace of our annual releases, and that meant migration was possible only every two to four years. You told us that the work associated with maintaining z/OS support currency was using valuable resources; the same resources that you might otherwise use to implement new functions or to help drive new applications to support business growth.

Naturally, you had questions:
What did the change imply about capability?
How much content would be in the new release?

• What did the change mean to coexistence?
• Will new functions ship in a timely manner?

Actions speak louder than words
Let’s look at some of the content in z/OS V2R1 and some of the things we did in between releases V1R13 and V2R1. The story is not complete without also looking at some of the hardware function made possible by z/OS and other parts of the z/OS environment.

In August 2012, we introduced you to the IBM zEnterprise® EC12 (zEC12). When you look at how you can use zEC12 capabilities, you quickly realize all the z/OS functions that shipped too. (Be sure to check out the latest zEnterprise hardware activity in “It’s a bird. It’s a plane. It’s the zBC12!” on page 7.)

zAware, Flash Express, cryptographic support, and other enhancements
These enhanced functions include enabling support for IBM zAware, which uses analytics technology from IBM’s Research division to look for patterns in your system that can help improve problem determination. The zAware technology was not possible or necessary when an experienced system operator was able to recognize message patterns by watching the console. Today’s message rates are much faster than human capability.

Along with the real storage manager (RSM) Enablement Offering, came Flash Express. Use Flash Express technology to improve performance by reducing any paging delays that might occur at the beginning of a shift or during other transitional periods.

Transactional execution, long, a computer science principle, can be used to improve performance and reduce execution time by eliminating software locking that increases processor usage. At the same time, there were updates for Cryptographic Support, High Performance FICON™, networking, and security standards.

zEnterprise is ready
Today, with the availability of the zEnterprise BC12 (zBC12), we extend many of the same functions and features to many of you who need less capacity. In February 2013, we announced a significant set of solutions that support Cloud, Data, and Security initiatives on IBM System z®.

Cloud ready
IBM Intelligent Operations Center for Smarter Cities® uses analytics to help cities anticipate problems and formulate solutions they can use to prevent them. These analytics can help accelerate responses (like those to super storm Sandy), improve inter-agency coordination, and provide near real-time visibility of key data to drive better decision-making.
Data ready
A new hardware option, IBM DB2® Analytics Accelerator for z/OS Version 2.1, provides up to three times faster hardware with 50% more capacity to help accelerate queries directly from the mainframe. IBM Smarter Infrastructure for Social Services (using Curam on zEnterprise) can accelerate the deployment of new social program policies and provide superior protection of personal data. IBM Smarter Analytics Anti-Fraud Infrastructure for zEnterprise can help reduce cost and time to value for integrating real-time, anti-fraud analytics into operational payment systems. IBM Enterprise Asset Management (Maximo) for Government can help you inventory, track, and manage assets. In addition, IBM zEnterprise Smarter Analytics™ for Retail can help you capitalize on point-of-sale and other opportunities to drive greater sales.

Security ready
IBM Enterprise Key Management Foundation solution enhances security by offering a centralized key management to reduce complexity and improve compliance to industry regulations. With these and other solutions, we now have a portfolio across the Finance, Insurance, Public, and Healthcare industries to help you succeed—and we have introduced several or them over the past 18 months.

z/OS V2R1 delivering value and innovation
That brings us to z/OS V2R1 and z/OSMF V2R1, which together offer significant new functions. The Smarter Planet® solutions and Cloud Ready, Data Ready, and Security Ready focus across our offerings, you can find more of the dedicated value the mainframe brings to your enterprise.

Cloud. All ready!
We know that you know System z was a cloud before cloud was a term. What makes z/OS stand apart, beyond multitenancy, is the ability to manage multiple, disparate workloads together. We bring decades of refinements that provide unprecedented scaling, availability, and virtualization in support of business critical and cloud workloads.

z/OS V2R1 supports:
- 100-way multiprocessing in a single LPAR
- 2 GB pages to reduce memory management processor usage and drive performance
- Expanded support for Transactional Execution in production environments, to encompass C/C++ and High Level Assembler applications in addition to Java.

Data. All ready!
z/OS on System z continues as the premier operating system for data storage, management, availability, and integrity and ready to meet those challenges.

Managing storage tiers: New storage tiers capability in DFSMShsm help you optimize policy-based data placement to match business need with data accessibility and retrieval times to meet your goals while controlling storage cost.

Record Level Sharing for catalogs:
Recent improvements in ICF catalog processing and scalability are crowned with record-level sharing access for user catalogs to help you simplify your catalog environments while improving performance.

zFS Extended Directory: A new zFS design is expected to provide dramatic real-world performance improvements for file systems with large directories, benefiting large SAP systems and other environments using large directories.

FICON support: Improved support for FICON channels and switches helps direct bandwidth where it’s needed while improving usability and reducing opportunities for error that can affect availability.

Batch support: We continue to help you shrink, or perhaps, in some cases even close, your batch windows. Parallelism in DFSMShsm enables recalls of migrated data sets in batch, improved data set sharing capabilities to make data available for processing more quickly and reduce how long jobs must wait. Improved generation data group processing can eliminate batch job steps and reduce processor usage. A number of other batch-oriented enhancements improve JCL usability (and reusability) and programmer productivity.

zEnterprise Data Compression (zEDC): On-board data compression with the zEDC feature available for zEC12 and zBC12 servers and the z/OS zEDC feature enables data compression for SMF data, BSAM and QSAM data sets, and Java. This can improve performance for existing compressed data. Yet, more importantly, it offers an opportunity to compress data to achieve storage savings or help improve data access time by keeping more data in the primary level of the storage hierarchy.

Security. All ready!
System z offers the ultimate in Security. z/OS V2R1 provides strengthened auditing and reporting for digital certificates and improved security-related health checks. There is support for more industry and government standards, including NSA Suite B, TLS 1.2, and enhanced certificate validation support, and improved Enterprise PKCS #11 processing. We also extend access to world-class System z cryptography, with crypto-as-a-service from Linux clients to enable encryption in a more secure environment that does not show the keys in memory.
**z/OSMF evolution**

z/OSMF continues to evolve in z/OSMF V2R1, which forms the foundation for z/OS simplification. The new IBM WebSphere Application Server Liberty profile simplifies setup while providing significant reductions in the resource requirements for z/OSMF. Helping you manage your z/OS Software inventory are new reporting and display capabilities, including a function to retrieve and display end of service information about installed products. Another provides cross-validation of SMP/E inventory, catalog, and volume information. New reporting functions are available to help you with service level (PTF) management, and to display the location and content of software instances created using the Software Deployment application.

A new z/OSMF Workflow Application is designed to enable setup and configuration assistance to simplify z/OS configuration. This application will route tasks to people in defined roles such as “system programmer” and “security administrator”, to complete setup and configuration tasks in the right order with wizard-like guidance.

**Ready for the future**

Outlined here are only some of the new capabilities in z/OS V2R1. Our challenge is to give you the tools you need for your business so that you can use every bit of it. As we approach the mainframe’s 50th birthday, we continue to provide the highest value content and innovation that you expect in z/OS and System z.

**More**

Be sure to check out the articles in this issue that address many of the functions and topics described in this article like zAware, z/OSMF, zEDC, zFlash and many more ... We are ready to help serve you!

---

**System z Platform Test Report for z/OS and Linux Virtual Servers**

A system programmer’s best friend

IBM System z Platform Evaluation Test is a team for all seasons — and for all releases of z/OS and System z hardware and System Storage®! We are a team of system programmers and testers that run a Parallel Sysplex on which we perform the final verification of a z/OS release and System z hardware and System Storage before they become generally available to clients. We gather our experiences, hints, tips, and recommendations and publish them in a report. We publish our report with each new release of z/OS and provide a refreshed edition in between releases.

Recently we released our December 2012 test report that is our latest refreshed edition. Highlights included in this edition are our experiences testing the new IBM zEnterprise EC12 server along with our implementation of the IBM System z Advanced Workload Analysis Reporter (IBM zAware), our implementation of the Flash Express feature and our z/OS exploitation of Flash storage.

To find all our test reports, as well as our latest tips and experiences, check out our website: [ibm.com/servers/eserver/zseries/zos/integtst/]
Organizations around the world are recognizing the increasingly crucial role that technology plays in driving change as they shift investments from infrastructure maintenance toward such new projects as cloud computing, analytics, and mobile applications. In order to capitalize on these emerging opportunities, organizations must respond with increased agility to deliver new services while addressing cost, complexity, and risk. This requires an optimized infrastructure that is integrated, agile, and secure.

**Introducing the IBM zEnterprise BC12**

IBM launched the IBM zEnterprise EC12 (zEC12) in 2012 and is now extending many of the same capabilities of this modern IBM mainframe to enterprises of all sizes with the introduction of the IBM zEnterprise BC12 (zBC12).

The zBC12 supports features and functions such as

- Flash Express, enabling the use of pageable 1 MB pages
- IBM zEnterprise Data Compression (zEDC)
- Shared Memory Communications through Remote Direct Memory Access (SMC-R)
- Improved processor facilities including transactional execution and enhanced out-of-order instruction processing;
- CryptoExpress4S
- CFCC Level 18 and 19 enhancements.

Like the zEC12, the zBC12, combined with the IBM zEnterprise BladeCenter® Extension Model 003 and IBM zEnterprise Unified Resource Manager, also offers a proven hybrid computing design that can help you manage and integrate workloads on multiple architectures within a single system.

**Unprecedented performance, flexibility, and scalability at a lower cost**

The zBC12 is powered by up to 18 microprocessors, running at 4.2 gigahertz (GHz), providing up to a 33% boost in per core capacity and up to a 50% increase in total system capacity for z/OS compared to its predecessor, the z114.

The zBC12 is available in two models;

- A single central processor complex (CPC) drawer model, the H06
- A CPC two-drawer model, the H13, which offers additional flexibility for I/O and coupling expansion, specialty engine scalability, and memory scalability if desired, up to 496 gigabytes (GB).

The zBC12 boasts up to 13 configurable cores that can be configured as general-purpose processors or as a variety of specialty engines. zBC12 offers up to 156 capacity settings, providing the freedom to choose the right capacity setting for your needs and the flexibility to scale on demand as workload demands increase.
An agile cloud solution
Unlike other proclaimed cloud solutions that are defined by a siloed architecture resource pool, the zBC12 leaps beyond this approach by including heterogeneous computing resources in the pool that can be fully optimized, quickly deployed, and centrally managed to meet ever-changing business requirements. And with z/OS, you have the ability to run multiple disparate workloads concurrently with different service levels that z/OS will work to deliver seamlessly.

Data ready
Two new z/OS V2R1 functions, IBM zEnterprise Data Compression (zEDC) and SMC-R, provide data performance benefits. These functions are supported on the zBC12 with new I/O adapter cards. zEDC provides compression acceleration to save storage and improve wall-clock time, while SMC-R optimizes server-to-server communication with reduced latency and lower processor overhead than traditional TCP/IP communication. The zBC12 includes an optional analytics availability offering, IBM zAware, a self-learning firmware-based solution to help organizations quickly identify anomalies using z/OS message logs. Data analytics solutions on the zBC12 include the IBM Smart Analytics System and the IBM DB2 Analytics Accelerator, which are designed to enable you to efficiently store, manage, retrieve, and analyze vast amounts of data for business insight.

Ultimate security
The zBC12 is designed to provide the highest levels of security with Common Criteria Evaluation Assurance Level (EAL) 5+ certification for security of logical partitions on the processor core. Crypto Express4S, the newest generation cryptographic feature, provides digital signature algorithms for use with smart cards and enhancements to IBM’s Europay, MasterCard, and Visa (EMV) support.

Summary
The IBM zEnterprise BC12 allows enterprises of all sizes to leverage IBM mainframe capabilities to embrace such new opportunities as cloud computing, analytics, and mobile computing in order to grow their businesses based on the foundational strengths of System z.
The z/OSMF V2R1 introduces many new capabilities and tasks for the system programmer, besides enhancing existing function. But there’s even more! z/OSMF V2R1 also delivers on an updated and much improved packaging and configuration. Here’s a summary:

**A new slimmer “look”**

In V2R1, z/OSMF delivers a totally updated product package that not only includes all the interim updates in it’s base but also improves on the overall client experience. This new version is rebased on the Websphere Application Server for z/OS V8R5 Liberty profile and includes only the features of Websphere needed for z/OSMF, making the package much lighter. For example, the Websphere application server for z/OS OEM edition (WASOEM) 7.0 is no longer needed and so is not included in the product package.

This means the full product package size is much smaller. It’s approximately 350+ MB rather than the 1+ GB it was before. The Liberty profile, which was a separate installable entity, is now part of the z/OSMF application. This reduces the configuration process because WASOEM is not necessary.

**Fewer steps for configuration**

Also in this release, the z/OSMF overall configuration itself has been simplified to require fewer steps and fewer configuration variables for z/OSMF. There is no separate deployment of applications required. Basically you specify the configuration variables, perform the resulting required security setup, complete the configuration, then start the application. That’s it!

z/OSMF V2R1 requires SAF mode authorization, and ZMFAPLA class to be enabled. Repository mode authorization is no longer supported, but not to worry, as scripts are provided to help with the migration, and so far it has been a smooth experience for clients who have tried it in z/OSMF V1R13. If you have hardware crypto, the support is built in, and no additional steps are required. In addition, the setup of the necessary certificates and keyring for secure connection from your browser and the z/OSMF server is included, and these can create the certificate authority and server self signed certificate if needed.

**Less is more: reduced start time and simplified service**

Using the the WebSphere Liberty profile, z/OSMF starts and stops much more quickly, taking just a few seconds instead of a minute plus, depending on system resources available. Even the available memory requirement has been reduced from 2 GB to 1 GB and requires about half the MIPS than was previously needed for most of the scenarios. Based on this change, clients with systems having fewer resources can now use z/OSMF easily after an easier configuration process.

The z/OSMF service process is also simpler now, to align with the normal z/OS service approach. Now, with V2R1, you can usually just apply a PTF and restart the application to activate it. Because there is no separate runtime (WASOEM), there is no requirement for the extra step of deploying the application. It’s that simple!
Working it: Workflow feature

z/OSMF 2.1 introduces the first stage of a brand new feature called Workflow. The workflow framework is part of base z/OSMF or z/OSMF core, so it is always available. This function provides a framework to help simplify configuration on z/OS by bringing together the end-end steps for configuration of selected software or features. Think of this as a means of providing a programmatic to-do list that shortens time to value. The workflow framework is not confined to configuration, and generic enough to make it easy to perform lots of tasks on z/OS.

Using this feature, a product owner or end user can create workflows defining the steps required to perform a z/OS task using the XML schema for the z/OSMF workflow metadata file. You can define manual steps, use a JCL (job) wizard, or specify that a REXX exec or shell script run.

Workflow includes concepts of step owner, assignment of steps, accepting and performing steps, marking steps complete, and overriding or skipping steps. When you assign a step to a user or a role, the user or set of users in a role are notified by the Notification task, also new in z/OSMF V2R1. When a user logs in to z/OSMF, any notifications pending for that user are presented in the Notification task.

The workflow function maintains a history of all actions performed on a workflow and allows you to store notes. We provide an initial workflow with V2R1 to documents the steps required to configure the pre-requisites for the various z/OSMF plug ins. We will exploit this workflow feature for z/OSMF configuration beyond the initial core configuration over time. Watch this space for future enhancements!

Enhancements galore

Since the availability of z/OSMF V1R13 in September 2011, we’ve made many enhancements through service updates. A few highlights of these enhancements in z/OSMF v1R13 are listed below:

Better application linking for monitoring and workload

The z/OSMF Resource Monitoring and z/OSMF Workload Management applications include improved application linking. For example, the System Status task is linked to the Workflow Management task, allowing you to open and view the active service definition, active service policy, or WLM status. While viewing the active service definition or service policy in the Workflow Management task, you can also see and open resource monitoring dashboards with performance metrics for service classes, workloads, and report classes. This function is available on z/OSMF V1R13 with PTF UK93933/APAR PM74508 and PTF UK83836/APAR PM74517.

Enhancements to REST JOBS Submit API

We’ve enhanced the REST JOBS Submit API to allow you to:

- Submit jobs from a z/OS UNIX file or data set, (including requesting recall of a migrated data set).
- Access the job control language (JCL) secondary JES2 subsystem to cancel a job, change job class, or delete a job (cancel a job and purge its output).
- Deprecate non X-IBM—prefixed custom headers on z/OSMF V1R13 with APAR PM74502.

z/OS Jobs REST Interface and JES2

In z/OS V2R1 with z/OSMF V2R1, we extended the z/OS Jobs REST Interface to make it easier to reuse existing JCL and detect job completion.
• Add support for optional asynchronous notification upon job completion.
• Allow passing of JCL symbols to a job.
• Use an optional job correlator unique across the JES2 spool.

Software deployment improvements
The z/OSMF Software Management application (previously called Software Deployment) extends the Software Deployment task. In addition to deploying software, it now provides additional actions on instances of SMP/E installed software to help you manage your system software more easily. This enhancement is available on z/OS V1R13 with the PTF UK83841 for APAR PM73833.

The Software Management task now provides a view of the product, feature and FMID content in the software instance, as well as the physical data sets that compose the software instance.

You can identify software products that are approaching, or have reached, end of service support, thus helping your customers with upgrade and migration planning. In addition you can validate that the SMP/E structure and content of a software instance is correct through the cross-check SMP/E inventory information about catalog entries, volume residency, and data set content. It also supports actions to analyze and report on software instances and products within instances to do the following:

• Identify missing HIPER and PE fixes, and fixes associated with one or more fix categories to help customers assess the risks and stability of installed software and ensure hardware and software requisites are installed.
• Determine if individual fixes are installed and in which software instances.

Finally it can compare the service and functional content of two software instances to aid in debugging or migration planning. These new functions are designed to help you manage your system software more easily.

For complete details, visit the z/OSMF web site:
ibm.com/systems/z/os/zos/zosmf/

New look for Configuration Assistant for Communication
In z/OSMF V2R1, the Configuration Assistant for Communication server has a new look and feel more aligned with the other z/OSMF plugins. It also supports the new Communication server Policy-Based Networking functions with AT-TLS support of TLS V1.2 and Policy-based Routing (PBR) support for IPv6.

Improvements to Performance task for Capacity Provisioning
The Performance task for Capacity Provisioning keeps on improving! In V1R13, we've enhanced it to allow you to create, edit, and activate domain configurations and capacity provisioning policies. (See PTF UK83852 for APAR PM74519.)

In V2R1 Capacity Provisioning features the introduction of the Defined Capacity and Group Capacity management capabilities, which broaden the range of environments to which Capacity Provisioning is applicable. The Capacity Provisioning task now lets you create policies to increase and decrease Defined Capacity and Group Capacity based on your schedules or the workload running on your system. You can also define additional scopes and limits to reflect your business needs for additional Defined Capacity or Group Capacity.

This new capability is seamlessly integrated into the z/OSMF Capacity Provisioning task, so you can install and activate your policies on the Capacity Provisioning Manager, monitor the status of your Capacity Provisioning Manager and display reports about your current configuration and active policy.

This integration allows you to either continue using your proven and tested policies or to extend them with the new capabilities and take additional advantage of them. Capacity Provisioning encourages the deployment of z/OSMF as the tool of choice to create and maintain Capacity Provisioning policies and domain configurations.

So check out the new look and improved function of z/OSMF V2R1. You'll be glad you did.

Acknowledgments
Thanks to Adrian Alvarez Diez, a software engineer at IBM Germany, for his contribution to the Capacity Provisioning section.
Wouldn’t it be nice to check the IBM z/OS Workload Manager (WLM) service definition quickly when you notice that workload performance is suffering? Wouldn’t you like to view the performance results of the service definition changes that you made? Now with z/OSMF, you can do all this with one mouse click!

To deliver these enhancements, z/OSMF released a V1R13 SPE that links the following existing tasks to each other in context:

- The z/OSMF Workload Management task, which allows you to define and manage your WLM service definitions, including installation and activation
- The z/OSMF System Status task, which provides a quick performance status overview of all the sysplexes in your data center.
- The z/OSMF Resource Monitoring task, which allows you to define monitoring dashboards that show performance results in detailed bar charts.

**Linking Workload Management and Resource Monitoring tasks**

By linking the Workload Management task and the Resource Monitoring task, you can open dashboards with performance metrics for service classes, workloads, and report class items, you can open the corresponding information in the WLM service definition.

**Linking Workload Management and System Status tasks**

By linking the Workload Management task and the System Status task, you can open the System Status task from the Workload Management task and view the status of the sysplex. And, you can open the Workload Management task from the System Status task and view the active service definition, active service policy, or WLM status.

If there is not a tab for the z/OSMF target task already open, application linking opens the target task in a new z/OSMF internal tab. If the z/OSMF target task tab is already open, application linking brings the existing tab in the foreground and adjusts the content of the tab according to the linking context. The z/OSMF internal tab of the task triggering application linking stays open but moves to the background.

**How it works — some examples**

Suppose that you are in the Workload Management task and you want to see how your sysplex performs overall. To get the answer, simply click the View performance of systems link in the WLM Status tab to open the System Status task, which shows you a green-yellow-red indication of your sysplex performance.
Now, let’s say that from the System Status task, you notice that your sysplex runs with a poor performance index. To get more information about it, just click the performance index status text in the table. z/OSMF opens the Resource Monitoring task with a dashboard that shows you the performance indexes that can help you determine which service class is in trouble.

And if you want to check the service class goal in the WLM service definition, just follow these steps:

1. Click the icon beside the service class name in the bar chart.
2. Click View WLM Service Class from the context menu (see Figure 1, item A).

z/OSMF switches to the Workload Management task, opens the active service definition in the View tab, and displays the Service Classes table with the service class period focused.

Now, use the Switch To > Editable Version of Service Definition menu item to switch from View mode to Modify mode and make the required changes. Then, you can reinstall and activate the service definition from within the Modify tab by using the menu entry Activate from the Actions menu of the Service Policies table.

To view the sysplex performance after the service class change, follow these steps:

1. Select the service class in the Service Classes table in the Modify tab.
2. Select View Performance of Selected (see Figure 1, item B).

This action opens the Resource Monitoring view and opens a dashboard that displays performance information of the service class.

If your updates did not improve the sysplex performance as you expected, you can continue to optimize the service definition by switching back and forth between the z/OSMF tasks.

As you can see in our scenarios, these new links makes some navigation and filtering steps obsolete and simplifies your work.

To use this feature, ensure that you have the PTFs for the following APARs installed:

- PM74502 (z/OSMF)
- PM74508 (Resource Monitoring)
- PM74517 (WLM)
- PM74925 (WLM)
Now, for the first time, flash storage is integrated into System z servers with the optional Flash Express cards on the IBM zEnterprise EC12 (zEC12). These PCI Express (PCIe) attached RAID 10 cards plug in as cable-connected pairs in the I/O expansion drawer, which enables some great functions and benefits now, and lays a foundation for further enhancements for future generations of System z.

Initially, the main application of Flash Express is as an extension to main memory in z/OS, where it’s integrated within the memory hierarchy and managed by the z/OS memory manager to provide increased system availability and resiliency during paging-intensive events.

Using Flash for auxiliary storage
The IBM z/OS Real Storage Manager (RSM) and Auxiliary Storage Manager (ASM) use Flash Express as another form of auxiliary storage. Auxiliary storage is used when pages of data must be moved out of real storage because of real storage usage requirements or when applications or components explicitly request to move some pages out of real storage. Before Flash Express was available, DASD was the only storage medium available for paging data to auxiliary storage. You use Flash Express with traditional paging data sets to form a pool of auxiliary storage. In this way, RSM and ASM can determine where to write a page of data (DASD or Flash) based on factors such as space availability, and the characteristics of the data being written.

Flash Express also offers reduced data movement times to-and-from real storage, which results in higher paging bandwidth. Where Flash Express really outshines DASD is in moving bulk data in a small window of time. z/OS exploits this advantage by batching I/O requests as much as possible, presenting them as a group to the hardware, and using 1 MB pageable pages.

Benefits of Flash Express
Installations are being pressured to deliver ever higher levels of availability in finance, banking, healthcare — just about every industry is driven by high service levels. The following examples show how Flash Express on IBM System z (which is used by z/OS), provides additional system availability and resilience transparent to applications:

• With Flash Express, your system can reduce workload and response time delays that are caused by increased memory pressure during periods of peak demand, transitions between workloads, or during diagnostic collection.

For example, when a workload shifts from a transactional workload during a prime shift to a batch workload, and then again back to a transactional workload during the next prime shift, response time delays can occur. You can reduce these delays by using Flash Express when data for the prime shift must be transferred from auxiliary storage into main memory.

Up in a Flash
Your system bounces back faster with Flash Express
Flash can also help when you must transfer data into main memory as part of a diagnostic dump. The increased I/O bandwidth you get with Flash Express can give you better first failure data capture-time, faster page-ins of critical work, and allows the system to return to normal workload operations significantly faster.

**Performance test scenarios**

**Transitioning workloads**

Performance analysis shows that Flash Express can provide big improvements to system transaction rates during paging-intensive events. In a morning transition scenario, you can see a transition from night batch workload to online transaction processing (OLTP). The scenario consists of the following tasks:

1. We ran three instances of IBM WebSphere® Application Server using IBM CICS® and IBM DB2.
2. We stopped the WebSphere Application Server instances and ran a large batch workload that pushed the OLTP data to auxiliary storage (DASD or Flash).
3. We stopped the night work and started the OLTP work, measuring the length of time it took for the OLTP work to reach full speed, requiring approximately 14 GB to be paged in.

As seen in Figure 1, it took approximately 44 seconds for the transaction rate to reach steady state when using DASD, but with Flash Express, full speed was reached in about 10 seconds!

In the first 45 seconds after the workload transition, using Flash Express resulted in a 37% increase in the number of transactions processed, and 90% reduction in average transaction response time.

**Boosting performance with pageable large pages**

As databases and Java virtual machines (JVMs) consume ever larger amounts of memory to provide better transaction response times, lowering the operating system memory management cost can significantly improve overall response times. The very fast random access and higher I/Os per second for reads in Flash Express memory (relative to disk drives) have enabled z/OS to provide support for pageable large (1 MB) pages.

As shown in Table 1, not only was the dump itself captured more quickly, but the system transaction steady-state was achieved four times faster using Flash Express.

In an SVC dump scenario, the scenario consists of the following tasks:

1. We started four WebSphere Application Server instances, and then stopped one of the instances; its data was paged out to auxiliary storage.
2. We initiated an SVC dump that included the stopped WebSphere Application Server instance. This required the paged out data to be paged in.

As shown in Table 1, not only was the dump itself captured more quickly, but the system transaction steady-state was achieved four times faster using Flash Express.

There was also an improvement in the address space and system non-dispatchable times. The system and address spaces being dumped are made non-dispatchable during certain portions of SVC dump capture processing, which means that nothing else can run in the address space during this time. In this scenario, reducing the time that the WAS address spaces were non-dispatchable means that they were able to resume normal operation faster when Flash was in use.

Now, for the first time, Flash storage is integrated into System z servers.
encrypted by the hardware at the device level and an authentication key is required to access the data. The authentication key is generated and managed by using a smart card on the Support Elements (SEs). With the standard mirrored SEs, the flash security protocol allows either SE to serve the authentication key in the event of an SE fault.

**Summary**

The initial z/OS support to exploit Flash Express for paging was generally available in December 2012 as a web deliverable, and can be downloaded from the z/OS website at [ibm.com/systems/z/os/zos/downloads](http://ibm.com/systems/z/os/zos/downloads). See “RSM Enablement Offering for Z/OS R13”. Additional support was also provided to dynamically reconfigure flash increments in March 2013 as PTF UA68146 (APAR OA40968).

You'll find additional information in the following publications:

- IBM z/OS MVS Initialization and Tuning Guide, SA22-7591, search for auxiliary storage management initialization
- IBM zEnterprise EC12 Technical Introduction, SG24-8049, IBM Redbooks® publication

**Acknowledgements**

Thanks to Harris Morgenstern and Peter Szwed for their contributions to this article.

---

**Reliability, availability, serviceability, security**

A key characteristic of System z hardware is its resilience through redundancy, and the Flash Express subsystem carries on this design principle. At all levels, the hardware and software design enables tolerance of spontaneous faults without interrupting applications.

To provide superior System z levels of reliability, availability, and resiliency, a number of technological innovations were introduced in the Flash Express design. These enhancements span the System z stack, including the adapter hardware and firmware, the System z firmware, and the virtualization of the flash architecture for the software. The Flash Express card on System z is made up of two adapter pairs formatted for RAID 10, with the data striped and mirrored across the adapters.

The dual-redundant physical and logical configuration of the Flash Express adaptors allows the system to repair and replace any component concurrently, without losing access to the data. For example, the Flash Express adapters are installed as pairs and the solid-state drives (SSDs) are configured as mirrored pairs across the adapters. SSDs on one adapter are accessible from the partner adapter through external cables. This means that if an adapter processor or SSD experiences a fault, or if firmware is being updated, data is still available through either the partner adapter or mirrored SSD.

Additionally, Flash Express allows an installation to monitor the health of the Flash Express card and to take preemptive action before any problems arise. For example, the number of overall write operations supported by flash memory technology is less than that of volatile memory technology. Flash Express is designed to have a life span comparable to that of other System z I/O subsystems. However, system firmware monitors the ‘lifetime-left’ of each SSD and schedules a repair action before the device wears out.

z/OS support for Flash Express allows an installation to configure increments of Flash Express storage offline to enable dynamic resources for the system images that require it most. The installation can also be configured to dynamically remove a Flash card from operation for maintenance. However, it’s worth noting that for enhanced availability, you can update the firmware for Flash dynamically, without first configuring the storage offline.

For security purposes, data on the Flash Express card pairs is protected by using strong encryption. Data is

---

**Table 1. SVC dump performance results**

<table>
<thead>
<tr>
<th>SVC dump metrics</th>
<th>DASD</th>
<th>Flash</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVC dump size</td>
<td>18 GB</td>
<td>18 GB</td>
</tr>
<tr>
<td>% of pages captured from aux. storage</td>
<td>50%</td>
<td>53%</td>
</tr>
<tr>
<td>DUMP elapsed time</td>
<td>189 s</td>
<td>143 s</td>
</tr>
<tr>
<td>Max address space non-dispatchable</td>
<td>58.89s</td>
<td>13.74s</td>
</tr>
<tr>
<td>System non-dispatchable</td>
<td>1.34 s</td>
<td>0.55 s</td>
</tr>
<tr>
<td>System steady state achieved</td>
<td>60 s</td>
<td>14 s</td>
</tr>
</tbody>
</table>

...
The new z Enterprise Data Compression (zEDC) Express offering provides low-cost data compression to z/OS system services and applications. By using the compression, clients could expect to see the following benefits:

- Reduced disk usage
- Increased single-thread performance of reading and writing compressed data
- Specifically for zEDC Express, optimized cross-platform exchange of data.

The zEDC Express complements existing z compression technology — the compression call (CMPSC) dictionary-based compression that DB2 and DFSMS use, for example.

Overview

The zEDC Express is implemented as a Peripheral Component Interconnect Express (PCIe) device that can be installed on zEC12 GA2 and zBC12 processors. You can install up to eight devices in a single machine where each device is sharable by up to 15 LPARs.

The zEDC Express support is available exclusively on z/OS V2R1.

The DEFLATE file format

The compression format generated and consumed by the zEDC Express is the industry-standard RFC1951 DEFLATE format. This compression format has two steps:

1. Repeated string references are removed and replaced with a length-distance pairing. These distances can go back up to 32,000 kilobytes (KB).
2. Each symbol, either an individual symbol or a length-distance pair, is replaced with a code.

You can think of this compression format as dictionary-less because it does not require any dictionary to be saved at compression time and restored at decompression time. Any repeated strings are resolved using the decompressed data itself.

For more information about the DEFLATE file format, see: ietf.org/rfc/rfc1951.txt

Setup and configuration

Each zEDC Express can be defined as part of an I/O configuration using the Hardware Configuration Definition (HCD) program. Each physical device has a physical channel identifier (PCHID) that represents where the PCIe device is installed. Each PCHID is mapped to up to 15 physical function identifiers (FIDs), which each represent a virtual function. Each virtual function is assigned to a single z/OS logical partition (LPAR).

z/OS includes new console commands to examine the state of a given virtual function. The term FID is mapped in z/OS as PFID. The D PCIE command can be used to list all virtual functions assigned to the LPAR, with further information available using the D PCIE,PFID=xxxx flavor of the command.

I/O queue and execution

Each zEDC Express virtual function receives work from a first in, first out (FIFO) queue that resides on each z/OS image. The z/OS driver chooses the best virtual device for an I/O operation as the requests are made by exploiters. The z/OS driver also monitors the maximum request size so that no request dominates a specific zEDC Express device.

After the queues are built on each z/OS image with a virtual function, access to the device is provided to each z/OS image in a round-robin fashion. For example, if 15 LPARs shared a single zEDC Express device and all 15 z/OS images had worked queued, each z/OS image would have one out of 15 requests run on its behalf. With multiple virtual functions per z/OS image and a throughput rate of up to 1 gigabyte (GB) per second, the queue delay for zEDC Express I/O is not expected to be an inhibiting factor.

As requests finish, I/O interrupts are presented back to the specific z/OS images. The unit of work that made the zEDC Express request is re-dispatched and able to process the results.

z/OS QSAM and BSAM exploitation

With PTFs on z/OS V2R1, sequential extended format data sets can support zEDC as a new type of compression. Your basic sequential access method (BSAM) and queued sequential access method (QSAM) applications can easily take advantage of the benefits of using zEDC Express for compressed format data sets.
Defining new data sets
You can request that new data sets be created in the new zEDC compressed format existing types of compression (generic and tailored) are requested today. zEDC compression can be selected at the data set level, the system level, or both. At the data set level, you can request zEDC compression through new values on the COMPACTION option in the System Managed Storage (SMS) data class. At the system level, you can request zEDC compression through new values on the COMPRESS parameter found in the IGDSMSxx member of SYS1.PARMLIB. When specified in both places, data class continues to take precedence over the system level.

Similar to the current types of compressed format data sets, there is a minimum space allocation requirement in order for System Managed Storage (SMS) to allocate a new sequential extended format data set as compressed format. The minimum requirement is 5 megabytes (MB) of primary space if a secondary space amount is specified or 8 MB of primary space if no secondary space is specified.

zEDC compression: required or preferred
In data class or in parmlib, you can specify whether zEDC compression is required or preferred for the allocation of a new data set. In the case where the zEDC Express function is not available, these options let you choose whether allocation fails or results in a tailored compressed data set.

System management facilities (SMF) types 14 and 15
You can calculate the compression ratio by using two existing fields in the SMF type 14 and type 15 records. SMF14CDS is the size of the compressed-format data set in terms of compressed user bytes. SMF14UDS is the size of the compressed-format data set in terms of uncompressed user bytes. A new field, SMF14CMPTYPE, is defined to indicate the type of compression used for the data set.

Coexistence
There could be times where a zEDC compressed-format data set is accessed on a system that does not support zEDC Express. This could be on a z/OS V2R1 system with a pre-zEC12 GA2 processor or on a z/OS V1R12 or z/OS V1R13 system with coexistence PTFs applied. For any of these cases when the zEDC Express is not available, compressed data will be read from data sets and decompressed using software algorithms. Any new data being written will not be compressed.

z/OS SMF exploitation
In z/OS V2R1, SMF can be configured to exploit zEDC Express for increased throughput of SMF record logging. This can increase the recording throughput, allowing:

• Capture of additional SMF data currently uncollected because of system logger constraints—coupling facility (CF) and SMS direct access storage device (DASD), for example.

You can think of this compression format as dictionary-less because it does not require any dictionary to be saved at compression time and restored at decompression time.

• Mitigation of z/OS image growth because of new or growing workloads, which will cause additional SMF data to be generated.

Recording configuration
SMF exploitation of zEDC Express is built on top of log stream recording. When SMFPRMxx specifies COMPRESS on the LSNAME or DEFAULTLSNAME parameters, SMF has zEDC Express compress a buffer of SMF records before it is written to the system logger.

All storage used for zEDC Express I/O requires fixed storage for the input and output buffers. A new SMFPRMxx parameter, PERMFIX, is available as a sub-parameter of COMPRESS and as a global SMF parameter, letting you specify the amount of storage used for the SMF buffers that can remain permanently fixed. Each time SMF requests zEDC Express to compress a buffer, the buffers must be fixed. Doing this for each zEDC Express I/O operation increases the overhead of the operation. Increasing this value decreases the overhead, but might decrease the amount of fixed storage available to other applications.

The PERMFIX parameter can range from 1 MB to 2 GB. New IFAQUERY and SMF type 23 record output fields can assist in the tailoring of the PERMFIX value.
When compressed data is processed by IFASMFDL, it decompresses the SMF records for selection and writing. SMF data is only compressed while it is resident in the system logger. Ideally, IFASMFDL always runs on a z/OS V2R1 system with access to zEDC Express. In the event that the z/OS image does not meet this requirement, IFASMFDL returns an error with message IFA849I. If compressed SMF records must be read on a pre-z/OS V2R1 system, or on a system without access to zEDC Express, the new SOFTINFLATE parameter lets installations process compressed SMF records using a software algorithm. IFASEXIT cannot be used to read compressed SMF records.

Coexistence
PTFs for OA41156 should be installed on z/OS V1R13 and V1R12 systems to tolerate the new SMFPRMxx keywords and to allow for the IFASMFDL SOFTINFLATE keyword and software decompression support.

On z/OS V2R1, any IFASMFDL job now needs to specify a region size of 4 MB or greater because IFASMFDL has been enhanced to take advantage of multi-block log stream browsing, which aids in processing compressed blocks and benefits all users. If you decide to compress SMF records, you might want to consider the dumping environments that are used. You might want to move or restrict IFASMFDL jobs to systems with access to zEDC Express for optimal performance. Consider adding SOFTINFLATE to IFASMFDL jobs after investigating the cost, performance, and compatibility implications. Also note that entry-to-element ratios of CF structure-based log streams might change as the data is compressed. The logger might encounter entry or element full conditions until it can resample to the new ratios.

For more information about zlib, see:
• MVS Programming: Callable Services for High-Level Languages, SA23-1377
• zlib.net

The SMF 74 subtype 9 record has been updated with new PCIe and zEDC Express statistics. You can use RMF™ Monitor I to post-process these records and obtain the following statistics:
• I/O queue and execution time
• Data transfer rates, compressed and uncompressed
• Number of compression and decompression requests.

The statistics provided can be used to determine the usage of the zEDC Express from a given z/OS image. It is important to note that the I/O queue time is reported for a single z/OS image; however, it provides a view of the zEDC Express across all z/OS images that share the device.

For a complete list of the zEDC Express statistics provided using RMF Monitor I, see z/OS RMF Report Analysis, SC34-2665.

Conclusion
The zEDC Express provides an exciting new compression technology on the zSeries platform. The exploitation of this technology throughout the z/OS software stack will provide benefits to several aspects of data serving and processing.
With the System zEC12, customers can keep critical sensitive data on flash adapters with a certainty of security on which their business can depend. Security administrators can now accomplish end-to-end encryption using a symmetric key also known as the flash authentication key. The flash authentication key is a 256-bit Advanced Encryption Standard (AES) key used for authentication of security critical host system communications to Solid State Drives (SSD). The flash authentication key controls access to the SSD where data is encrypted and decrypted.

Using a smart card and an integrated smart card reader on a Support Element (SE), the flash authentication key is managed within this environment in such a way that compromising the key is extremely difficult, if not impossible. All key generation, encryption, and decryption, takes place on the smart card. Keys are never in the clear. The truly sensitive key, the flash authentication key, is located only in a file on the SE until served.

How it works
The SE is a physically secured personal computer running Linux. It has one integrated card reader that reads one smart card. A monitored event happens when inserting or removing the smart card. Simultaneously, tight coupling validation occurs, which means the SE verifies that a reinserted smart card belongs to the SE before it proceeds.

In Figure 1, the SE initializes the environment by invoking APIs within the Integrated Key Controller (IKC). The IKC loads an applet to the smart card inserted in the integrated card reader. The smart card applet, as part of its installation, creates a 2048-bit RSA key pair, the private component of which never leaves the smart card. The applet also creates two 256-bit AES keys, of which one, known as the key-encrypting key (KEK), remains on the smart card. The other AES key becomes the flash authentication key and is encrypted by the KEK. An allocated buffer contains the KEK-encrypted flash authentication key and the unique serial number of the SE. The buffer is padded for Public-Key Cryptography Standards #1 (PKCS#1), and then encrypted by the RSA public key. The resulting encrypted blob then writes to a file on the SE — the flash file.

This design defines a tight-coupling of the flash file on the SE to the smart card, which ensures that any other SE will not be able to share the file or the smart card associated with a given SE. Specifically, the tight-coupling refers to the unique relationship between the encrypted flash file on the SE and the smart card. The unique KEK on the smart card and the unique RSA key pair encrypt the flash key first (with the KEK) and then encrypt the encrypted flash key along with the SE serial number (with the public key of the key pair on the smart card). This makes all the flash files unique and all such smart cards uniquely tied to the SE because the KEK and the RSA key pair used are unique to that smart card.

High-level data protection
The end-to-end encryption along with the tight coupling of an SE to a smart card provides the System z Flash environment with a secure solution to data protection during upgrades, power outages, and other unplanned disruptions. With separation of the flash authentication key from the smart card, and tight coupling, the SE cannot serve the encryption key without its associated smart card, and its associated smart card cannot be used with any other SE. The secure encryption design point discussed in this article insures the high level of data protection required by System z customers. ■
Integrated Cryptographic Services Facility (ICSF) proudly introduces Enterprise PKCS #11 (EP11) services on the IBM zEnterprise EC12 (zEC12) and the IBM zEnterprise BC12 (zBC12). This support enables new workloads requiring secure key PKCS #11 services and also allows existing clear key PKCS #11 workloads to benefit from secure key without modification. EP11 is available as a web deliverable in Cryptographic Support for z/OS V1R12 and V1R13 (FMID HCR77A0), and will be in the base of z/OS V2R1.

**How does it work?**

Previous releases of ICSF already provide clear key PKCS #11 services. EP11 introduces a new secure key PKCS #11 firmware load for the Crypto Express4S cryptographic coprocessor. This gives the Crypto Express4S card two flavors of cryptographic firmware: Common Cryptographic Architecture (CCA) and EP11.

To externalize the EP11 services available in this new firmware load, the ICSF callable services have been enhanced to provide secure key PKCS #11 support.

**PKCS #11: an overview**

RSA Laboratories of RSA Security Inc. offers its Public Key Cryptography Standards (PKCS) to developers of computing systems that use public key and related cryptographic technology. PKCS #11, also known as Cryptoki, is the cryptographic token interface standard. It specifies an application programming interface (API) to devices, referred to as tokens, that hold cryptographic information and perform cryptographic functions. ICSF supports PKCS #11, providing an alternative to IBM’s Common Cryptographic Architecture (CCA) and broadening the scope of cryptographic applications that can make use of System z cryptography.

On most single-user systems, a token is a smart card or other plug-installed cryptographic device or coprocessor, accessed through a card reader or slot. In contrast, z/OS PKCS #11 tokens can be created using system software such as Resource Access Control Facility (RACF®), the gskkyman utility, ICSF Interactive System Productivity Facility (ISPF) panels, or by applications using the C API. Each token has a unique token name or label that is specified by the end user or application at the time the token is created and later used to refer to the token.

ICSF support for PKCS #11 provides encryption and decryption, signature generation and verification, message digestion, key generation and transport, and random number generation.

**EP11 secure key types**

The EP11 Cryptographic Coprocessor supports these secure key types:

- Data Encryption Standard (DES)
- Advanced Encryption Standard (AES)
- Generic Secret–for keyed-hash message authentication code (HMAC) services
- RSA
- Digital Signature Algorithm (DSA)
- Elliptic Curve DSA (ECDSA).

Existing mechanisms will support clear and secure key objects as appropriate.

Secure keys are created by specifying `CKA.ibm_secure=true` in the object’s attribute list. A new System Authorization Facility (SAF) profile has been created in the CRYPTOZ class that governs the creation of secure and clear key objects, in case this attribute is not
specified when a key object is generated. By varying the access granted to the profile, the administrator has total control over the conditions that are required to generate clear key objects. The default behavior, that is, when the CKA _IBM _SECURE attribute is not present and the SAF profile does not exist, is to make sensitive keys secure. Clear sensitive keys are allowed only when an EP11 coprocessor is not available. The format of the profile is:

**CLEARKEY.token-name**

### FIPS 140 and BSI compliance modes

More and more regulatory standards require computing systems that perform cryptography to adhere to government regulations. In the United States, the National Institute of Standards and Technology (NIST) is the governing body. NIST publishes the Federal Information Processing Standards (FIPS), which are guidelines for use by government and commercial vendors. FIPS 140, currently in its second revision, details the Security Requirements for Cryptographic Modules, such as minimum key sizes and the startup self checks that are to be performed. The European Union (EU) has its own set of requirements published by Bundesnetzagentur, known as BSI. They are similar, but not exactly the same, as the FIPS 140 standards. As a general rule, the BSI standard is slightly more restrictive than FIPS 140.

The EP11 coprocessor firmware was designed to comply with both of these standards. There is a compliance mode setting that can be configured for each cryptographic coprocessor domain from the Trusted Key Entry (TKE) workstation. The minimum compliance mode is FIPS 2009, which has the following restrictions:

- Algorithms and keys below 80 bits of security are not permitted.
- RSA private keys cannot be used without padding.
- Newly-generated asymmetric keys always undergo self tests.
- The minimum key size on HMAC is one-half of the algorithm’s output size.
- Non-FIPS-approved algorithms (as of 2009) are not present.

The other compliance modes — FIPS 2011, BSI 2009, and BSI 2011 — have more or different restrictions.

### The z/OS security stack: before and after

Figure 1 depicts the z/OS security stack prior to the introduction of EP11.

```
Figure 1. The z/OS security stack prior to EP11
```

- The middle box in the diagram represents ICSF. Note that while PKCS #11 services are fully-functioning, they don't offer the secure key capability that the CCA services are famous for. Any use of the Crypto Express coprocessors is limited to processor offloading. Hence, this is a clear key solution only. All key material stored in the token data set (TKDS) is in the clear.

Introduce the EP11 coprocessor and the picture changes, as shown in Figure 2 on page 23.

The EP11 coprocessor has full support for the PKCS #11 standard and is secure-key-capable. Keys stored in the TKDS can be protected by a coprocessor-bound master key, known as the P11 master key.

### Initializing an EP11 coprocessor

In order to initialize an EP11 coprocessor, you must load a P11 master key. The P11 master key is used to protect all secure PKCS #11 keys. This master key must be loaded in parts from a TKE workstation using smart cards. The P11 master key cannot be loaded by ICSF alone.
If an instance of ICSF is configured with multiple EP11 coprocessors, the same P11 master key must be loaded on each coprocessor. Additionally, if you are planning to share your TKDS across ICSF instances in a sysplex, you can use the TKE workstation to define an EP11 domain group to load the same P11 master key on all cryptographic domains (mapped to z/OS LPARs) sharing the same active TKDS. After the new P11 master key is loaded, it must then be committed by the TKE workstation. After your new P11 master key is loaded and committed by TKE, the next step is to initialize your TKDS.

In order for applications to use secure key PKCS #11 services, ICSF must have a TKDS specified in the ICSF options data set and the TKDS must be initialized in accordance with the EP11 cryptographic coprocessor configuration. The process for initializing an empty TKDS and an existing TKDS that is populated with clear key objects is the same. There is no need to create a new TKDS for secure key processing. By initializing the TKDS, the EP11 master key will become active and secure key operations will be enabled.

To initialize a new or existing TKDS:

1. Select option 2, “MASTER KEY MGMT”, from the ICSF primary ISPF menu.
2. From the “ICSF - Master Key Management” panel, select option 3, “TKDS MK MANAGEMENT”.
3. From the “ICSF - TKDS Master Key Management” panel, select option 1, “INIT/UPDATE TKDS”. This will initialize the active TKDS header record and make the EP11 master key active, thus making secure PKCS #11 services available to applications on that system.

If the TKDS is shared between systems in a sysplex, each instance of ICSF that shares the TKDS must repeat the TKDS initialization steps or must be restarted before it can begin to use secure PKCS #11 services.

Changing the EP11 master key

The ICSF Coordinated Change Master Key function has been extended to support changing the P11 master key for the active TKDS. Similar to initialization of the P11 master key, the changes require a TKE workstation to load new P11 master key parts into the P11 new master key registers from smart cards. When all of the new P11 master key parts have been loaded, the TKE workstation must commit the new P11 master key value. Once the new P11 master key has been loaded and committed on all applicable EP11 coprocessors, the coordinated change master key function must be called to re-encipher the secure EP11 key material in the TKDS and promote the new P11 master key to the current P11 master key register. You must use the coordinated change master key function for sysplex and non-sysplex environments.

Summary

Enterprise PKCS #11 support brings ICSF to a whole new level of secure key cryptographic processing. By moving PKCS #11 functionality into the secure boundaries of the cryptographic coprocessors, ICSF can now provide highly-secure key operations using an industry-accepted standard. EP11 is yet another tremendous enhancement to ICSF that we hope you find brings great value to the z/OS platform.

References

For more information, see the following publications:

• z/OS Cryptographic Services ICSF Administrator’s Guide, SA22-7521
• z/OS Cryptographic Services Integrated Cryptographic Service Facility Writing PKCS #11 Applications, SA23-2231.
Your data has to be secure 24/7, especially data transported on the Internet. But have no fear! This is where SSL (Secure Sockets Layer) and TLS (Transport Layer Security) saves the day, by encrypting and ensuring data integrity for data in transit.

z/OS System SSL now supports TLSv1.2, the latest TLS protocol version. TLSv1.2 is documented in IETF RFC (Internet Engineering Task Force Request for Comments) 5246. TLSv1.2 is available with:

- z/OS V2R1
- z/OS V1R13 with APAR OA39422

TLSv1.2 provides a bunch of security enhancements for your z/OS System SSL applications we think you’ll endorse.

What are the advantages of using TLSv1.2 protocol?

TLSv1.2 provides the following security enhancements:

- Cipher suite specific Pseudo-Random Functions (PRFs) replace the MD5 and SHA-1 PRFs. Now, the default PRF is a SHA-256 based one.
- For message authentication, more secure SHA-256 and SHA-384 algorithms replace SHA-1 and MD5 algorithms, which are now considered weak hashes.
- Clients and servers can now specify the hash and signature algorithms they’ll accept.
- TLSv1.2 no longer supports DES (56-bit) and RC2/RC4 (export) ciphers because they are now considered weak encryption algorithms.

More secure cipher specifications are now available!

TLSv1.2 introduces more cipher choices than ever before. As in earlier versions, the cipher used between the client and server application is negotiated during the TLS handshake and the final cipher choice depends on the ciphers in common between the applications and the certificates used.

System SSL already provides a variety of certificate types for use with TLSv1.2 and earlier TLS protocols, allowing certificates that contain an RSA, DSA, DH (Diffie-Hellman), or ECC (Elliptic Curve Cryptography) key with varying key sizes. On z/OS, you can generate the keys and certificates using the RACF RACDCERT command, System SSL gskkyman utility, or z/OS PKI Services.

The new cipher choices provided by System SSL include:

- 003B–0040
- 0067–006B
- 009C–00A5
- C023–C032.

These ciphers allow you to use 128-bit or 256-bit AES encryption with either SHA-256 or SHA-384 message hashing. (In earlier TLS protocol versions, these encryption mechanisms only provided SHA-1 and MD5 message hashing.) Additional new cipher choices allow the specification of 128-bit or 256-bit AES-GCM (Galois Counter Mode) encryption with either SHA-256 or SHA-384 message hashing. To use the new AES GCM encryption ciphers, you must have ICSF available.
Table 1. The System SSL supported hash and signature algorithm pairs

<table>
<thead>
<tr>
<th>Signature algorithm enumerator</th>
<th>Hash and signature algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101</td>
<td>MD5 with RSA</td>
</tr>
<tr>
<td>0201</td>
<td>SHA-1 with RSA</td>
</tr>
<tr>
<td>0202</td>
<td>SHA-1 with DSA</td>
</tr>
<tr>
<td>0203</td>
<td>SHA-1 with ECDSA</td>
</tr>
<tr>
<td>0301</td>
<td>SHA-224 with RSA</td>
</tr>
<tr>
<td>0302*</td>
<td>SHA-224 with DSA*</td>
</tr>
<tr>
<td>0303</td>
<td>SHA-224 with ECDSA</td>
</tr>
<tr>
<td>0401</td>
<td>SHA-256 with RSA</td>
</tr>
<tr>
<td>0402*</td>
<td>SHA-256 with DSA*</td>
</tr>
<tr>
<td>0403</td>
<td>SHA-256 with ECDSA</td>
</tr>
<tr>
<td>0501</td>
<td>SHA-384 with RSA</td>
</tr>
<tr>
<td>0503</td>
<td>SHA-384 with ECDSA</td>
</tr>
<tr>
<td>0601</td>
<td>SHA-512 with RSA</td>
</tr>
<tr>
<td>0603</td>
<td>SHA-512 with ECDSA</td>
</tr>
</tbody>
</table>

* Indicates that the algorithm is not supported with z/OS

Signature algorithm and hash pair support

With TLSv1.2, clients and servers can specify a list of the signature algorithm and hash pairs they want to allow when establishing a secure connection. Clients and servers specify one or more signature algorithm and hash pairs for consideration by their session partner. This signature algorithm list is then transmitted during the TLS handshake process between the client and server. These algorithm pairs are useful for peer applications to limit the ciphers and certificates allowed during the handshake process.

Putting it all together: TLSv1.2, hash and signature algorithm pairs, and the new ciphers

Now, how do you put all of these great security enhancements to work in your System SSL applications?

If the client or server specifies a subset of the list of signature algorithm pairs listed in Table 1, then the signatures on the certificates presented in the TLS handshake must be included on that list. The examples below illustrate how these elements work together.

Example 1: Client/server handshake failure – mismatching signature algorithm list

In this example, we show a server that is configured for server/client authentication. This indicates that the server requests the client’s certificate during the TLS handshake process. (By default, applications are only configured for server authentication, which indicates that the server will send its certificate to the client.)

In this case, however, the TLS handshake fails with error 467 because the signature algorithm associated with the server’s certificate is not represented in the signature algorithm list sent from the client to the server.

• Server Configuration
  - Certificate: Signed by a RSA CA
  - Ciphers: 003E
  - SSL/TLS protocol(s): TLSv1.2
  - Signature Algorithm Pairs List: 06010603050105030401040304020301030302020102032020101
  - Server/client authentication is configured

• Client Configuration
  - Certificate: Signed by a DSA CA
  - Ciphers: 003E
  - SSL/TLS protocol(s): TLSv1.2
  - Signature Algorithm Pairs List: 0202020303040305030603

Example 2: Successful client/server handshake

The configuration in Example 2 is the same as in Example 1, but the client’s signature algorithm pairs list now includes 0301 and 0401. This means that the TLS handshake is successful because 0301 and 0401 are RSA signature algorithm pairs.

• Server Configuration
  - Certificate: Signed by a RSA CA
  - Ciphers: 003E
  - SSL/TLS protocol(s): TLSv1.2
  - Signature Algorithm Pairs List: 06010603050105030401040304020301030302020102032020101
  - Server/client authentication is configured

• Client Configuration
  - Certificate: Signed by a DSA CA
  - Ciphers: 003E
  - SSL/TLS protocol(s): TLSv1.2
  - Signature Algorithm Pairs List: 0202020303040305030603

Example 3: Client/server handshake — no appropriate cipher for certificates

In Example 3, the handshake fails with an error 402 (no SSL ciphers specifications) because the certificates used by both session partners are Elliptic Curve Digital Signature Algorithm (ECDSA), but neither the server nor client cipher list contain ECC ciphers appropriate for the certificates.

• Server Configuration
  - Ciphers: Using the default SSL V3 cipher list.
  - SSL/TLS protocol(s): TLSv1.2
  - Signature Algorithm Pairs List: 06010603050105030401040304020301030302020102032020101
  - Server/client authentication is configured

• Client Configuration
  - Ciphers: 00050009000A
  - SSL/TLS protocol(s): TLSv1.2
  - Signature Algorithm Pairs List: 0303

To resolve this problem, you must add an ECC cipher such as C005 to both the server’s and client’s cipher list.
How do I take advantage of TLSv1.2?
To enable your applications for TLSv1.2, do one of the following:

- Use z/OS Communication Server’s AT-TLS (Application Transparent – Transport Layer Security) support. See the z/OS Communication Server: IP Configuration Guide, SC27-3650-00, for information about enabling TLSv1.2 with AT-TLS.
- Specify TLSv1.2 SSL environment variables
- Update your application code to support TLSv1.2

Specifying TLSv1.2 System SSL Environment Variables
If your application is already enabled for 4 character ciphers, you may be able to simply export the following environment variables and values prior to running your application to enable TLSv1.2:

- GSK_PROTOCOL_TLSV1_2 — Specify that you wish to enable TLSv1.2 (TLSv1.2 is disabled by default).
- GSK_V3_CIPHER_SPECS_EXPANDED — Specify the 4-character cipher specifications that you want to take advantage of. The order of precedence is from left to right.
- GSK_TLS_SIG_ALG_PAIRS — Specify the list of signature algorithm pairs that your application can support.

Example 4 – Exporting SSL environment variables in USS
Example 4 shows the following SSL environment configured:

- TLSv1.2 is enabled
- The application is enabled for 4-character ciphers 00A1, 00A0, 0069, and 0068.
- The signature algorithms in use are 0501 (SHA-384 with RSA) and 0401 (SHA-256 with RSA).

```bash
$> export GSK_PROTOCOL_TLSV1_2-ON
$> export GSK_V3_CIPHER_SPECS_EXPANDED=00A100A000690068
$> export GSK_TLS_SIG_ALG_PAIRS=05010401
```

Updating your application code
In Example 5, we show an example of application code that enables the System SSL environment to use TLSv1.2, enable usage of a small subset of new cipher specifications, and define a list of supported signature algorithm pairs. This accomplishes the same configuration documented in Example 4.

Example 5 – Enabling an application for TLSv1.2
```c
#include <gskssl.h>
...
int initialize_ssl_environment ( void ) {
    gsk_status retval;
gsk_handle envHandle = NULL;
gsk_handle conHandle;
char *safKeyRing = “MYRING”;
char *certLabel = “mylabel”;
GSK_ENUM_VALUE tlsv12 = GSK_PROTOCOL_TLSV1_2_ON;
char *v3CipherSpecsExpanded = “00A100A000690068”;
char *sigAlgs = “05010401”;
...
return retval;
}
```

TLSv1.2 deciphered
And now you have an idea of how to update your applications to take advantage of all the great new security features offered by TLSv1.2!

For more information about TLSv1.2 and the new ciphers in System SSL, see z/OS Cryptographic Services System Secure Sockets Layer Programming, SC14-7495.
In z/OS V2R1, the z/OS file system (zFS) extended directory removed an architected limitation that affected performance for some zFS users. In earlier releases, when a zFS directory contained a large number of file names (say, over 10,000 entries), certain zFS file operations began to experience delays because zFS stored directory entries in a linear sequence of 8K blocks. Searching for a name in a zFS directory meant reading through the 8K blocks in order until the name was found (or not found). The larger the directory, the more names, the longer it took to find the name.

The zFS extended directory support introduces a new version of the zFS file system, version 1.5 aggregate, which removes this restriction and two others. The result is overall better performance for most zFS users.

Decision time
There are several options to consider when you migrate or convert—time to make some decisions. Conversion programs support changing the aggregate version to 1.5 and optionally converting the directories. To use this new support in a way that makes sense for your installation, let’s explore the options, and look at the pros and cons of each option.

Some details before migration
The new zFS extended directory introduces aggregate version 1.5. IBM continues to support version 1.4 aggregates, so conversion to 1.5 is optional.

There are two restrictions removed with version 1.5 aggregates:

- Maximum aggregate size changes from 4 TB to 16 TB.
- Maximum number of sub-directories in a single directory changes from 65535 to 4,294,967,293.

The new version 5 directory can aid migration, and allows you to convert specific directories. All newly created directories in version 1.5 aggregates are version 5 (v5). Additionally, zFS supports both the current directory format (v4) and v5 directories in a version 1.5 aggregate. This article refers to 1.4 and 1.5 aggregates as 1.4 and 1.5 file systems.

Option 1: Brute force
This is the most straightforward option if you have 1.4 file systems and you want to start using 1.5 file systems. It is also an option if you are still using HFS type file systems.

In this procedure, you create corresponding 1.5 file systems for each of the current file systems that you want to convert. Next, copy the current file system to the new 1.5 file systems.

1. Create new 1.5 file systems that correspond to each of your current file systems that you plan to migrate.
2. Mount the current file systems.
3. Mount the new 1.5 file systems.
4. Copy the contents of each current file system to the new 1.5 file systems using the copy command. Use one of the following commands to copy all the contents from the current working directory to /tmp/z:

   ```bash
   cp -R -p . /tmp/z or pax -rw -pe . /tmp/z
   ```

5. Verify file tree and all file contents match.
6. Begin to use new aggregates (update BPXPRMxx, and so forth).
Advantages:
• You have a backup of all data.
• If you keep your current file systems, the data will be there if needed.

Disadvantages:
• You temporarily need at least double the DASD space that is currently in use.
• You need to update BPXPRMxx members or other scripts.
• The source file system must be inactive, or you can lose recent updates.

Option 2: Some finesse
BPXWH2Z is the HFS to zFS migration tool introduced several years ago. This is a handy tool if you had to remain with HFS until the extended directory support was available.

BPXWH2Z has a panel interface to alter the space allocation, placement, SMS classes, and data set names. When using BPXWH2Z for the new zFS file systems, you can control which file system version, 1.4, or 1.5, it creates. The default is 1.4. Change this default by specifying format _ aggrversion=5 in your IOEPRMxx configuration options file before IPL or by dynamically changing the option by using the zfsadm config -format_aggrversion 5 command.

Advantages:
• Migrate one or many file systems
• Support for character substitution
• Pre-allocate the zFS file system or let the tool create it
• If the file system is mounted, it will be unmounted and the new zFS put in its place.

Disadvantages:
You must temporarily double (at least) the DASD space that is currently in use.

Option 3: Mounting change
Use the new change _ aggrversion _ on _ mount option in IOEPRMxx to convert version 1.4 aggregates to 1.5 aggregates at mount.

Advantages:
• No directories are converted
• Existing v4 directories remain v4. New directories are v5.
• Preserve your v4 directories while trying v5 directories
• No disadvantages.

Option 4: Commanding change
Use the new zfsadm convert {-path <pathname> | -aggrversion <aggrname>} command to either change the aggregate version or convert a directory.

1. Use the -aggrversion parameter to change the file system to version 1.5.
2. Use the -path parameter to convert pathname from a v4 directory to a v5 directory.
3. If the file system is not already 1.4, it changes to 1.5 as well when the directory is converted.

After successful running of the zfsadm convert command, all new directories are v5 directories.

The zfsadm convert command marks the file system as a 1.5, but it does not convert any of the directories at that time. Alternatively, it converts a single directory to a v5 directory using pathname.

Advantages:
• Conversion is in place; there is no need for a second file system. Note that conversions are atomic.

Disadvantages:
• You must invoke the command for each directory that you want to convert.

Option 5: Take your time
Use the new converttov5 mount parameter on targeted file systems. This parameter is helpful if you want to convert your directories over time; you do not have immediate need for the conversion. When you use the converttov5 parameter it:

• Marks the aggregate as a 1.5
• Creates all new directories as v5 directories and converts existing v4 directories to v5 directories as you access them.

Advantages:
• This option does not require double the DASD space.

Disadvantages:
• You might have unexpected delays as a conversion takes place on first access to a v4 directory within the 1.5 file system.
• You must consider not to auto-convert too much per system IPL or startup because this can slow performance significantly. Therefore, you must analyze the size of the zFS file tree and the amount accessed at startup.

More about converttov5
zFS provides CONVERTTOV5=ON parameter on IOEPRMxx. Using it, results in all file systems mounted on the system to change to 1.5 file systems. Directories are also converted to v5 format as accessed.

• To exclude specific file systems, use the new NOCONVERTTOV5 mount parameter.
• The CONVERTTOV5 parameter stays in effect on the mounted file system until it is unmounted, and then mounted. Without the attribute, dynamic conversion cannot occur.
You can see if this attribute is set on the output of the `zfsadm agrinfo -long` command as:

```
$ zfsadm agrinfo -long usszfs.bigzfs1.zfs
```

For example purposes, here's partial output:

```
sysexp-aware, converttov5
```

### Option 6: Offline conversion

Use the new `converttov4` or `converttov5` function in `ioefsutl`. z/OS V2R1 introduces the `ioefsutl` batch utility suite that contains four supported functions that can run as batch jobs:

- Format
- Salvage
- Converttov4
- Converttov5.

Before `ioefsutl` can process a zFS aggregate, you must unmount (and not attach). What this does:

- Use `converttov5` to change a version 1.4 file system to a version 1.5 file system and convert all the existing directories to v5 directories.
- Use `converttov4` to convert all v5 directories to v4 directories, and then change the version 1.5 file system to a version 1.4 file system.

**Advantages:**
You can target specific file systems to convert.

**Disadvantages:**
The file system must be unmounted. When a file system is larger than 4 TB, or has a directory with more than 64K-1 subdirectories, or both, there's no support for conversion from 1.5 to 1.4.

Another new IOEPRMxx parameter `zfsadm` also introduces IOEPRMxx parameter `format_aggrversion= 4 5`.

The default aggregate version remains 1.4.

The `ioefsutl` format utility (and the stand-alone utility `IOEAGFMT`) now supports a flag to specify `–version4` or `–version5`. Additionally, the `zfsadm format` command now supports `–version4` or `–version5`. If specifying no option, the value of the `format_aggrversion` parameter is used.

Note that as is the case with most parameters that can be set in the IOEPRMxx member, all three new parameters can be queried using `zfsadm configq` and set by using `zfsadm config`.

This configuration option affects users of the following:

- Format aggregate API such as the `BPXWH2Z` tool
- z/OS UNIX TSO/E ISHELL command
- z/OS UNIX automount shell command with the allocany or allocuser keyword.

### New fileinfo command

New `zfsadm fileinfo` command reports several key fields of information on files and directories, including the directory version:

```
$ zfsadm fileinfo /u/
```

For example purposes, here's partial output:

```
dir version 5
```

zFS Extended Directory Support

The `aggrinfo` command shows the version of the file system

```
$ zfsadm agrinfo -long
```

For example, here's partial output:

```
version 1.5
```

### Migration points

zFS makes clear two messages in z/OS Migration, GA32-0889-00:

- You must be running z/OS V2R1 in your enterprise before using zFS version 1.5 aggregates.
- Do not mount zFS version 1.5 aggregates on any release prior to z/OS V2R1.

### Find out more

The options and suggestions in this article give you a well-rounded view of the benefits and methods to work with 1.5 file systems in your installation. For complete details, see V2R1 Distributed File Service zFS Administration, SA23-6887-00. Soon, you will be reaping the benefits of efficient, better performing large directories. ■
After the advent of System-Managed Storage (SMS), there was little change in the landscape of space-management processing for a long time. We got very comfortable with the standard three-tier hierarchy, which is made up of the primary tier (level 0) and the two migration tiers: migration level 1 (disk) and migration level 2 (tape). Recent advances in tape and disk technologies have changed that landscape dramatically and provided exciting new opportunities for managing data on the System z platform. Coupled with these changes in the hardware landscape, we are proud to present the next step in the evolution in space management—the Data Facility Storage Management Subsystem (DFSMS) storage tiers solution.

First, a little background
DFSMS provides information lifecycle management (ILM) on System z through the data class, storage class, management class, and storage group constructs. These constructs enable automated, policy-based ILM of data through its creation, useful life, and eventual deletion. When done optimally, information is managed in a manner that aligns storage costs with the changing business value of information.

A storage tier is a class of devices that has a defined set of performance, availability, accessibility, and capacity characteristics. Today, DFSMS enables users to define tiers using the SMS storage class and storage group constructs. A storage class categorizes the various performance, availability, and accessibility attributes of data to named constructs.

Two or more storage tiers comprise a storage hierarchy. Tiers comprised of devices with better performance, availability, and accessibility are higher in cost and reside higher in the hierarchy. Tiers comprised of devices with slower performance, lack of availability and accessibility features, or both, are less expensive and reside lower in the hierarchy. A fundamental function of storage management software is to place data appropriately on the storage tier, within the hierarchy that best matches the data's performance, availability, accessibility, and capacity requirements.

Data is currently managed in two distinct storage hierarchies. The first hierarchy is called the primary hierarchy (level 0). Data in this hierarchy is owned by users and applications and is readily accessible to them. This data is managed by DFSMShsm. While there could be multiple tiers of storage within the primary hierarchy to which a data set can be allocated, once it is allocated on a tier, DFSMS has a static view of that data. There are no policies to enable automated data movement within this hierarchy. The second hierarchy is the migration hierarchy. After data in the primary hierarchy remains inactive for a policy-based amount of time, DFSMShsm moves data down to the migration hierarchy. This data becomes owned by DFSMShsm because users and applications cannot access the data until DFSMShsm recalls it back to a device in the primary storage hierarchy.
As the business needs and value of data have evolved, shortcomings with this fundamental architecture are magnified:

1. There is no policy-based data movement solution available for moving data within the primary storage hierarchy

2. Data that is in active use cannot be moved between tiers of storage.

The problem is that the data remains on the primary hierarchy where it was allocated, which may not be the best business value location for it.

Welcome to storage tiers!

In z/OS V2R1, DFSMS introduces the storage tiers solution, which enables automated, policy-based space management that moves data from tier to tier within the primary hierarchy (level 0). Automated movement is provided within the existing DFSMShsm space management functions. This movement is referred to as a class transition. During class transitions, data remains in its original format and can be immediately accessed after the movement is complete. SMS automatic class selection (ACS) routines are used to define the class transition policies.

What are the benefits?

There are many good reasons to implement class transition policies within the primary hierarchy over the traditional DFSMShsm migration policies. For one, you might have some data that always needs to be readily accessible. It might never be acceptable to migrate and subsequently wait for a recall before accessing it. In this particular scenario, a data set could be allocated on a higher tier and then later moved to a lower tier for permanent retention. A second example of how to use class transitions is that as data sets age, they might go through frequent periods of migration and recall activity before they are migrated for a final time. As shown in Figure 1, multiple migration and recall iterations can be replaced with a single transition and migration to migration level 2 (ML2).

Implementing storage tiers

Because class transitions are just another form of space management, transition processing is integrated into the existing automated DFSMShsm space management functions of primary space management, interval migration, and on-demand migration. When these automatic functions select a volume for processing, in addition to the existing processing, they will perform class transitions.

A data set is eligible for a transition when one of the three “class transition” criteria in the management class indicates that the data set should be moved. The criteria are:

- Time since creation
- Time since last use
- At periodic intervals.

If DFSMShsm determines that a data set is eligible for migration and transition processing, the data set is migrated. This prevents the data set from being transitioned one day, only to be migrated the next.

When a data set is eligible to be moved, DFSMShsm calls the ACS routines with a new environmental variable named SPMGCLTR (for “space management class transition”). SPMGCLTR indicates that the routines are being called for a class transition. The storage class, management class, and storage group routines are processed, in that order. One or all of the routines can return a value that dictates the need for a transition. If none of the routines return a value that differs from the data sets’ currently assigned values, no action is performed. If only the management class differs, the management class for the data set is updated and no data movement occurs. If the storage class or storage group differs, the data set is moved physically.

DFSMShsm performs class transition movement by calling DFSMSdss to do a data set level copy with delete. The management class indicates which type of movement should be performed: standard I/O, fast replication, or preserve mirror. When standard I/O is used for the data movement, the transition can be across storage control units. Restrictions apply as to when fast replication and preserve mirror can be used. After the transition is complete, the data set is immediately available for access by users and applications. DFSMShsm does not maintain any control data set records for transition processing. New FSR type 24 records are created for reporting purposes.

If a DB2 object, a CICS object, or a zFS data set is expected to be open at the time of the transition, the new serialization error exit management class...
attribute can be used to indicate that DFSMSdss should temporarily close the data set, move it, and then reopen the data set. The serialization error exit attribute indicates the type of data set, so that DFSMSdss can call the appropriate interface to perform the close and subsequent open. The close operation is performed only if there are no active transactions against the data set.

A “real world” example

Let’s walk through an example of using this new function. Let’s say that a health insurance company creates a DB2 object for each patient visit. For the first 45 days since creation, the insurance company has determined that access to these objects is critical in order to achieve specific service-level objectives. But, after 45 days, the access to these objects becomes more historically interesting rather than critical to achieving service levels. The company wants to store these objects on a lower-cost storage device after 45 days, regardless of the current patterns of “hot” or “cold”. The company does not want to migrate the objects to tape because it wants to be able to immediately access the objects at all times. As Figure 2 shows, this can be accomplished with the DFSMS storage tiers solution.

The first implementation step is to define one or more storage groups made up of low-cost volumes to which the objects can be transitioned—a storage group made up of nearline SAS devices, for example. These devices are approximately one-half to one-third the cost of standard enterprise disks. We’ll name this storage group SASDISK.

The second step is to update the ACS storage group routines so that when the environmental variable is SPMGCLTR, the SASDISK storage group is returned.

The third step is to update the management class to which these DB2 objects are defined to indicate that a class transition should occur 45 days after the creation date. Within the management class serialization error exit criteria, DB2 is specified to indicate that these are DB2 objects that need to be closed before the transition occurs.

That’s it! Notice that no changes are required within DFSMSHsm. The next time one of the DFSMSHsm automatic space management functions processes, it will identify that the SMS policies indicate that transitions should be performed and will process the transitions as defined by the policies.

This is just one of many use cases that this new function enables. Explore the possibilities that this function provides as you take your storage architecture to the next tier. Try it out and let us know what you think!
Are your batch workloads trickling through? Does it take forever to recall your data sets? Is SYSIGGV2 your most feared ENQ? Act NOW and unclog your systems with Record Level Sharing for Catalogs available in z/OS V2R1! With this new release, Parallel Sysplex exploiters can optionally enable catalogs to use VSAM Record Level Sharing (RLS) as the access method of choice.

VSAM RLS provides record level locking and cross system sharing capabilities to help remove the need for the dreaded SYSIGGV2 global ENQ. Along with the improved serialization, RLS provides for a large global cache and 64 bit buffering, allowing your catalog requests to complete faster and with less CPU time. In addition to the new RLS support, z/OS V2R1 provides increased catalog availability and integrity with new sysplex wide commands to temporarily suspend or lock catalog requests when required.

This article guides you through the configuration requirements to enable your catalogs for RLS and discuss the potential performance benefits. Armed with this information, you should have all the plumbing power you need to unclog and get the flow going with Record Level Sharing for Catalogs.

Trickling workloads

One reason your workloads are trickling through might be because of contention with the SYSIGGV2 “BCS” reserve/enqueue. The “BCS” resource (majorname=SYSIGGV2 minorname=catalogname) must be obtained exclusively by the Catalog Address Space (CAS) for every update request. Once held, all readers and all other updaters to the catalog must wait until that update releases the resource. This problem is most common during a batch window where multiple define jobs and/or delete jobs are running at the same time, updating the same catalogs.

One widely adapted solution to minimize such contention is to split a catalog into several smaller ones and implement multi-level alias to reference each one of them. But this approach might not be practical and might only reduce contention temporarily as these smaller catalogs become bigger over time. Also, you must take the catalog you’re splitting offline, making it unavailable for general use. An operation like this can be disruptive to continuous 24x7 availability and should always be avoided if possible.

Another reason, though less obvious than the first one, is that CAS only supports limited catalog buffering and buffer invalidation with the current design of both the In Storage Cache (ISC) and the Virtual Lookaside Facility (VLF) buffering techniques. When a catalog in ISC is shared by multiple systems, any update by one system invalidates all the records in ISC on all other sharing systems.

When a catalog in VLF is shared by multiple systems, only the updated records will get invalidated; the rest will remain in VLF. VLF, though a better choice of sharing a catalog than ISC, has one limitation: because it keeps track of all updates from the sharing systems in the catalog’s VVR, VLF can only keep track of a very small number of updates before the information wraps because of the limited space in the VVR. This means that if a sharing system is very behind with updates from the remote systems, VLF would behave like ISC and invalidate all the records in VLF for that catalog in that system.

The VSAM plumbing solution

Beginning with z/OS V2R1, you can share your user and volume catalogs in RLS mode. (Master catalogs are not currently supported with RLS.) Sharing a catalog in RLS mode not only provides more granular locking through the use of a coupling facility (CF) lock structure, it also offers more buffering and caching using the SMSVSAM address space and its associated buffer pools. When a catalog is in RLS mode, each catalog request no longer raises the SYSIGGV2 BCS resource to access the catalog. Instead, SMSVSAM obtains data set and record locks from the CF lock structure on behalf of a catalog. Also, the ISC and VLF functions are not used for any catalog that is in RLS mode; CAS also relies on SMSVSAM to do the buffering and caching.

Each buffer is independently registered with the CF cache structure connected by SMSVSAM from all sharing systems. Any record updates to the buffer from any system trigger the
CF to cross invalidate the copies in all other sharing systems. Once a buffer is invalidated, the most current copy can be read from the CF cache structure, if present, instead of performing an I/O to the DASD.

Furthermore, you’ll no longer need to implement the Enhanced Catalog Sharing (ECS) function for any catalog in RLS mode, because any update to the catalog is no longer tracked in its VVR. Thus, a catalog in RLS mode completely eliminates the use of ISC, VLF, and ECS for that catalog, and the responsibility of locking, buffering, and caching falls onto SMSVSAM’s hands.

It is recommended that you dedicate a separate CF cache structure for any catalogs in RLS mode for better performance isolation. To protect catalog’s integrity, the SYSIGGV2 BCS resource will be held shared by each SMSVSAM address space while a catalog is opened for RLS access. This is done to ensure that applications that explicitly raise the ENQ exclusive because they think they have exclusive access to the catalog can no longer do so.

**Externals**

Before you access your catalog with RLS, ensure the catalog is SMS-managed with a storage class that contains a RLS CACHESET. Next define or alter a catalog with LOG(NONE) to make the catalog eligible for RLS access. When RLS-eligible, specify RLSENABLE to indicate that the catalog is to be accessed by RLS on the next reference. (If you don’t specify RLSENABLE, the default value is RLSQUIESCE, which indicates that the catalog be accessed by non-RLS on the next reference.)

In the example shown below, the catalog is accessed with existing ISC, VLF, or ECS functions that are set up for this catalog:

```sql
DEFINE USRCATALOG(NAME(ucatname) =
                    LOG(NONE) =
                    RLSQUIESCE/RLSENABLE
ALTER ucatname =
                  LOG(NONE) =
                  RLSQUIESCE/RLSENABLE
```

We’re introducing four new modify commands to assist the switch between RLS and non-RLS access:

```sql
F CATALOG,RLSENABLE(ucatname*)
F CATALOG,RLSQUIESCE(ucatname*)
F CATALOG,RLSQUIESCE,SYSTEM
F CATALOG,RLSENABLE,SYSTEM
```

- The first two commands work the same way as the ALTER command. Users can optionally specify the “wildcard” * to switch a group of user catalogs and volume catalogs between RLS and non-RLS access.
- The latter two commands are SYSTEM commands, and when issued, apply to all RLS-eligible user catalogs and volume catalogs accessed by the system. These commands switch those catalogs between RLS and non-RLS access across the sysplex.

The interfaces mentioned above enable your catalogs to enter RLS mode and fall back to non-RLS mode only during an emergency. They are not intended to be used to switch a catalog back and forth between RLS and non-RLS on a regular basis, as a catalog in RLS mode can really stay in RLS mode as long as you want!

**Bonus features**

We are also introducing two bonus features to assist the recovery of both RLS and non-RLS catalogs:

- The ability to LOCK and SUSPEND a catalog with a serialized close across the sysplex. While LOCK invokes a sysplex-wide close of the catalog and fails unauthorized requests, SUSPEND invokes a sysplex-wide close of the catalog but suspends unauthorized requests in the client’s address space instead of failing them.
- The ability to retain all the alias information across a catalog delete by specifying NODISCONNECT on the delete and RECONNECT on the redefine.

Together, these two features ensure the integrity of a catalog during the recovery.

**Benchmarks for RLS**

We performed several test scenarios to test the effects of placing a catalog in RLS mode versus non-RLS mode. The performance improvements we found are based on internal IBM laboratory tests, and your results will vary. But these performance comparisons all show that there is a huge potential performance improvement for putting your catalogs in RLS mode!

- In one update scenario where 300,000 non-VSAM data sets are deleted across 3 LPARs, each running 1,000 jobs, RLS outperforms non-RLS by almost 90% in elapsed time and almost 80% reduction in CPU time. This means that what used to take 80.42 minutes in elapsed time and 1269.3 seconds in CPU time in non-RLS mode now takes only 8.42 minutes in elapsed time and 298.7 seconds in CPU time in RLS mode.
- In another update scenario where 300,000 non-VSAM data sets are deleted across 3 LPARs, each running 1,000 jobs, RLS outperforms non-RLS by almost 60% in elapsed time and around 80% reduction in CPU time.
- In one read scenario where 300,000 non-VSAM data sets are deleted across 3 LPARs, each running 1,000 jobs, RLS outperforms non-RLS by almost 30% in elapsed time for direct read via LISTCAT of all 300,000 data sets.
- In another read scenario, RLS outperforms non-RLS by almost 25% in elapsed time for direct read through TSO ALLOCATE for all 300,000 data sets.

Try the master RLS plumber for help with your workloads, and see the results for yourself!
For this edition, I got an assist from Eric Harris and Ryan Wisniewski on our Service Team. Thanks for the help, guys!

In this edition of Ask Mr. Catalog, we’ll cover these topics:

- The new features coming up for z/OS V2R1
- New, or newer, diagnostic features, including new messages
- Lastly, a few bits of wisdom about how to properly care for your Catalog environment.

**z/OS V2R1 new features:**

**RLS for catalogs**

Users can now access individual user and tape volume catalogs with VSAM RLS in a parallel sysplex. This change improves performance, integrity, and availability. You can use the new sysplex wide commands to control access to individual user catalogs. For detailed information, see “How to unclog your systems with Record Level Sharing for Catalogs in z/OS 2R1” on page 34.

**GDG order enhancement**

In previous releases, when a GDG name was specified on a DD statement, the GDSs were concatenated from newest to oldest (LIFO). This new feature of z/OS V2R1 allows you to specify either LIFO or FIFO concatenation order, using a DEFINE or ALTER command for the GDG. To take advantage of this feature, just set new Catalog parmlib option GDGFIFOENABLE. Note also the new JCL key word: GDGORDER=USECATLG | LIFO | FIFO. USECATLG is the default and specifies to use the GDG setting, LIFO|FIFO can be used to override the GDG setting.

**New catalog search interface (CSI) fields**

In response to requests for additional CSI fields, in z/OS V2R1, CSI can now return fields ASSOC, ASSOCSYB, BUFND, BUFNI, HILVLRBA, INDXLVLS, SEQSTRBA, STRNO, and TRACKS.

**Contention management enhancement**

In z/OS R1.12, Catalog introduced a function to detect SYSZTIOT contention. We expanded this function for z/OS R2.1 to also monitor the SYSZVVDS, SYSIGGV2 and ALLOCLCK resources:

- SYSZVVDS is used for the serialization for the VVDS of a volume.
- SYSIGGV2 provides the essential mechanism to facilitate cross-system sharing of catalogs.

- ALLOCLCK is an internal CAS lock to protect allocations, deallocations, opens and closes.

We’ve also added two modify commands:

- F CATALOG,CONTENTION reports the current threshold wait times and associated actions with contention.
- F CATALOG,CONTENTION(resource,wait_threshold,action) sets both the wait time and the action for a resource (SYSZTIOT, SYSZVVDS, SYSIGGV2, and ALLOCLCK).

The F CATALOG,REPORT command no longer displays the wait time for SYSZTIOT. This information is now contained in the new report output for the F CATALOG,CONTENTION command.
**Table 1. New parameters in the IGGCATxx parmlib member**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default Value</th>
<th>Valid values</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALIASLEVEL(n)</td>
<td>Alias name qualification level</td>
<td>1</td>
<td>1–4</td>
</tr>
<tr>
<td>AUTOADD(ON</td>
<td>OFF)</td>
<td>Specifies whether ECS Autoadd is to be enabled or disabled</td>
<td>Off</td>
</tr>
<tr>
<td>CATMAX(n)</td>
<td>The maximum number of catalogs that can be open at any time.</td>
<td>1024</td>
<td>1–9999</td>
</tr>
<tr>
<td>DUMP(ON</td>
<td>OFF)</td>
<td>Specifies whether dynamic dumping is to be made available or not</td>
<td>OFF</td>
</tr>
<tr>
<td>DUMPON (rc,rsn,mod,count)</td>
<td>Specifies the Catalog return code, reason code, and 2-character Catalog module ID to take a dump</td>
<td>No defaults for RC, RSN, or MOD. Count default is 1</td>
<td>Any combination of return or reason codes and modules</td>
</tr>
<tr>
<td>SYS%(ON</td>
<td>OFF)</td>
<td>Specifies whether SYS% to SYS1 conversion is allowed</td>
<td>OFF</td>
</tr>
<tr>
<td>TAPEHLQ(string)</td>
<td>High-level qualifier for a volume catalog</td>
<td>SYS1</td>
<td>String can contain alphabets, numbers, or national characters (@,$,#)</td>
</tr>
<tr>
<td>TASKTABLESIZE(n)</td>
<td>The maximum number of service tasks that can run at any time</td>
<td>200</td>
<td>200–400</td>
</tr>
<tr>
<td>TASKMIN</td>
<td>The low limit on the number of Catalog services tasks that can run at any given time</td>
<td>60</td>
<td>24–180</td>
</tr>
<tr>
<td>RESOURCENAME (delay,action)</td>
<td>Specifies the resource name, delay and action to be taken for contention detection</td>
<td>None</td>
<td>Resource name: -SYSZTIOT -SYSZVVDS -SYSIGGV2 -ALLOCLCK Delay: 0 -65535 Action: N (Notification) and R (Redrive)</td>
</tr>
</tbody>
</table>

**Catalog parmlib enhancements**

We enhanced the IGGCATxx catalog parmlib member that contains the values that you want loaded when you IPL or restart the system. The settings in this member override equivalent parameters in the SYSCATLG or LOADxx members. The new parameters include those shown in the following table.

**Interface enhancements**

We’ve made numerous Catalog interface enhancements, for example:

- The catalog latch number was added to the F CATALOG LIST command output.
- The date and time from export data set that is used by ICFRU is included.
- The return and reason code that was returned to HSM was corrected. See the PTF for APAR OA23698.
- The message IEC363D IS THIS RESTART RELATED TO AN EXISTING CATALOG PROBLEM (Y OR N)? was enhanced, and the following IEC364D message to accept YES and NO, in addition to Y and N.
- Support was added for Async Events Task to the F CATALOG ABEND command
- The STGADMIN.IGG.DEFINE.RECAT and STGADMIN.IGG.DEFDEL.UALIAS.FACILITY classes were changed from LOG=None to LOG=NOFAIL for logging and auditing purposes.
- A new IBM Health Checker for z/OS check was added that looks at Global Resource Serialization (GRS) Resource Name Lists (RNL) to ensure they match IBM recommendations for SYSIGGV2, SYSZVVDS, and SYSVTOC.
- A creation date to the alias entry was added.
• Association verification during data set deletion was added to ensure associations are not lost.
• The restriction for using SYMBOLICREATE to the same catalog on a DEFINE ALIAS AMS service was removed.

New(er) diagnostic features
System settings, vendor products, and other varying variables can make diagnosing and recreating some problems difficult. Here are a few of the recent APARs that describe the diagnostic information provided the system provides:

APAR OA36585
The Catalog Auxiliary Work Area (CAXWA), an area that contains important information about catalogs on the system, is contained in common storage and thus is susceptible to overlays. APAR OA36585 adds a function that runs every 30 seconds to check the integrity of the work area. If a problem occurs, the system tries to correct the issue. If it can't, it issues message IEC366E to tell the user that an IPL is necessary to correct the CAXWA issue. This function can help prevent further errors and provides documentation that helps resolve the problem.

APAR OA38220
The system issues message IDC3009I RC50 RSN255 when the total number of current CI count is incorrect. The PTF for this APAR adds function to correct the value if possible, and to write a symptom record with more diagnostic information.

Catalog best practices
Here are a few tips to ensure that your catalogs are in the best possible shape and won't let you down by going down.

A healthy catalog environment is a happy catalog environment
• One of the easiest ways to avoid catalog-related downtime is to keep current on maintenance. Although PTFs for HIPER APARs are important, they shouldn't be the only PTFs applied!
• If a catalog goes down, having a proper backup will reduce the downtime. When backing up a catalog, remember these simple instructions:
  • ALWAYS use AMS commands DIAGNOSE and EXAMINE on the catalog before backing it up.
  • NEVER store a backup copy on the same volume as the catalog.

You can use the new sysplex wide commands to control access to individual user catalogs.

• ALWAYS quiesce activity to the catalog and lock the catalog before backing it up.
• ALWAYS collect SMF 60–66 records for forward recovery.

Measure twice, DEFINE USRCATALOG once
When it's time to build a new catalog, ensure that you define it correctly:

• Set CISIZE to 4096—this gives you the right amount of VSAM I/O performance while avoiding costly CI splits or spanned records.
• Always allocate primary and secondary amounts in CYLINDERS.
• ALWAYS define a large sized secondary allocation amount—the number of allowed extents are finite!
• BUFND, BUFNI, and STRNO settings are important for system performance:
  – BUFND should be STRNO +1, which is the default.
  – BUFNI should be the entire index set, plus a sequence set control interval for each string.
  – STRNO should be at least 3.
• Do not specify BUFFERSPACE. It is the minimum buffer size and applies to the whole cluster.
Time for a reorg
If a catalog has too many extents, reorganize it to consolidate space. The recommended process includes these steps: EXPORT, DELETE, DEFINE, IMPORT. A reorg can be used as a basis for moving, expanding, splitting, or merging catalogs, too. Be sure to perform every step of the processes listed in z/OS V2R1 DFSMS Managing Catalogs, SC23-6853. And if your data set names contain date or time stamps, consider using CA reclaim processing to reduce unused CAs. CA reclaim processing can reduce the need to extend a catalog.

Sharing is caring
If you want to share catalogs and you are not using RLS for Catalogs, make sure that the catalog is defined with SHAREOPTIONS (3 4) and are defined on a UCB marked as shared. VVDS mode is the default sharing method. If a coupling facility is available, ECS sharing is recommended for improved system performance; you must specify ECSHARING for ECS catalogs. For serialization considerations and GRS configuration tips on sharing catalogs, please read (memorizing wouldn't hurt either) information in APAR III14297. See “How to unlog your systems with Record Level Sharing for Catalogs in z/OS 2R1” on page 34 for more information.

It’s always in the last place you look
Using aliases is an art, and properly planned conventions can eliminate headaches later. Catalog allows both single or multi-level aliases. Multi-level aliases can provide more granularity in catalog searches. For example, if you have two sets of data sets, you could use aliaslevel=1, such as STUFF.PROD and STUFF.TEST to direct them to different catalogs. In either single or multi-level environments, avoid short catalogs names where #HLQs = aliaslevel. Catalog names are automatically treated as aliases!

General no-no’s
• Do not vary a volume offline on only one system and reinitialize or restore it. Other systems will not know if it. Vary offline on all systems.
• Do not delete aliases while jobs are running. This will lead to failed extends and DSCB/Catalog mismatches.
• Never delete the index (.CATINDEX) component of a catalog.
• Do not import (by using the IMPORT command) a catalog without the LOCK function.

IBM System z Security Portal
As the best practice, IBM strongly recommends that you obtain access to the IBM System z Security Portal. Subscribe to the automatic notification process to get access to the latest service information on security and system integrity APARs for z/OS and z/VM®. IBM treats vulnerability information in connection with System z as IBM confidential. By accessing the Security Portal, you agree to treat such information as confidential in accordance with the terms set forth. Visit:

ibm.com/systems/z/advantages/security/integrity_sub.html
Looking for new ways to improve system performance? Do you develop or run applications that use large amounts of data? System programmers and application developers, please read on to discover how large pages can help you get a performance boost. Application developers, you will want to know how to choose the right page size for your application and how to code the request. System programmers, you will want to know how to configure the system for large pages and make tuning adjustments. But first, let’s start with an overview of the performance benefits.

Why are large pages faster?
A little history: In the early 1970s, IBM added dynamic address translation (DAT) to the System/370 architecture. Initially, a segment table and page table scheme was introduced to translate addresses, using either 2 KB pages (eventually dropped from the architecture) or 4 KB pages. In 2008, IBM introduced its first large page when it added a 1 MB page size to the z/Architecture® with the enhanced-DAT facility. Dynamic address translation for a 1 MB page reference does not require a page table. The segment table entry that would normally point to a page table instead points to the 1 MB page directly. The translation lookaside buffer (TLB) coverage is also effectively increased, because one entry for a fixed 1 MB page provides a greater coverage area than one entry for a 4 KB page. z/OS was enhanced to support fixed 1 MB pages, which reside in the large frame area (LFAREA) and which can result in these benefits when 1 MB pages are used by exploiting programs:

- The translation requires less storage (one less page table for each 1 MB page).
- The translation is faster because the steps using the page table are eliminated, and the number of potential real memory read operations is reduced.
- The increased coverage of the TLB will help reduce TLB misses.

Fast-forward to today: The IBM zEnterprise EC12 (zEC12) provides two new enhancements: FLASH storage capability and the enhanced-DAT facility 2 for a new 2 GB page size. In turn, z/OS is enhanced to provide two new page types: pageable 1M pages and fixed 2 GB pages. FLASH will be used for paging the pageable 1M pages (as well as 4 KB pages, in some cases). The fixed 2 GB pages will reside in the LFAREA with the fixed 1 MB pages.

Pageable 1 MB pages inherit the same DAT performance benefits as fixed 1 MB pages with the additional feature of being pageable to FLASH storage.

Fixed 2 GB pages provide an even greater DAT performance benefit because the translation does not require a segment table or its page tables. Instead, the region third table entry points directly to the fixed 2 GB page instead of a segment table. This means less storage for DAT structures, faster translation, and increased TLB coverage because one entry for a fixed 2 GB page provides an even greater coverage area than one entry for a 4 KB page.

So, why not use fixed 2 GB pages all the time if they are so fast? Read on...
How does an application developer choose the right page size?

Table 1 provides guidance for the application developer. In an ideal world, your data area will fully occupy the large pages you obtain and your references will be evenly distributed across the entire area. In practice, not every case will be as clear cut. The important thing is to consider the tradeoffs in terms of performance and storage resources, with a goal toward achieving the best overall balance for the system.

<table>
<thead>
<tr>
<th>Page frame size</th>
<th>Best uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 KB</td>
<td>Applications with a small amount of data, or any amount of data with low temporal locality</td>
</tr>
<tr>
<td>Fixed 1 MB</td>
<td>Long-running authorized applications with a large amount of data with high temporal locality that needs to be in fixed storage</td>
</tr>
<tr>
<td>Pageable 1 MB</td>
<td>Applications with a large amount of data with high temporal or sequential locality that can be paged by the system as needed</td>
</tr>
<tr>
<td>Fixed 2 GB</td>
<td>Long-running authorized applications with very large amounts of data with high temporal locality that needs to be in fixed storage</td>
</tr>
</tbody>
</table>

Table 1. A guide to page frame sizes

How does an application developer request large pages?

Table 2 shows the available z/OS services that support large page requests.

<table>
<thead>
<tr>
<th>z/OS services to acquire</th>
<th>Page frame size</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPOOL BUILD</td>
<td>Pageable 1 MB</td>
</tr>
<tr>
<td>DSPSERV CREATE</td>
<td>Pageable 1 MB</td>
</tr>
<tr>
<td>STORAGE OBTAIN</td>
<td>Pageable 1 MB</td>
</tr>
<tr>
<td>IARV64 GETSTOR</td>
<td>Fixed 1 MB, pageable 1 MB, fixed 2 GB</td>
</tr>
<tr>
<td>IARV64 GETCOMMON</td>
<td>Fixed 1 MB, pageable 1 MB</td>
</tr>
</tbody>
</table>

Table 2. z/OS Services for large pages

We enhanced IARV64 syntax to make it easier to request large pages. For example, to allocate 4 GB of 64-bit private storage backed by fixed 2 GB pages, you can code:

```
IARV64 REQUEST=GETSTOR,UNITS=2,UNITSIZE=2G,
PAGEFRAMESIZE=2G,TYPE=FIXED,...
```

When using z/OS services (PGSER FIX, for example) on a pageable 1 MB page, you must act on the entire 1 MB page to prevent it from being demoted permanently into 4 KB pages. For more information about 1 MB demotion, refer to the publications listed at the end of this article.

Applications that run in different environments can check CVTEDAT and CVTEDAT2 to determine whether fixed 1 MB pages and fixed 2 GB pages are supported, respectively, before making a request.

Application developers should document their fixed large-page requirements so that system programmers will be able to factor in the number of fixed large pages to reserve in their large frame area (LFAREA).

How does the system programmer configure large pages?

Each of the large page types has its own set of hardware and software requirements, as shown in Table 3.

<table>
<thead>
<tr>
<th>Page frame size</th>
<th>Hardware requirements</th>
<th>Software requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed 1 MB</td>
<td>Minimum: z/10 server</td>
<td>Specify LFAREA requesting 1M MB pages</td>
</tr>
<tr>
<td>Pageable 1 MB¹</td>
<td>Minimum: zEC12 server</td>
<td>RSM Enablement Offering²</td>
</tr>
<tr>
<td></td>
<td>Flash Express Card</td>
<td>Specify PAGESCM enabling 1 MB pages</td>
</tr>
<tr>
<td>Fixed 2 GB</td>
<td>Minimum: zEC12 server</td>
<td>RSM Enablement Offering²</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PTF UA68170³</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify LFAREA requesting 2 GB pages</td>
</tr>
</tbody>
</table>

Table 3. Large page requirements

Notes:

1. z/OS automatically determines the size of the pageable 1M area (PLAREA).
2. Download this offering from the z/OS website (ibm.com/systems/z/os/zos/downloads). See “RSM Enablement Offering for Z/OS R13”.
3. PTF UA68170 is for the 2G Enablement APAR (OA40967).
We enhanced the IEASYSxx LFAREA parameter to support reserving fixed 2 GB pages. You can also now request target and minimum values and percentages for fixed 1 MB and 2 GB pages. Several new and changed IARxxx messages are provided to assist with problem determination.

Configuring the system properly involves selecting the correct number of 1 MB and 2 GB pages based on the requirements from various applications that use large pages. This should be balanced with the overall system storage requirements, such as 4 KB and pageable 1 MB page usage. If constrained, the system will satisfy 4 KB and pageable 1 MB pages out of the LFAREA. Periodically fine-tuning your system will help maintain an efficient use of storage.

How does the system programmer tune large pages?

Using monitoring functions such as those available in Resource Measurement Facility (RMF) is your best bet for obtaining useful information for tuning your large pages. A quick and useful alternative is the DISPLAY VIRTSTOR,LFAREA system command, which shows the current fixed large-page usage of a system, as well as the high-water marks.

Tuning essentially boils down to some combination of adjusting the LFAREA specification, adjusting the various workloads that use large pages and 4 KB pages, and adjusting the amount of real storage. See Figure 1 for an example.

Referring to Figure 1, here are a few tips to consider:

- To determine whether you have the right 1M LFAREA size defined, check the data reported in lines 4, 8, 9, and 11. Lines 9 and 11 indicate, respectively, the maximum amount of LFAREA used to satisfy 4 KB and pageable 1 MB page requests when the system was constrained.
- To determine whether you have the right 2G LFAREA size defined, check the data reported in lines 4 and 13. Unlike fixed 1 MB pages, fixed 2 GB pages are not used to satisfy any other type of page request.
- Review the 4 KB high-water mark (line 9) to determine whether you need to decrease your 1M LFAREA total, 2 GB LFAREA total, or both, reduce your 4 KB workload, or add more memory to the system.
- Review the PAGEABLE1M high-water mark (line 11) in conjunction with your 1M LFAREA total (line 4) to determine whether you need to increase your 1M LFAREA total. Note: You cannot increase the number of pageable 1 MB pages defined to your system. You can, however, increase the number of fixed 1 MB pages defined to your system that can be used to satisfy pageable 1 MB page requests.

You should now understand how large pages can provide a performance kick. There are many things to consider in choosing the right page size and in configuring and tuning the system.

References

We encourage you to further explore the topic of large pages in the following publications:

- z/OS MVS Programming: Authorized Assembler Services, Volumes 1 to 5 and z/OS MVS Programming: Assembler Services, Volumes 1 and 2 contain information about using z/OS services that pertain to large pages.
- z/OS MVS Initialization and Tuning Reference, SA23-1380, and z/OS MVS Initialization and Tuning Guide, SA23-1379, contain information about the LFAREA and PAGESCM SYSPARM keywords.
- z/OS MVS System Messages, Volume 6 (GOS-IEA), SA38-0673, contains information about messages related to the LFAREA.
Problems that occur in today’s systems are often difficult to diagnose. And who has the time or the resources to watch the z/OS console every second of every day trying to catch small problems early enough to stop them from becoming disastrous? That is where IBM System z Advanced Workload Analysis Reporter (IBM zAware) can help. IBM zAware uses z/OS message traffic (operlog) sent by the System Logger to quickly show you if something out of the norm is occurring in near-real time.

In this article we’ll explore how to get the data to zAware, load the data, analytics and real time analysis behind zAware, automated learning and training sets, and a typical problem diagnosis using the zAware interface.

Getting the data to z/Aware

By default IBM zAware uses 90 days of operlog data to build a model of an image’s activity.

There are two methods by which this can be accomplished:

1. Configure the z/OS client and IBM zAware server, connect the client, and then wait 90 days. When IBM zAware has collected data for 90 days, it builds a model and begins producing analysis data. Simple, except for that part about waiting 90 or more days.

Many people do not want to wait 90 days; one has a new toy and wants to use it. Which brings us to the next method:

2. Use the IBM zAware bulkloader process to load 90 days of syslog data through a batch job.

Option 2 has the advantage of taking just a day or so to get all the syslog data for all the images of a sysplex loaded into IBM zAware versus a 90-day wait.

The bulkloader is REXX EXEC (found in SYS1.SAMPLIB) that is supplied by IBM. It takes a list of data sets in SYSLOG format and sends them to the IBM zAware server as part of the same process that the z/OS client uses to send real-time OPERLOG data to IBM zAware. This function can be run from any z/OS system that is configured to connect to IBM zAware.

The bulkloader builds a log stream under the HLQ specified within the bulkloader JCL, and then begins to send the data to the IBM zAware server through IXGLOGR, the MVS log stream. There are two default limits defined within the EXEC that you should note:

- The size of the individual syslog or operlog data sets that are being sent to the IBM zAware server. The default is 250 MB and this works just fine, but performance can be improved if this value is increased.
The downside of increasing this value is a requirement for more private storage for either the TSO/E ID or batch job that is running this EXEC.

- The total number of bytes that can be sent in a single invocation of the EXEC. The default is 1 GB. This might be too low, especially if the syslog for a large sysplex is being transmitted.

Changing these defaults requires that you edit the bulkloader REXX EXEC EXEC. The symbolic that manages the data set size is g.Chunk and the one that limits the total number of bytes is g.MaxImportBytes.

Bulk load as much recent data as possible, keeping in mind that data older than your training period will not be used to build the model. The default training period is 90 days, but you should ensure that this period spans important dates in your business cycle.

**Loading data**

When the syslog (bulk-load) or operlog (real-time) data is received by IBM zAware, it is parsed to break out fields like the time stamp, system name, message ID, and message text, which are fed to the Analytics to summarize and store in an internal database. This summary greatly reduces the data volume by keeping only one example of each message ID’s text, a text summary and appearance statistics within each 10-minute interval. The parsing recognizes any well-formed message ID, including non-IBM products and customer applications.

Bulk loaded logs contain the system names but not the sysplex name, so IBM zAware summarizes and stores the system names without a sysplex name. After you send all of this historical data to IBM zAware, you must assign it to the correct sysplex through the Priming page. You should assign all the priming data at once because IBM zAware must automatically disconnect all monitored systems while this assignment is done. Don’t worry, the data is buffered and the reconnection happens automatically, too.

**Analytics**

At the heart of IBM zAware is the analytics, invented by IBM Research, which has been refined and tailored for z/OS message traffic.

**Building a model of normal behavior**

When you have enough days of data loaded into IBM zAware, it’s time to use this data to build a model of normal behavior. This action is called training, and it occurs when the machine-learning algorithm analyzes the data. During the training process, IBM zAware examines all the message occurrences for one monitored system, and groups together message IDs that tend to occur at about the same time. These clusters of message IDs are stored into a model that is specific to that system’s message behavior.

If you are patient, the initial training will be done automatically by IBM zAware, based on the earliest day you’ve loaded and your training period. But most of you will manually request the first training on a system as soon as you’ve sent priming data and assigned it to the sysplex. That request is done on the Training Sets page. The training process is an expensive operation (in terms of processor and memory use), so we queue up these requests and run one at a time to avoid resource contention.

**Real-time analysis**

After a model is built, and a system is connected to IBM zAware, the current operlog data is run through the analytics process to compare the data against the system model. This analysis works on 10-minute intervals, and scores each message ID for how anomalous it appears. The message anomaly score is based on a combination of factors:

- Whether the message was clustered with others
- Whether the message occurred within its normal cluster
- Whether the number of appearances was normal
- The rarity of the message
- An adjustment of the score based on specific message-ID rules. The information is based on z/OS knowledge of messages that range from critical to “non-interesting,” that is, messages that probably don’t pose a problem.

The message anomaly scores are combined into an interval anomaly score for the 10-minute interval, which is rendered in a bar chart on the GUI Analysis page. The colors on the chart range from light blue (consistent with the model) to orange (significantly different). The bar height shows the number of unique message IDs that appear in the interval.

“Wait!” you say... “10 minutes is too long to wait for IBM zAware to indicate an anomaly that is due to a system problem!”

We agree! For the current interval, we use a sliding window of the last 10 minutes to update the current interval scores every 2 minutes. So, we’ll give you near real-time results to get a jump on diagnosing an anomaly. Keep in mind that a high anomaly score is not necessarily a system problem, but it is worth looking into to possibly head off a cascade of related issues.

**Automated learning and training sets**

In addition to analyzing the live data stream, the summarized data is also stored in the database, so it can be used for future models. By default, IBM zAware automatically initiates a training request to build a new model for each system every 30 days, using the last 90 days of data.
You can change the frequency of the automatic training, and the number of days to include in the Analytics Configuration tab. You can also exclude specific days. (You know, that day when a system gave you a pounding headache.) Excluding these days helps ensure that they are not used in the future model of expected behavior (see Figure 1).

In the latest IBM zAware update, you can mark specific message IDs to be ignored in the data analysis. This is useful when you made workload changes or upgraded software levels on a z/OS system. The new message traffic is likely to generate high anomaly scores. After you determine that the contributing messages are the new “normal,” and that they don’t indicate something such as a configuration glitch, you should mark those messages as ignored. Ignored message IDs do not contribute to the anomaly scores for subsequent analysis intervals. You can set certain options to ignore a message ID until the next training (when it will be part of the new normal), or indefinitely. Ignored message IDs can also be specified before they are produced, for example, when you are planning to deploy new software on a system.

“Wait!” you say... “10 minutes is too long to wait for IBM zAware to indicate an anomaly that is due to a system problem!”

**Diagnosis with IBM zAware.**

The Analysis panel is the primary interface to IBM zAware. It shows the status of all monitored sysplexes in an easy-to-understand graphical display. The display shows the monitored systems in a horizontal timeline with vertical bars for each 10-minute interval of the day.

The vertical bars provide two pieces of information:

- Rate of unique syslog messages, indicated by the height of the bar
- Calculated health of the system during that interval, indicated by the color of the bar. Health means how well the traffic matches the model.

A light blue bar indicates all is well. A dark blue bar is an area of concern to be examined. A yellow bar indicates a potentially serious problem, and a red bar indicates that a very serious anomaly occurred, possibly indicating a system problem.

In the example in Figure 2, system CB8A is showing continuous dark blue intervals since the beginning of the day. Placing the cursor over one of the bars shows the time frame for that bar, the number of unique messages, and the anomaly score. The value of the anomaly score is what determines the color of the interval bar.
Left clicking on the interval bar shows the Interval View for that image for that period (see Figure 3). In the preceding case, from 3:10–3:20 a.m. UTC, 117 unique messages were issued for an anomaly score of 99.4. The Interval View displays each unique message that was issued during the interval, a count of each message's occurrence, part of the message itself, plus the message's anomaly score. There is also a timeline for the interval and a green vertical bar that shows the timeline for which the message was issued.

In the preceding example, from the Interval View, we can see that message IST264I occurred three times with an anomaly score of 1. We see that the text of the message is that VTAM resource CBT8A is not active. From experience we know that CBT8A is the resource for System Automation for z/OS.

Starting the address space that is associated with CBT8A resolves the issue, which can be seen in the color of the interval bar following the 3:10–3:20 a.m. interval.

More details

Like what you see? IBM zAware is available today as a feature on the IBM zEnterprise EC12 or zEnterprise BC12 server. Each z/OS system that is being monitored must be running z/OS V1R13 or later with appropriate service applied for System Logger and the IBM zAware bulkloader.

For more information about IBM zAware, see these resources:

- System z Advanced Workload Analysis Reporter (IBM zAware) Guide, SC27-2623
- Extending z/OS System Management Functions with IBM zAware, SG24-8070
- Preparing for z/OS IBM zAware log stream client usage, MVS Setting Up a Sysplex, SA22-7625

Figure 3. Interval View for the CBT8A system for 3:10–3:20 a.m. UTC

zFavorites

Go to the zFavorites website where you will find the latest information for all of your System z needs including product documentation, software, ISV development marketing info, education, links to FREE downloads, and much much more!

ibm.com/servers/eserver/zseries/zos/zfavorites/
Professor Kimura here, with important tips on migrating to z/OS V2R1 for DFSMS and Non-VSAM, System REXX, along with other important message changes for DFSMS and JES2.

**Enhancement in z/OS V2R1 DFSMS**

**New free space information in IDCAMS DCOLLECT command output**

Before V2R1 when you ran the LISTVTOC command using the IEHLIST utility program, the free space information was reported in the SYSPRINT data set for each empty cylinder and empty track. On the other hand, if you specified the IDCAMS DCOLLECT command with volume information, it used the LSPACE macro to collect and calculate the amount of free space, and combined information from two fields (LSPDTCYL and LSPDTTRK) into one field - DCVFRESP in the DCOLLECT “V” record.

Now, in z/OS V2R1 DFSMS, to offer more granular information about free space, we’ve added two new 4-byte fields to the DCOLLECT “V” record (see Figure 1):

- Cylinder freespace on a volume (DCVFCYLS)
- Track freespace on a volume (DCVFTRKS)

**Volume information (Record type “V”)**
 Offset Type Length Name Description
148 (X’94’) Signed 4 DCVFCYLS Free cylinders on volume
152 (X’98’) Signed 4 DCVFTRKS Free tracks on volume

**Figure 1. Output record structure for DCOLLECT “V”**

Figure 2 shows the output from the IEHLIST LISTVTOC command for volume SYSWKA on the DVCFCYLS and DCVFTRKS offsets. The offsets indicate that on this volume there are 4295 empty cylinders (field DCVFCYLS contains X’10C7’, which is equal to 4295) plus 1741 empty tracks (field DCVFTRKS contains X’6CD’, which is equal to 1741). In total, there are 4295 empty cylinders and 1741 empty tracks on this volume.

**Enhancement in z/OS V1R13 for Non-VSAM data sets**

**New EXPIRATION_MESSAGE keyword in parmlib member DEVSUPxx**

On a pre-z/OS V1R13 system, when you issue the OPEN macro for a non-expired, non-VSAM data set with write access, the system issues WTOR message IEC507D to get verification from the system operator. (The system also issues associated message IEC108I OPERATOR ACTION HAS BEEN REQUESTED FOR YOUR DATA SET.)

DFSMSdfp issues message IEC507D if the format 1 DSCB (DASD VTOC) of an already existing data set indicates that the data set has not expired.

Let’s take a look at the following cases:

- **Case-1:** When the expiration date for the data set has not expired, the system issues the following messages:

\*IEC507D E D7E7,SYSWK2,BEANSZZ,STEP1,BEANS.ZOS.HOT.TOPICS.SAMPLE1
\*02 IEC507D REPLY ‘U’-USE OR ‘M’-UNLOAD

- **Case-2:** the data set has a never-expire date of 99365 or 99366. (The “NEVEREXPIRE” indication was added in z/OS V1R11 to support message automation in order to distinguish between these two cases.) In this case the system issues the following messages:

\*IEC507D E D7E7,SYSWK2,BEANSZZ,STEP1,BEANS.ZOS.HOT.TOPICS.SAMPLE2,NEVEREXPIRE
\*03 IEC507D REPLY ‘U’-USE OR ‘M’-UNLOAD
Now, with z/OS V1R13 introduced the new EXPIRATION_MESSAGE parameter in the DEVSUPxx parmlib member that allows you to specify the options NEVER or ALWAYS for the keyword.

Specify NEVER if you want OPEN to disable expiration date processing when opening all non-VSAM data sets on DASD for output processing.

- Specify NEVER to eliminate these messages for all non-VSAM data sets on DASD. Note that this keyword has no effect for data sets on magnetic tape even if you specify NEVER.
- All DEVSUPxx parmlib members processed at IPL.
- In the last (or only) DEVSUPxx parmlib member processed during or after IPL, by specifying system command: SET DEVSUP=xx, as described in DOC APAR OA40242.

**Tip:** By default, the EXPIRATION_MESSAGE parameter is set to the default value ALWAYS before the system processes the current DEVSUPxx parmlib member. That means that if you want the the non-default value of NEVER, you must specify it in either of the following places:

---

**Enhancement in z/OS V2R1 BCP**

**New entries for System REXX in the system’s default program properties table**

The System REXX address space and eight other TSO Server address spaces should run in the SYSSTC service class and should not be explicitly classified to a different service class. You can ensure that the address spaces run in service class SYSSTC by using the recommended PPT "PRIV" attribute in the SCHEDxx parmlib member for both AXRINIT and AXRRXTSS programs.

However, by default in z/OS V2R1, the output of D PPT,NAME=AXR* command now includes entries for the AXRINIT and AXRRXTSS program in the system default program properties table, which explicitly specifies a PRIV attribute (see Figure 3). So, when you completed your migration to z/OS V2R1, there is no need to specify the associated entries in SCHEDxx parmlib member.

**Tip:** By default, the EXPIRATION_MESSAGE parameter is set to the default value ALWAYS before the system processes the current DEVSUPxx parmlib member. That means that if you want the the non-default value of NEVER, you must specify it in either of the following places:

---

**Changes to messages in z/OS V2R1 DFSMS**

**All O/C/OEV ABEND messages now issued as Multi-line WTO**

In z/OS V2R1, all OPEN, CLOSE and EOV ABEND messages are now issued using a Message Buffer Manager service, even if you haven't enabled the new VERBOSE message function using the .MSGOPTION VERBOSE(Y) keyword in parmlib member MPFLSTxx. Now the system issues these messages as MLWTO, which means that the messages now take up at least two lines, each line containing up to 70 characters.

**Note** that most of the first line of these messages is unchanged in z/OS V2R1, except that the data set name now appears on the second line and anything that was after the data set name also appears on the second or subsequent lines.
For example, if you specify a non-existent member name "ZZZZZZZZ" for partitioned data set "BEANS.ZOS.HOT.TOPICS.SAMPLE", the OPEN processing results in ABEND013-18. ABEND message IEC141I 013-18 now displays as a multi-line WTO in z/OS V2R1 (see Figure 4).

z/OS V1R12 SYSLOG (Single-line WTO message)

```
JOB00911 00000090 IEC141I 013-18, IGG0191B, BEANSSZ, STEP1
958 00000090 BEANS.ZOS.HOT.TOPICS.SAMPLE(ZZZZZZZZ
```

z/OS V2R1 SYSLOG (Multi-line WTO message)

```
JOB00911 00000090 IEC141I 013-18, IGG0191B, BEANSSZ, STEP1
958 00000090 BEANS.ZOS.HOT.TOPICS.SAMPLE(ZZZZZZZZ
```

Figure 4. Single and multi-line WTO ABEND message format

Because the message is now issued as a multi-line WTO message, automated operation services that parse these messages might be affected. For example, the contents of the subsequent "D" or "E" lines cannot be evaluated in a NetView message table comparison with a regular NetView trap.

This change was originally part of z/OS V1R13 with PTF UA64502 (DFSMS APAR OA37505). When applying the PTF to z/OS V1R13, the same automation consideration applies.

**Tip:** To address this particular NetView issue, use the ACQUIRE and FINDLINE msg traps, as described in the following Technote: ibm.com/support/docview.wss?uid=swg21253770

Changes to JES messages in z/OS V1R13

**MAXCC** in the JES2 NOTIFY message $HASP165 is now 4 digits and new format of CC display in command response message $HASP890

In z/OS V1R13, you can now request that the return code reported on JES2 notify message $HASP165 be the highest code, as usual, the code on the last step, or the code for a specific step.

There new options are available on the JCL JOB statement JOBRC parameter as well as the JES2 JOBCLASS statement JOBRC parameter. The default is JOBRC=MAXRC, which specifies the same behavior as earlier releases. However, now MAXCC $HASP165 message output is always displayed as 4 digits, regardless of the JOBRC parameter setting (see Figure 5).

z/OS V1R12

```
SE '05.53.20 JOB03405 $HASP165 BEANSSZ ENDED AT N1
MAXCC=0', LOGON, USER=(BEANS)
SE '05.58.54 JOB03407 $HASP165 BEANSSZ ENDED AT N1
MAXCC=8', LOGON, USER=(BEANS)
```

z/OS V1R13

```
SE '14.42.40 JOB00160 $HASP165 BEANSSZ ENDED AT N1
MAXCC=0000', LOGON, USER=(BEANS)
SE '16.15.05 JOB02838 $HASP165 BEANSSZ ENDED AT N1
MAXCC=0008', LOGON, USER=(BEANS)
```

Figure 5. Changes between z/OS V1R12 and z/OS V1R13 for MAXCC output from JES2 NOTIFY message $HASP165

The CC format in the JES2 response message $HASP890 has also changed in z/OS V1R13 for the $DJnnnnn,CC and $DJnnnnn,LONG commands (see Figure 6).

z/OS V1R12

```
$HASP890 CC=(COMPLETED, CODE=0)
$HASP890 CC=(COMPLETED, CODE=8)
```

z/OS V1R13

```
$HASP890 CC=(COMPLETED, RC=0)
$HASP890 CC=(COMPLETED, RC=8)
```

Figure 6. Changes between z/OS V1R12 and z/OS V1R13 for CC format in JES2 $HASP890 message

Note that if you run multiple JES2 levels in a JES2 MAS configuration, and the job executes on a pre-z/OS V1R13 members, the system issues message $HASP165 with the single digit version of MAXCC.

**And here’s more information!**

Be sure to read “Get ready for your next move: Migrating to z/OS V2R1!” on page 49.
Get ready for your next move: Migrating to z/OS V2R1!

BY MARNI WALLACE

It’s finally here! z/OS V2R1. You’ve hopefully been learning about all of the great new functions you can use in z/OS V2R1, but you’ve got to get there first. That’s where I come in. Let me tell you about some of the important migration actions that you should consider now while running z/OS V1R12 or z/OS V1R13. I can’t fit all of the information about migration actions into this one article, so make sure to review z/OS V2R1 Migration, GA32-0889.

Speaking of the z/OS V2R1 Migration book, you’ll notice that we’ve changed the format. We wanted to provide a more customized migration path to users, so we’ve divided the chapters in the book for those on a specific migration path. Chapter 1 contains an introduction to migration for all to use. Chapter 2 contains the migration actions just for those on the z/OS V1R13 to z/OS V2R1 path. Chapter 3 contains the migration actions just for those on the z/OS V1R2 to z/OS V2R1 path, which also includes all of the z/OS V1R13 migration actions.

If you are coming from z/OS V1R13, you need only read Chapters 1 and 2 and Chapter 3. If you are coming from z/OS V1R12, you need only read Chapters 1 and 2 and Chapter 4. We hope that you find this format easier to use and that it gives you a more useful scope when seeing your migration actions! Let us know your impressions.

Use the IBM Health Checker for z/OS… really … right now … seriously

In z/OS V2R1, the IBM Health Checker for z/OS will be started automatically for you at initial program load (IPL). If you are already starting the Health Checker, great! If you don’t start the Health Checker, you might be surprised by how many exceptions you’ll see on the console. In fact, you’ll probably want to do something so you don’t see those messages until you can deal with them. There are several ways you can stop seeing exception messages, but the simplest one that we recommend is to make the messages go to the hardcopy log. You can use this in your HZSPRM00 parmlib member:

```
****** **************************************** Top of Data **
000001 ADDREPLACE POLICY (HCONLY)
000002 UPDATE CHECK (*,*) WTUETYPE (HARDCOPY)
000003 REASON=('Put everything to hardcopy to avoid exceptions on the console.')
000004 DATE=(20130831) /* can use symbolics here to keep the date more recent than any check */
000005 ACTIVATE POLICY (HCONONLY)
****** **************************************** Bottom of Data **
```
Doing this will still allow you to see all of the health checks and their output, but will also eliminate all of the console messages for exceptions.

When z/OS V2R1 starts the Health Checker automatically, it will start with the name HZSPROC by default. If you set up Health Checker with that name now, there'll be slightly less work to do for z/OS V2R1.

**Final removal of zFS multi-file system aggregates. Gone, baby, gone.**

It’s been happening over time, and z/OS V2R1 is the release in which they are finally gone. Yes, multi-file system aggregates — the ability to have more than one file system in an MVS data set — are gone in z/OS V2R1. You should have been using compatibility mode aggregates — just one file system in an MVS data set — for a while now. Associated with the multi-file system aggregate removal are some zfsadmin commands for cloning multi-file system aggregates that have also been removed: zfsadmin clone and zfsadmin clonesys. In addition, you will not be allowed to mount any zFS file system aggregate that contains a cloned (.bak) file system.

**Tip:** Use the ZOSMIGREC_ZFS_RM_MULTIFS health check to see if you have any multi-file system aggregates attached.

**Is your hardware ready? Get modern.**

z/OS V2R1 will only support a z9® server or later. That means if you are running z/OS V1 on a z800, z900, z890, or z990, you will need to move to a z9 (minimally) before migrating to z/OS V2R1. Also, z/OS V2R1 requires storage control units that support the extended count key data (ECKD) command set, so you need one of these storage control units (or later): 3990 Model 3 or Model 6, 9393, 2105, 2107, 2421, 2422, 2423, or 2424.

**z/OS Communications Server and Capacity Provisioning on z/OSMF**

Do you use the IBM Configuration Assistant for z/OS Communications Server on Windows? Do you use the Capacity Provisioning Control Center (CPCC) on Windows? If the answer to either of these questions is yes, you need to move to z/OS Management Facility (z/OSMF). Starting with z/OS V2R1, these applications will be provided on z/OSMF and not as Windows-based applications.

**BPX_DEFAULT.USER...everyone is unique now!**

In the article “Nobody’s default but mine” in z/OS Hot Topics Newsletter Issue 26, August 2012, GA22-7501-22, there's some helpful information about kicking the BPX.DEFAULT.USER habit. z/OS V2R1 is the first release of z/OS where the BPX.DEFAULT.USER profile will be ignored, and you will need to make other accommodations. You can still use the BPX.NEXT.USER profile in z/OS V2R1.

**Tip:** APAR OA37164 gives us a new health check - ZOSMIGV2R1_DEFAULT_UNIX_ID - to see if you are relying on assigning default z/OS UNIX identities for users without OMVS segments.

**Into FIPS 140-2? Then you are also into ICSF.**

Starting with z/OS V2R1, the Federal Information Processing Standard (FIPS) 140-2 support in System Secure Sockets Layer (System SSL) and Application Transparent Transport Layer Security (AT-TLS) require that the Integrated Cryptographic Service Facility (ICSF) is active. If you are using AT-TLS or System SSL configured for FIPS 140-2 support, you must be using ICSF. If you verify that ICSF is active now before you migrate to z/OS V2R1, you can put this requirement behind you.

**Tip:** APAR OA40816 introduces the use of the z/OS tracking facility to monitor System SSL running in FIPS mode without ICSF active. By monitoring System SSL, you are also covering your AT-TLS usage for this migration action! Follow the instructions in the PTF for OA40816 for how to turn it on and what to look for.

**z/OS V2R1 has fonts ...and a lot of them!**

Having fonts you need in the z/OS base means they’ll always be there. In most cases, there’s no need to order and install other font products with z/OS V2R1. The new base element, z/OS Font Collection, contains the AFP Font Collection for S/390TM (5648-B33), IBM Infoprint Fonts for z/OS V1.1 (5648-E76), World Type fonts that were not previously available in the z/OS environment but form part of the InfoPrint Font Collection V3.1 available for other operating system platforms, and double-byte Asian fonts.

**Tip:** The fonts will be installed in their own file system when delivered by ServerPac. Now in z/OS V2R1, the z/OS product will be supplied in two file systems: the root (or “version root”), and the font file systems. You might have other file systems depending on what else you order with z/OS. The font file system is fairly large, approximately 2,050 3390 cylinders. If you want to merge the two file systems together during the ServerPac installation process, you can.
CEEPRMxx is required for z/OS V2R1.

Ho hum.

Ever since Language Environment® allowed us to get rid of those pesky assembler language usermods to specify the system-level runtime default options through parmlib member CEEPRMxx in z/OS V1R7, it’s been much easier for migration. This function caught on like wildfire with system programmers who were tired of updating the usermod in every release. Now, in z/OS V2R1, you have to use the parmlib member, and not the usermods. This should be no big deal, as most of you have no doubt been using CEEPRMxx for years now!

Tip: The CEE_USING_LE_PARMLIB health check for verifying your options is set using the parmlib member.

Some general installation reminders

Don’t forget about two items that you should be familiar with by now:

1. FIXCATs: The important FIXCATs for z/OS V2R1 are

IBM.Coexistence.z/OS.V2R1,
IBM.Coexistence.z/OSMF.V2R1,
IBM.TargetSystem-RequiredService.z/OS.V2R1,
IBM.TargetSystem-RequiredService.z/OSMF.V2R1,
and

2. Electronic delivery with Secure FTP (FTP using SSL or FTPS): Planned for October 1, 2013, all downloads will require the use of File Transmission Protocol Secure (FTPS) or Download Director with encryption. Make sure you are ready for this by doing a download connectivity test from this website:


Having these migration actions taken care of before you install z/OS V2R1 should get you off to a great start! For those of you who are migrating from z/OS V1R12, remember that the end of service date for z/OS V1R12 is planned for September 2014, so you should complete your migration to z/OS V2R1 by then.

More migration news

Be sure to check out the article “Are you ready to migrate to z/OS V2R1? Some valuable migration advice from Professor Kimura” on page 46 for more migration tips.
**set up high availability when failure is not an option**

*Incremental update capture engine*

**BY MARK HODSMAN, PETER MAILAND, AND DANIEL MARTIN**

The IBM DB2 Analytics Accelerator (IDAA) was a game changer for data warehousing on System z. With IDAA, long-running SQL queries can be offloaded by the DB2 z/OS optimizer to a specialized data warehouse appliance that processes SQL in a massively parallel manner. Before offloading a query however, the data must be copied and transferred to the Accelerator, but what happens when the data on DB2 for z/OS has changed?

**The problem and the solution**

In the past, you had to refresh the entire table or the partitions of the table that changed. With the new “incremental update feature” that uses IBM Infosphere Change Data Capture (CDC) technology to continuously replicate changes from DB2 for z/OS to the Accelerator, tables are now updated as changes happen on DB2. This feature not only reduces the data latency between the DB2 tables and their copies on the accelerator significantly, but also avoids the overhead caused by a full copy of the table or changed partition. Now, only the changed rows are replicated.

To complement the high-availability configuration on the appliance, IBM integrates the capture engine and the dynamic VIPA functionality of z/OS Communication Server to implement an automatic fail-over solution for incremental update on the System z side as well.

**How it works: an example**

The setup shown in Figure 2 uses a DB2 data-sharing group with two members. Each member runs in a different logical partition (LPAR) in the sysplex. Separate instances of the CDC replication engine run on each LPAR. The names are CDCSZAI on the LPAR SYS1 and CDCSZA2 on the LPAR SYS2. Only a single instance of the replication engine can be active at any time. The other instance will be in Hot Standby mode, which means that the instance has a connection to DB2, but remains in a waiting state and will not propagate data changes.

When a replication engine instance enters the active mode, it initializes its own TCP/IP task, which in turn issues a BIND operation. The Communication Server is configured such that this BIND operation is a request to pass ownership of the dynamic VIPA to the TCP/IP stack of the LPAR where the BIND command originated. If the CDC instance that is active becomes unavailable, one of the Hot Standby instances enters active mode, and issues a BIND operation. As a result, the dynamic VIPA address is moved to the TCP/IP stack that is used by the BIND operation of the newly activated instance. The connected accelerator hardware (IBM PureData™ System for Analytics N1001) uses the DVIPA address for communication that is always assigned to the active CDC instance.

In addition to the Infosphere CDC for z/OS installation, you need the PTF for APAR PM82593.
IP configuration
The IP configuration for the Infosphere CDC for z/OS is defined in the CHCCMMxx parmlib member. In this example, the instance uses TCP port 5999, and the TCP/IP stack name TCPIP.

TCP/IP SERVICENAME=5999,
SUBSYSTEM=TCPIP

Now we need to configure the zOS Communication Server. Here’s a quick summary of what we did:

1. Adapted the PROFILE.TCPIP data set for the TCP/IP stack on each of the LPARs.
2. Reserved port 5999 for the Infosphere CDC instances and associate the port number on BIND with the dynamic VIPA address 10.101.8.249.
3. For DVIPA, configured network address 10.101.8.248 with the subnet 255.255.255.252 and from this subnet, used the client address 10.101.8.249.

PORT
  5999 TCP CDCSZA1 BIND 10.101.8.249;
  VIPADYNAMIC
  VIPARANGE MOVEABLE DISRUPTIVE 255.255.255.252
  10.101.8.248
ENDVIPADYNAMIC

The first started Infosphere CDC replication agent becomes the active instance. Verify that the dynamic VIPA address is created correctly, indicated by the EZD1205I message.

EZD1205I DYNAMIC VIPA 10.101.8.249 WAS CREATED USING BIND BY CDCSZA1 ON TCPIP

You can also validate the DVIPA status of a system by using the following command:

Output from SHUTDOWN
The following example shows the output of a SHUTDOWN command invocation with the IMMED and HANDOVER parameters for the CDCSZA1 instance, while CDCSZA2 is in Hot Standby mode. The message CHC9211 indicates that the active instance ended and one of the Hot Standby instances took over.

More information
There you have it. If you’d like more information, see the DB2 library.

All subsequently started instances enter the Hot Standby mode, indicated by the CHC9212I message.

CHC9212I The meta data owned by <owner> is being used by another instance of InfoSphere CDC for z/OS, this instance is entering Hot Standby mode

To initiate a failover manually, use the SHUTDOWN command with HANDOVER parameter:

MODIFY A/SName,SHUTDOWN,[SCHED=<DateTime>|NORM|IMMED|ABORT],[HANDOVER]

where A/SName is the name of the instance of the Infosphere CDC replication agent.

If you shut down the active instance with the HANDOVER parameter, one of the Hot Standby instances becomes the new active instance. If you use this command for the active instance without the HANDOVER parameter, or issue a z/OS STOP command for this instance, the active instance and all Hot Standby instances are shut down. If you use this command for a Hot Standby instance or issue a z/OS STOP command for such an instance, only the specified Hot Standby instance is shut down.
Optimizing batch workloads is essential to the success of any z/OS production environment, and because keeping batch cycles on schedule is so important, it is crucial to be able to identify and resolve delays.

Before a batch job starts processing, it must first acquire all of the ENQs that it needs to protect its data sets and provide data integrity. If it cannot acquire all the ENQs needed, the job is delayed and must either be cancelled or left waiting in allocation jail until all the ENQs are available. These data set ENQ delays can wreak havoc on batch scheduling, resulting in a chain reaction of further delays. With no automatic detection in place, this can result in priority jobs being delayed for minutes, if not hours.

In z/OS V2R1, there’s a new SYS1.SAMPLIB tool, ISGECJES, the latest addition to the family of GRS debugging tools, and the successor to ISGECMON. The purpose of ISGECJES is to help keep batch workloads on schedule, by sending contention notification alerts that can identify batch jobs that are blocked either by TSO/E users or other batch jobs.

**ISGECJES parameters**

Like its predecessor, ISGECJES is an assembler program that scans for data set ENQ contention at a given interval specified on INTERVAL parameter (in seconds). For each instance of a contention found, ISGECJES can issue an alert, as specified on the SENDMSG parameter, as follows:

- **SENDMSG=TSOONLY**, which is the default, specifies that if the blocking job is from a TSO/E user, ISGECJES sends a message to their session, politely inviting them to free the resource, as shown in Figure 1.

**Figure 1.** ISGECJES message prompting a TSO/E user to free a resource.
SENDMSG=ALL specifies that in addition to alerting TSO/E users of a job that is blocking other jobs, ISGECJES alerts operations with a write-to-operator (WTO) message, as shown in Figure 2. The SENDMSG=ALL parameter also specifies that you want ISGECJES to flag batch jobs that are causing contention, and to issue a Display GRS,Contention command.

To prevent inattentive users from being deluged with messages, the ISGECJES MAXMSG parameter limits the number of messages that are issued for the same contention instance.

You can set these parameters either by using the PARM DD statement in JCL, or by using the MODIFY command (see Figure 1).

These dynamic tuning options make ISGECJES easy to use, and you’ll see the benefits quickly.

**Leveraging ISGECJES to optimize batch work**

Some of our enterprising customers have already found a special use for ISGECJES—pairing it with systems automation to free priority jobs by taking action on delay-causing jobs that are identified by ISGECJES messages. This is a great approach because automation can be more efficient than relying on operator response.

Other ingenious customers analyzed their ISGECJES messages to find the jobs that are prone to causing delays. Then, using this information they were able to make improvements to batch scheduling.

Customers also found TSO/E alerts useful in pointing out those cases where a user’s job held data set ENQs that, ironically, blocked their own jobs!

---

**More ENQ monitoring tools**

Global resource serialization provides a number of other ways to monitor ENQ contention; the choice of tools depends on timing and purpose:

- ENF 51 broadcasts signals when the system encounters contention.
- The ISGQUERY REQINFO=QSCAN service provides a programmatic snapshot of resources and requesters.
- The ISGQUERY REQINFO=USAGESTATS service provides statistics at an address space and overall system level.
- The DISPLAY GRS,CONTENTION command provides a simple display of resource contention. The DISPLAY GRS,ANALYZE command provides a more detailed view of the same information.
- The IPCS GRSDATA and GRSTRACE subcommands provide ENQ contention information.
- The ENQ/DEQ monitor provides a log of specific requests. It is designed primarily for resource name list (GRSRNL) planning, but it provides a detailed history of ENQ activity for diagnosis.

Added to this arsenal, the existing ISGECMON SYS1.SAMPLIB tool provides runtime contention messages, while the new ISGECJES tool adds excellent enhancements to this function, especially for batch jobs. ISGECJES messages give you the hooks for automation, which helps ensure that your batch processing runs smoothly. Be sure to take a look at this tool in SYS1.SAMPLIB!
The clock is ticking!

Implementing the TOD clock accuracy monitor service

BY DAVID WHITNEY AND RITA BEISEL

If your installation requires that your system's clock be set within a specified tolerance of standard time, you'll want to check out the TOD clock accuracy monitor service!

With IBM z/OS V2R1, the TOD clock accuracy monitor service allows any installation using a Server Timer Protocol (STP) Coordinated Timing Network (CTN) and an External Time Source to specify a maximum time variance between the external time source and the hardware Time of Day (TOD) clock. When this variance is exceeded, z/OS lets you know immediately so that your operations team or automation procedures can correct the problem quickly.

This function not only helps stock exchange members meet Securities and Exchange Commission (SEC) rules for record time stamps for the Order Audit Trail System (OATS), but it can also help any installation with security and audit concerns in fields like accounting, banking, insurance, government, healthcare, retail, and more.

Implementing the TOD clock accuracy monitor service

Setting an acceptable time deviation for the TOD clock from an external time source is easy.

1. First, ensure that your system has the following requirements:
   a. It is running the z/OS V2R1 operating system.
   b. It is operating in an STP-only CTN.
   c. It is using one of the following External Timing Sources (ETS) as the source of time:
      • The dial-out service on the HMC
      • A Network Time Protocol (NTP) server
      • An NTP server with a pulse per second output option.

2. Next, update the ACCURACY parameter in the active CLOCKxx member of SYS1.PARMLIB to indicate the acceptable time deviation for the TOD clock from the system's external time source in milliseconds. Valid values are 0 - 60000 milliseconds (60 seconds). A value of 0 is the default, and indicates that the time accuracy check function is not enabled.

That’s it! Your system is now in record time stamp compliance. You can be sure that when the specified ACCURACY bounds are exceeded (or in other words, the system TOD clock deviates from the external time source by an unacceptable amount), the system issues error message IEA032E so that you can correct the problem quickly.

Tip: If you want to be extra cautious about your record time stamp compliance, you can set the ACCURACY parameter in the active CLOCKxx member to a value smaller than the maximum deviation allowed by the installation. For example, suppose the maximum deviation allowed from the system's external time source is 3 seconds or less. Setting the ACCURACY parameter to 2500 (2.5 seconds) or 2750 (2.75 seconds) milliseconds ensures that you’ll be notified of the problem before your installation is out of compliance.

References

• See IBM z/OS MVS Planning: Operations, SA23-1390, for information about setting the TOD clock accuracy monitor service.
• See IBM z/OS MVS Initialization and Timing Reference, SA23-1380, for information about the CLOCKxx member of SYS1.PARMLIB.
As a z/OS expert, do you count on SMF records as the standard repository for all kinds of long-term measurement and event data? Of course you do!

With the IBM zEnterprise BladeCenter Extension (zBX), z/OS experts like you are more than just experts in z/OS. You’re also managing other operating systems, like IBM AIX, Linux, and yes, even Microsoft Windows.

So, if you are juggling multiple types of systems, wouldn’t it be great to have all the performance data—from every operating system type—at one place and in the same common format? If so, RMF Cross Platform Monitoring (RMF XP) is the right fit for you.

RMF XP, introduced with IBM z/OS V1R13 RMF, provides an integrated performance monitoring solution for heterogeneous operating systems running on IBM System z® or zBX hardware: AIX, Linux on System z and Linux on System x®. Since that release, IBM expanded the operating environment support for zBX to include Windows. And now, with z/OS V2R1, RMF XP also supports certain levels of Windows Server on System x! (For more information see z/OS V2R1 RMF User’s Guide, SC34-2664.)

RMF XP (also known as GPM4CIM) runs as a server instance in the z/OS Unix System Services (USS) environment. After RMF XP has accessed the performance data from the CIM API, this data can be surfaced seamlessly by combining performance data of all operating systems using the following RMF GUIs:

- The RMF Performance Data Portal (Web Browser GUI)
- The Resource Monitoring plug-in of the z/OS Management Facility (also known as z/OSMF).


And there’s more great stuff!

Beyond the online monitoring capabilities, with z/OS V2R1, the RMF GPM4CIM component is enhanced with SMF persistence capability as an additional tool for long-term performance analysis and capacity planning. Performance data that is collected from the distributed operating systems can be written to the new SMF Record Type 104 (x’68’). The type 104 SMF record, RMF Distributed Platform Performance Data, serves as a container for all measurement data of non z/OS platforms.

For each platform, the system writes the following range of subtypes:

**Subtype 1-12:** AIX on System p® performance data  
**Subtype 20-31:** Linux on System x performance data  
**Subtype 40-53:** Linux on System z performance data  
**Subtype 60-64:** Windows on System x performance data
You can enable this function with just a few configuration steps:

1. Use the SMF built-in facilities to control the SMF 104 recording on subtype level. You can dynamically switch on or off collection of the new RMF XP type 104 SMF records for each specific subtype by changing the active SMFPRMxx parmlib member. For example:

   ```
   SYS(TYPE(30,42,70:79,103,104(1:12,20:31,40:53,60:64),108)),
   ```

   The new RMF XP SMF records can now be written to SMF data sets or log streams.

2. Now, you’re ready to receive and hold a real-time copy of the SMF records in the in-storage SMF buffer of the RMF Sysplex Data Server. To use this feature, modify the PARM statement of the RMF procedure to include SMF type 104 as follows:

   ```
   //RMF		PROC
   //IEFPROC	EXEC	PGM=ERBMFMFC,REGION=32M,TIME=1440,
   //				PARM='SMFBUF(RECTYPE(30,70:79,104))'
   ```

3. Configure RMF XP. To do this, use the new RECORD/NORECORD option in the RMF XP configuration file (/etc/gpm/gpm4[A|X|Z|W].cfg) to specify that the GPM4CIM server writes SMF type 104 records.

4. (Optional) Of course, you can also enable or disable SMF recording with the MODIFY command: F GPM4CIM,[NO] RECORD. After you set the RECORD parameter on, the system writes SMF 104 records frequently. The system issues the following message to indicate that an interval completed:

   ```
   GPM280I	RMF	XP	INTERVAL	PROCESSING	FOR	name	COMPLETE
   ```

You can also reuse your existing SMF record-based reporting infrastructure to keep track of the performance of your Linux, AIX and Windows systems in terms of a detailed after-the-facts analysis. For example, you can export your processor utilization data to a spreadsheet and generate an overview chart as shown in Figure 1.

![Processor use chart generated from SMF 104 data.](image)

**Figure 1.** Processor use chart generated from SMF 104 data.

Last, but not least, you can use your old friend ERBSCAN data-set-name in SYS1.SERBCLS to take a deeper look at the new RMF XP SMF 104 records by using ISPF. To display a detailed view of a single SMF record with subtype data, use ERBSHOW nn X which displays record nn from data-set-name.

**Our website has more**

Check out the RMF home page [ibm.com/servers/eserver/zseries/rmf](http://ibm.com/servers/eserver/zseries/rmf) and the IBM z/OS RMF XP Implementation Guide for further setup hints!

---

*With the IBM zEnterprise BladeCenter Extension (zBX), z/OS experts like you are more than just experts in z/OS. You’re also managing other operating systems, like IBM AIX, Linux, and yes, even Microsoft Windows.*
Now, z/OS V2R1 has more Job Control Language (JCL) and batch processing enhancements than any release in recent history, covering everything from constraint relief and performance features to ease-of-use items. Read on to see what you can use today or exploit in the future!

**Limits on JCL PARM — solved!**
The parameter string that can be specified on the existing JCL EXEC statement PARM= keyword is limited to no more than 100 characters. Many programs need parameter strings that exceed this limit, so some programmers might have already invented their own solutions.

The new PARMDD= keyword specifies the data definition (DD) name. The data set associated with the DD name can contain a parameter string of up to 32,760 bytes. The PARMDD= keyword is mutually exclusive with the PARM= keyword.

No changes are required to unauthorized programs that adhere to the previously-documented program interface, which has always pointed to a halfword-length field located before the parameter string field. The parameter string field can now be up to 32,760 bytes in length.

The data set must be a physical sequential (PS) data set, but can have fixed, fixed-block, variable, or variable-block record formats.

```
//LONGPARM JOB MSGLEVEL=1
//STEP0001 EXEC PGM=MYPROGRM,PARMDD=PARMINDD
//PARMINDD DD DSN=MYPARMS.TEXT,DISP=SHR
```

Partitioned data set members with those record formats are also supported:

The data set referred to by the PARMDD= keyword can be a Job Entry Subsystem 2 (JES2) in-stream DD * or DD DATA data set. The in-stream data set can contain symbols that are resolved by JES2.

```
//LONGPARM JOB MSGLEVEL=1
//STEP0001 EXEC PGM=MYPROGRM,PARMDD=PARMINDD
//PARMINDD DD DSN=MYPARMS.LIBRARY(PARMS1),DISP=SHR
```

The parameter string passed to the MYPROGRM program is:

```
Parameters for MYPROGRM running on SY1
```

where SY1 is the name that JES2 substituted for the &SYSNAME system name symbol.
The data set referred to by PARMDD= can be a z/OS UNIX System Services (z/OS UNIX) file:

```bash
//LONGPARM JOB MSGLEVEL=1
//STEP0001 EXEC PGM=MYPROGRAM,PARMDD=PARMINDD
//PARMINDD DD PATH='/tmp/myparms.txt'
```

The parameter string can be formed by concatenating data sets, members, or files together, subject to data set rules similar to those for basic sequential access method (BSAM).

**GDG ALL concatenations — reordered!**

z/OS MVS has always let you refer to all generations of a generation data group (GDG) with one simple DD statement. Specify the GDG base name and all of the individual generation data sets (GDSes) are concatenated together, allowing easy reference and easy reading. However, the order in which they were concatenated (with the newest generation first) was backward for GDSes that contained time-ordered data. One example of this was system management facilities (SMF) data, where each GDS contained one day of SMF data, and the whole GDG together provided a month’s worth of data. With the new GDGORDER=FIFO JCL keyword, you can request that the data be concatenated with the oldest generation first. This means that the program reading the data can get all of the data in order, without having to sort the records by time. Removing a SORT step from a job can result in a great reduction in processing overhead and elapsed time.

**Batch job parallelism — increased!**

It has often been noted that jobs that access a data set with OLD, NEW, and MOD dispositions in an early step continue to hold that data set with exclusive control, even though the rest of the job has specified DISP=SHR. This exclusive control keeps it from other jobs that want to share it, reducing the parallelism that batch jobs can achieve.

In z/OS V2R1, we introduce new options to give you increased control over those jobs. With the new DSENQSHR=ALLOW JCL keyword, the JCL coder can request that data sets that start with DISP=NEW, MOD or OLD in early steps, but have DISP=SHR in later steps, actually change to shared control after the last step where it was needed as OLD, MOD, or NEW.

Of course, changing each job to include this new keyword can be tedious, so we also have a higher level of control, at the JES2 job class level. Add DSENQSHR=AUTO to a JES2 JOBCLASS definition and all jobs running in that job class will be allowed to downgrade their SYSDSN ENQs to shared control, to increase your batch job parallelism.

Be sure to check out the product documentation for the dependencies and additional keyword options that give you full control over your batch jobs and their data sets!

**Jobs delayed waiting for DFMSHsm recall — reduced!**

Have you ever submitted a job, expecting it to complete within a minute or two, and five minutes later wondered why it was still running? Often this occurs when the data sets the job uses have been migrated by DFMSHsm. The processing behind the allocation of data sets used by batch jobs has always recalled each data set as it is encountered in the JCL, resulting in serial DFMSHsm recalls. This leaves the job owner wondering impatiently, “Why didn’t it recall all the data sets at once??!”

With z/OS V2R1, there is now a system option in the ALLOCxx parmlib member that you can use to change this behavior. Coding SYSTEM BATCH_RCLMIGDS(PARALLEL) in ALLOCxx (or using the SETALLOC SYSTEM,BATCH_RCLMIGDS=PARALLEL command) will enable a new function in Multiple Virtual Storage (MVS) allocation that looks at all of the data sets needed as each job step is processed, and recalls all of the DFMSHsm-migrated data sets in parallel. The result is shorter processing time for jobs, and freeing up of initiators earlier for other jobs to run.

**JCL needing system symbols — enhanced!**

Prior releases of z/OS have always limited the use of system symbols to system-oriented JCL, such as that used by started tasks. This was important because system symbols often varied by system, and in a JES MVS environment, unexpected results could occur when a job used one system’s symbols during its initial parsing, but then ran on a different system.

Static system symbols are often used for more than just system-oriented values. Installation-wide values, such as a company division name or even the location of the installation, are encoded in system symbols. Isn’t it about time that regular batch jobs are able to access those symbols?

With z/OS V2R1, a new job class option, SYSSYM=ALLOW, allows jobs in that class to use system symbols, even if they aren’t started tasks or started jobs. For example, you could create a system symbol named LOCATION and give it the value “ORLANDO”, then use &LOCATION in your batch JCL, wherever you previously had to hard-code “ORLANDO”. This makes it easier to reuse JCL across your enterprise.
It’s still not recommended for batch job JCL to use system symbols that vary by system within the JES complex. However, if it is necessary, we suggest including JCL keywords that will ensure that the job runs on the system intended. In z/OS, this can be the new JCL keyword, SYSTEM= on the JOB statement, which takes the place of the JES2 SYSAFF= and JES3 SYSTEM= JECJL keywords. The older JCL keywords can also be used.

**JCL symbol processing — improved!**

JCL symbols allow you to modify statements in a job easily. They are only available to the job converter, but are discarded by the time the job runs. We knew that improving this process would simplify the writing of JCL statements, for example, by allowing substitution in in-stream (that is, SYSIN) data sets. In z/OS V2R1, we added new JCL statements, keywords, and services to make JCL symbols associated with a step at conversion time available to the step at execution time.

The new JCL EXPORT statement lets you specify the symbol names that you want to pass to the step at execution time:

```
//EXPRT3 EXPORT SYMLIST=(SYM1,SYM2,SYM3)
```

Use the wild card character (*) to include all:

```
//EXPRTALL EXPORT SYMLIST=* 
```

The SYMALLOC keyword on a SYSIN DD statement allows you to indicate to JES2 which set of symbols are to be used in the in-stream data:

```
//SYSUT1 DD *,SYMLIST=(JCLONLY,LOGDD)
```

The first parameter can be JCLONLY (only JCL symbols on EXPORT SYMLIST), EXECSYS (system symbols on the executing system plus JCL symbols that were exported earlier), or CNVTSYS (system symbols on the converting system plus JCL symbols that were exported earlier). The second parameter is a valid DD name. Figure 1 shows an example:

```
//MYSYMTST JOB MSGCLASS=A,MSGLEVEL=(1,1),REGION=0M,CCLASS=1
//MYEXPRT EXPORT SYMLIST=* 
//MYSET1 SET PARM2=HIFROMPOK 
//STEP1 EXEC PGM=IEBGENER 
//SYSPRINT DD SYSOUT=A 
//SYSIN DD DUMMY 
//SYSUT1 DD SYSOUT=* 
//SYSPRINT DD DD SYSOUT=A 
//SYSIN DD DUMMY 
//SYSPRINT DD DD SYSOUT=* 
//SYSPRINT DD DD SYSOUT=* 
//SYSPRINT DD DD SYSOUT=* 
```

Figure 1. Indicating a set of symbols to be used for in-stream data

This results in the symbols being resolved as shown below:

```
PROCESSING ENDED AT EOD 
&PARM2 &SYSOSLVL 
```

— as opposed to the following without EXPORT statement or a SYMALLOC keyword:

```
PROCESSING ENDED AT EOD 
&PARM2 &SYSOSLVL 
```

An internal reader is a special type of SYSOUT data set which other jobs can use to submit jobs, control statements, and commands to the JES. The SYMALLOC keyword (or its corresponding Dynalloc text unit ‘802B’s) on a SYSOUT DD statement enables you to pass a list of symbol names into the internal reader. You can use an asterisk (*) on SYMALLOC as a wild card character to allow all. Figure 2 shows how the local variables set in the first job are passed through the internal reader to the second job:

```
//FIRSTJOB JOB MSGLEVEL=(1,1),MSGCLASS=A,NOTIFY=IBMUSER 
//MYSET1 SET DSNAME=HASP.TEST.MACLIB 
//MYEXPRT EXPORT SYMLIST=(DSNAME,VOLSER) 
//SYSOUT DD SYSOUT=* 
//SYSIN DD DUMMY 
//SYSPRINT DD SYSOUT=(A,INTRDR),SYMLIST=(DSNAME,VOLSER) 
```

Figure 2. Setting and passing local variables
Here, FIRSTJOB submits SECJOB. The local variables are passed to the submitted job for processing.

It is not enough if we just exported the symbols; programs need someone to fetch them. For this purpose, we added the scheduler JCL facility (SJF) symbol service IEFSJSYM. It retrieves the specified symbol (REQUEST=GETBYNAME) or all (REQUEST=GETALL) JCL symbols and their values for the job step in the area provided by the caller. Together with the EXPORT function, IEFSJSYM enables you to pass JCL symbol information from the submitted JCL to the program running under the submitted JCL.

IEFSJSYM
REQUEST=GETALL,SYMBAREA=SYMBOLAREA,SYMBAREALEN=256,RETCODE=SJSYM_RC,RSNCODE=SJSYM_RSN,DIAGDATA=SJDIAG

In-stream data in JCL procedures and 8-character job classes — available everywhere!
In z/OS V2R1, the JOB CLASS= keyword has been updated to allow 8-character job classes, which are now supported in JES2 and JES3. In addition, JES3 now supports the ability to include in-stream data in JCL procedures, originally supported for JES2 in z/OS V1R13. These long-standing requirements are now fully resolved in z/OS V2R1.

Ready when you are!
All of these items have enablement controls, so you can install z/OS V2R1 easily, then “kick the tires” on these upgrades when you’re ready. Be sure to check out the official documentation before experimenting; for example, not all features are available on JES3 systems, but when the time comes, which will you choose first?

Act Now! Be ready for secured downloads
You probably heard all about the secured downloads requirement that was in the February 5, 2013 preview announcement for z/OS V2R1. If not, here is a friendly reminder about this change.

All direct-to-host downloads of z/OS product and service orders require secured connections such as File Transfer Protocol (FTP) with the Secure Sockets Layer (SSL) or FTPS. This change affects:

- ServerPac
- CBPDO
- CustomPac product orders
- Shopz service orders
- SMP/E RECEIVE ORDER.

On August 26, 2012, there was a change to the host names and IP addresses of the IBM servers for orders. See the Washington Systems Center Flash 10780: ibm.com/support/techdocs/atsmastr.nsf/WebIndex/FLASH10780

Test now
If you are not ready for the withdrawal of standard FTP connections, act now to verify that your system is set up for secured downloads

2. Start using the FTPS option on the Shopz download page to download your z/OS product and service orders.

If you cannot use FTPS in your environment, use the store-and-forward alternative to download to your workstation. The store-and-forward requires no system setup because it uses Download Director with encryption.
Quick and Easy: BCPii!

BY STEVE WARREN AND RITA BEISEL

It's time to check out the Base Control Program internal interface (BCPii) in z/OS Version 2 Release 1 (V2R1). The improvements in BCPii function might be the quick and easy recipe to help you start using this base function of the z/OS operating system. If you are already using BCPii, you can now use it more efficiently than ever.

BCPii at your service
In z/OS V2R1, BCPii supports applications written in the REXX programming language, known for its ease of use. BCPii also minimized the traffic to the support element (SE). Less traffic to the SE might equal improved performance for you. Let's first take a step back and look at BCPii.

BCPii is a cool way to access System z hardware controls from any z/OS authorized application running in any address space. For example, you might want to:

• Find out what is going on with the hardware
• Perform powerful tasks like re-IPL or load an LPAR
• Receive notification when certain hardware events occur.

Do you want to do all these things from the convenience of your z/OS application? If so, BCPii is at your service! It’s not necessary to install a suite of products or complete a complicated install process to start using it.

Ready for REXX?
Before z/OS V2R1, the BCPii APIs supported applications using either the C or assembler programming languages. Over the years, there has been a growing and vocal demand for REXX programming language support in BCPii API.

We listened and delivered
In z/OS V2R1, the BCPii support for REXX and a much simpler programming model than either the C or assembler programming languages, you can get applications up and running quickly and easily.

BCPii APIs support applications using REXX in the z/OS System REXX, TSO/E REXX, and independent software vendor (ISV) provided REXX programming environments. Not only does writing with the REXX programming language allow you to develop BCPii applications in record time, but also maintains your investment in your existing BCPii applications written in C or assembler. These REXX applications can work right along side them.

Sample BCPii REXX exec
Here is a simple BCPii REXX exec that lists all the interconnected processors in your Hardware Management Console (HMC) network.

Notice the intuitive programming style. Just specify the list type and voilà, BCPii returns the data in a stem variable. The zero element of the stem variable contains the number of items returned and the 1 to n elements contain the actual names of the processors connected to the system. This is only an example, but the other BCPii API calls are just as intuitive and easy to use.

```
LISTTYPE = HWI_LIST_CPCS
ADDRESS BCPII "HWILIST"
RETURNCODE
CONNECTTOKEN
LISTTYPE
CPCLIST.
DIAGAREA.”

IF RC <> 0 | RETURNCODE <> 0 THEN
/* IF THE REXX RC IS NOT GOOD OR THE BCPII RETURN CODE IS NOT GOOD, HAVE ERROR HANDLING CODE HERE */
ELSE
DO
SAY 'NUMBER OF CPCS RETURNED = ' CPCLIST.0
/* WRITE THE LIST OF CPCS RETURNED. THE .0 ELEMENT CONTAINS THE NUMBER OF ITEMS RETURNED */
DO I = 1 TO CPCLIST.0
SAY 'CPC '|| I ' = ' CPCLIST.I
END
END
```

Figure 1. Sample BCPii REXX exec
Which REXX is best?

No matter which REXX environment you choose to run your BCPii REXX exec, the exec will run the same way. You write your BCPii REXX exec once and it runs in any REXX environment that you choose. Of course, there’s always an exception that proves the rule - HWICMD and HWIEVENT are not supported in the TSO/E REXX or the ISV-provided REXX environments. Table 1 shows the BCPii API and corresponding REXX environment.

<table>
<thead>
<tr>
<th>BCPii APIs</th>
<th>System REXX environment</th>
<th>TSO/E REXX environment</th>
<th>ISV-provided REXX environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWICONN</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>HWDISC</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>HWILIST</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>HWQUERY</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>HWSET</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>HWIEVENT</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>HWICMD</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. BCPii APIs supported in the REXX environment

z/OS System REXX

z/OS System REXX is the IBM-recommended programming environment for running authorized REXX execs. In this environment, you can use all BCPii API services, including HWICMD and HWIEVENT (Note that HWICMD and HWIEVENT require a non-REXX ENF event exit, such as the IBM-supplied sample REXX exec provided in SYS1.SAMPLIB).

BCPii REXX execs running in the z/OS System REXX environment can also interoperate with existing non-REXX BCPii applications written in C or assembler. A connect token can be passed to and from the REXX exec to other compiled or assembled BCPii applications. BCPii REXX execs, running under the z/OS System REXX programming environment, also have full address space affinity connections, just like their C and assembler counterparts. Connections stay persistent until the address space starting the REXX exec ends, or if running in an initiator, when the job ends. This allows flexible design of your applications.

You can start the BCPii REXX exec in the z/OS System REXX programming environment by:

- Using the new helper program, HWIREXX. For a simple example of how your BCPii REXX exec can be started from JCL, see the SYS1.SAMPLIB JCL member HWIXMRJL.
- Coding an assembler program to start the AXREXX macro. In addition to the name of your BCPii REXX exec, remember to specify any environmental options that your BCPii REXX exec requires.

TSO/E REXX

What is simpler than running a BCPii REXX exec from a TSO user? Just type the TSO EXECUTE command, specify the name of your BCPii REXX exec, and say hello to BCPii! The system administrator just adds a line to the IXJTSOxx parmlib member to allow BCPii to run authorized under TSO/E. That’s it. (See the section “From hours to minutes” for where to find the complete details.)

Next, BCPii, as it does for any API request, checks the SAF authorization of the TSO user attempting to run the exec to determine if the user has authorization to perform the requested BCPii function.

When the BCPii REXX exec completes, TSO/E REXX connections created by the BCPii REXX exec are automatically cleaned up by BCPii. BCPii connections created while running in the TSO/E REXX environment cannot be shared with other BCPii applications.

Note: BCPii supports all BCPii APIs except HWICMD and HWIEVENT in the TSO/E REXX environment.

ISV-provided REXX

When you want to use one of the independent software vendors (ISV) REXX environments to run your BCPii REXX exec, it’s not a problem! We got you covered there as well.

To work in this environment, have your BCPii REXX exec enable the BCPii host environment by coding HWIHOST(“ON”) at the beginning of your BCPii REXX exec. Presto, your ‘address bcpii’ statements in your REXX exec are enabled. Now that’s easy!
When the BCPII REXX exec completes, ISV-provided REXX connections created by the BCPII REXX exec are automatically cleaned up by BCPII. BCPII connections created while running in the ISV-provided REXX environment cannot be shared with other BCPII applications. Note that BCPII supports all BCPII APIs except HWICMD and HWIEVENT in the ISV-provided REXX environment.

Get ready for less traffic to the SE

Let’s face it, BCPII list and retrieval requests can be slow, especially when multiple values are requested from the support element (SE). BCPII calls requesting multiple attributes are just plain faster on z/OS V2R1 because BCPII minimizes the data flows to the SE for multiple attributes on HWIQUERY requests, as well as practically all HWILIST requests.

Most BCPII applications that issue an HWILIST request or an HWIQUERY request that queries many attributes should experience an improvement. For example, if you have ten LPARs on one CPC and you issue an HWILIST to list all the LPARs, the number of calls to the SE shrinks from eleven interactions to one or two, at the most. How’s that for efficiency? Less calls to the SE results in quicker performance for you.

How do BCPII applications use this?
The efficiencies are in the BCPII code. For HWILIST calls, you don’t do anything. For HWIQUERY calls, consider bundling your query calls. For example, instead of having two HWIQUERY calls that query two attributes modify the code to have one HWIQUERY call that queries all four attributes. By doing so, the number of interactions to the SE goes from four to just one. If you choose not to bundle, in this example, BCPII on z/OS V2R1 has two interactions with the SE (one for each HWIQUERY request), instead of four.

From hours to minutes!

BCPII application development that took hours or even days might now take minutes thanks to the ease of use of the BCPII REXX programming language support. Minimized traffic to the SE is also a beneficial improvement. For more information, see the BCPII chapter in z/OS MVS Programming: Callable Services for High-Level Languages, SA23-1377.

Free checking with z/OS V2R1!

For years, IBM Health Checker for z/OS has been helping systems run smoothly, running checks that ensure correct configurations, help with migration, and prevent outages.

Now, beginning in z/OS V2R1, the HZSPROC procedure starts IBM Health Checker for z/OS automatically during IPL. All you have to do to start using this valuable service is to ensure that the IBM-supplied PROCLIB data set is in your JES procedure library concatenation for started tasks, or that HZSPROC is copied from PROCLIB to one of those libraries. Then, if a health check finds exception conditions, it sends explanatory messages to the console. You can use the convenient SDSF CK panel to view the full check message buffer that contains the details and suggested actions to help you fix each problem. After you address the initial set of check exceptions, IBM Health Checker for z/OS is quietly there for you.

If you are already using IBM Health Checker for z/OS, you’ll want to make a few adjustments to take full advantage of automatic start-up. See IBM z/OS V2R1 Migration, GA32-0889.

For more information, including check descriptions, see IBM Health Checker for z/OS V2R1 User’s Guide, SC23-6843.
A
n application is traveling through instructions, instructions not only representing business logic, but user interfaces; a transaction of the utmost importance and urgency. An error lies in the instructions ahead; if only it had stepped into: the HEAPZONE!

P.A. Rolle is an often-overlooked application who has lived quietly on a large z/OS image far away from the spotlight. Up until now, he has gone about his mundane task of processing input records and producing its output, day after day, and night after night. P.A. Rolle doesn’t know it yet, but the strange, unrecognized input he is about to receive will spiral him out of control. Alarms begin to rock the otherwise quiet z/OS image and cause a frenzy of unwanted attention. Cries of heap damage fill the screens; phones begin to ring; pagers begin to sound; and poor P.A. Rolle is terminated...

P.A. Rolle had done something horrible. Horrible and undetected! The faulty input had caused P.A. Rolle to deface important control information in storage managed by Language Environment, called heap. Even though P.A. Rolle was done with his storage, the damaged information lingered and moved throughout the heap before being detected and eventually setting off the frenzy.

If only P.A. Rolle had stepped into the HEAPZONE! You see, P.A. Rolle was running on a z/OS V2R1 system, where the hard-working Language Environment developers had provided a way to help. A new run-time option is available that enables quicker detection and can, in some cases, prevent the frenzy from occurring. HEAPZONES would have let P.A. Rolle request that an extra piece of storage (a check zone) be automatically added to the end of the storage otherwise requested. Additionally, that storage can be checked, when it is freed, to ensure that P.A. Rolle or others did not incorrectly write data into the check zone.

P.A Rolle could have chosen to step into the HEAPZONE simply and easily by setting a Language Environment run-time option called HEAPZONES. However, he cannot set HEAPZONES in a CEEPRMxx parmlib member or with a SETCEE command.

The HEAPZONES run-time option is used to turn on user heap overlay toleration and checking. By default, no extra storage is obtained; however, when activated, the run-time option affects any request for storage that is satisfied in a Language Environment user heap. This is storage controlled by the HEAP or HEAP64 run-time options. HEAPZONES also affects storage obtained from a heap pool.

P.A. Rolle could have turned to the z/OS Language Environment Programming Reference, SA38-0683 to learn the details of using and setting the HEAPZONES run-time option:

**size31**
Controls the size of the check zone for all user heap storage that is below the 2 gigabyte (GB) bar. Check zone size is rounded up to the nearest multiple of 8 bytes. The maximum size allowed for a check zone is 1024 bytes. Specifying a value of 0 indicates that no check zone is active.

**size64**
Controls the size of the check zone for all user heap storage that is above the 2 GB bar. Check zone size is rounded up to the nearest multiple of 8 bytes with a minimum size of 16 bytes. The maximum size allowed for a check zone is 1024 bytes. Specifying a value of 0 indicates that no check zone is active.
With the HEAPZONES run-time option, P.A. Rolle could have chosen the amount of information that was gathered when a check zone was violated.

ABEND The default – Specifies that a heap check zone is appended to every allocated element and that the check zones are validated. If an overlay is detected, a U4042 ABEND reason code 3 is issued.

QUIET Specifies that a heap check zone is appended to every allocated element, but that the check zone will not be validated.

MSG Specifies that a heap check zone is appended to every allocated element and that check zones are validated. If an overlay is detected, informational messages are issued.

TRACE Specifies that a heap check zone is appended to every allocated element and that check zones are validated. If an overlay is detected, in addition to informational messages, a CEEDUMP containing only a traceback is produced.

P.A. Rolle can also use the CLER CICS transaction to manipulate the HEAPZONE run-time option:

Of course, P.A. Rolle must use this new-found power wisely. Specifying HEAPZONES to check user heap storage elements for an overlay can result in a performance degradation and significant additional storage usage. IBM recommends setting the size of a heap check zone to the smallest amount possible for the application.

The incidents of that fateful day contained a valuable lesson. While there are no guarantees in the life of any application, those that take precautions can live to run another day. The now-wiser P.A. Rolle has learned that when unforeseen circumstances arise, it is always better to step into … the HEAPZONE!

Syntax

```
HEAPZones (size 31, QUIET, MSG, TRACE)
HEAPZones (size 64, QUIET, MSG, TRACE)
```

CLER		CICS CICS660

Language Environment Region Level Run-time Options

Type in your Choices.

<table>
<thead>
<tr>
<th>Runtime option</th>
<th>Choice</th>
<th>Possible choices.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL31</td>
<td>=&gt; ON</td>
<td>ON, OFF</td>
</tr>
<tr>
<td>CBLESHPOP</td>
<td>=&gt; ON</td>
<td>ON, OFF</td>
</tr>
<tr>
<td>CHECK</td>
<td>=&gt; ON</td>
<td>ON, OFF</td>
</tr>
<tr>
<td>INFOMSGFILTER</td>
<td>=&gt; OFF</td>
<td>ON, OFF - ON equates to INFOMSGFILTER(ON,CICS)</td>
</tr>
<tr>
<td>HEAPZ size31</td>
<td>=&gt; 0</td>
<td>0 - 1024</td>
</tr>
<tr>
<td>HEAPZ out31</td>
<td>=&gt; ABEND</td>
<td>QUIET, MSG, TRACE, ABEND</td>
</tr>
<tr>
<td>RPTOPTS</td>
<td>=&gt; OFF</td>
<td>ON, OFF</td>
</tr>
<tr>
<td>RPTSTG</td>
<td>=&gt; OFF</td>
<td>ON, OFF</td>
</tr>
<tr>
<td>TERMTHDACT</td>
<td>=&gt; TRACE</td>
<td>QUIET, MSG, TRACE, DUMP, UAONLY, UATRACE, UADUMP, UAIMM</td>
</tr>
<tr>
<td>TRAP</td>
<td>=&gt; ON</td>
<td>ON, OFF</td>
</tr>
</tbody>
</table>

When finished, press ENTER.
The IBM Rational® Development and Test Environment for System z (RD&T) can improve the development, maintenance, and testing of mainframe applications. This article explains what RD&T is and describes a sample usage scenario.

**z/OS on your laptop**

We all know mainframes run continuously. It’s rare for them to stop. In contrast, it’s common to shut down and restart personal computers.

What do you think about a z/OS system running on your notebook that you can IPL any time you want?

RD&T creates a System z environment on a personal computer, enabling you to run typical z/OS middleware for development and testing. RD&T is based on the IBM System z Personal Development Tool (zPDT®) and runs on a Linux system based on an x86 personal computer (PC). It provides a System z environment on a PC can run current System z operating systems, including virtualization of selected System z input/output (I/O) devices and control units. The intended use is a development, demonstration, and learning platform—it is not designed to be used as a production system.

**Overcoming obstacles**

Sometimes, mainframe developers feel constrained when developing mainframe applications because they might not have the same flexibility and availability that is typical when developing in a distributed environment. Some development obstacles include:

- Sharing the system with production and having a lower priority
- Changes to the environment that depend on the z/OS system
- Programmers, who are busy with the production environment
- Developing applications using the Interactive System Productivity Facility (ISPF) user interface, which can be unfamiliar or intimidating to new hires
- Using MIPS for development versus production.

The RD&T objective is to mitigate these inhibitors and give mainframe developers the same agile capabilities that are available in the PC environment.

**What is RD&T?**

The Rational Development and Test Environment for System z (RD&T) provides a small-scale, personal development and testing environment for developers that can run IBM z/OS and middleware on a PC. RD&T helps provide a low-cost, flexible development environment.

RD&T comes packaged with a newly-updated and preconfigured set of IBM software entitled for development use specifically in the Rational Development and Test Environment for System z (for specific details, see “More information”).
You can have more than one developer connected to an RD&T server; the number of users you can connect depends on the capacity of your server. For example, tests conducted developing COBOL applications for Customer Information Control System (CICS) with a Lenovo ThinkPad model W510 with 16 gigabytes (GB) handled 15 concurrent developers connected with an acceptable response time.

The many RD&T benefits include using a less-expensive computer, such as a laptop, to develop mainframe applications. Other important benefits include the ability to:

- Develop and test System z applications anywhere at any time
- Prototype new applications
- Leverage an agile team workbench and help accelerate delivery of new workloads that span multiple platforms
- Integrate easily with other Rational and IBM tools for additional productivity and MIPS savings
- Reduce dependencies on operations staff.

Using RD&T
RD&T is based on the zPDT. You need a desktop or laptop computer that can run Linux and have the RD&T software installed on the Linux system. See the resources under “More information” for the supported version of Linux.

The installation process must follow this sequence:

1. Install and customize Linux.
2. Install the zPDT software.
3. Configure the license server
4. Activate the RD&T Universal Serial Bus (USB) hardware key
5. Configure the z/OS installed software and required components.

Sample scenario
The example in Figure 2 uses Rational Developer for System z (RDz) and RD&T. The steps reflect the sequence of tasks involved, along with some representative screen captures.

In this example, an RD&T server is running on a laptop with 16 GB of random-access memory (RAM). The developer (client) is using another laptop with an RDz client. Also added is a mainframe, located in Texas, and a developer who is working remotely. Figure 2 shows the architecture of this scenario. Figure 2 also shows Remote System Explorer (RSE), which is one of the components that you must install on z/OS to use RDz.

1. **Boot Linux.** You must install the RD&T USB key before initializing. The process of booting Linux is the same as any operating system boot. You type an authorized ID and password and the system opens a terminal where you enter Linux commands. In Figure 3, the boot process takes about three minutes.

2. **Start zPDT.** When Linux is running, enter a command to start zPDT and z/OS. Figure 3 shows the messages that are displayed on the Linux terminal. Usually, the Linux commands to start and stop zPDT are as follows:
Start zPDT by entering the awsstart aprof1 command. In Figure 3, the aprof1 file is a configuration map for zPDT. It defines the memory that you give to z/OS and the devices that z/OS virtualizes, such as the direct access storage devices (DASDs).

Stop zPDT and shut down Linux by entering the awsstop command. This command instantly ends System z operation. You should always stop the System z operating system normally before issuing awsstop.

3. IPL z/OS under Linux. The z/OS IPL will initiate the z/OS console and load z/OS. It’s up to the installation to initialize the products that you need for development. For example, an IPL loads z/OS and might automatically start the job entry subsystem (JES), CICS, and Virtual Telecommunications Access Method (VTAM). When the IPL is complete, testing begins.

4. Connect to RD&T and the z/OS mainframe. This is a client operation. The client laptop connects to the mainframe in Texas (optional) and to RD&T.

5. Copy the code from the mainframe to RD&T. In this step, the developer has a remote project defined and starts working with the code. Figure 4 shows a drag-and-drop operation where the developer copies the COBOL program BK92S1 to RD&T for the changes. After this transfer, the developer can disconnect from the mainframe.

6. Edit and modify the COBOL code. The activities here are traditional for RDz. You can make changes using the editor, compile, deploy to RD&T, and so on. CICS Explorer® can help with CICS deployment.

7. Using z/OS Debug running on RD&T. The developer debugs the code using the z/OS Debug tool that is part of RD&T.

More information

- IBM Redbooks IBM System z Personal Development Tool for details on installing and configuring the zPDT environment: redbooks.ibm.com/redbooks/pdfs/sg247721.pdf

Figure 3. Starting zPDT on a Linux terminal

Figure 4. Example of copying COBOL program from mainframe to zPDT
You can’t argue with the numbers. Over one hundred businesses in the enterprise computing community have enrolled employees in Marist College online z/OS classes. Through its Institute for Data Center Professionals (IDCP) program, IBM clients, IBM Business Partner, and students from Marist College and other universities, as well as working professionals from around the world have taken classes remotely.

The programs
The programs are designed to educate personnel who are entering the field plus those with experience in System z and z/OS. There is a variety of enterprise systems education programs from which to choose. This includes system programmer certificates and application programming tracks including COBOL, DB2, Assembler, and IMS™. A combination of lecture and “hands on” experience helps build valuable skills needed by mainframe professionals.

The program, content, advisory board, and teachers are from the enterprise computing community — experts in their field. The z/OS advisory board includes 24 enterprise computing experts. The teaching staff contains seasoned professionals who know z/OS and zEnterprise middleware.

Student testimonials
The long list of student testimonials impressed the Hot Topics staff, but we had a bit of trouble fitting the long list into this issue. Please check them out at.

Affordable and flexible
Education budgets and time away from your job are challenges. Now, there’s an opportunity to continue your education, without travel, and to train yourself at an affordable price. New York State and IBM also provide scholarship funding. Consider the value of enrolling in a professional led System z Associate Certificate — three courses, each 10 to 12 weeks in length, all at a reasonable cost.

These courses were designed to offer flexibility and to meet all levels of expertise. Consider one of the following:

- Systems Programming Track
- System z Associate Certificate (3 courses)
- System z Professional Certificate (3 courses)
- System z Expert Certificate (3 courses)
- Application Programming Tracks
- COBOL Application Programming Certificate
- Assembler Application Programming Certificate
- IMS Application Programming Certificate
- DB2 Application Programming Certificate

Don’t re-invent the wheel
We tell IBM clients to use the work already done from many in the enterprise computing community. Don’t build internal training programs within your business that duplicate what’s already available at low cost and high quality.

Internal training programs are important, but it’s better when they focus solely on the unique aspects of your business and processes, not the fundamentals of z/OS, middleware, and programming languages. People can obtain systems programming and application development skills remotely from the courses offered by Marist College through the IDCP. Some IBM clients use the System z Associate Certificate as entry-level criteria for new hires.

Getting started
For complete information, see: idcp.marist.edu/learnzos or write learnzos@Marist.edu.
Making the switch

Simplified cascade support

BY STEPHEN M. KOCK AND DALE RIEDY

Planning and setting up a large I/O configuration takes considerable time and effort, not to mention the skill and knowledge involved in factoring in performance needs, availability needs, and workload balancing. Finding the right balance can be difficult and some might argue it is more of an art rather than a science when defining and implementing an I/O configuration. Fortunately, there is dynamic channel-path management (DCM), which helps simplify the I/O configuration.

DCM simplifies the I/O configuration by letting you identify channels that are to be “managed”. Managed channels are channels that are not assigned to any specific control unit, but instead are viewed as a pool of channels, or channel path IDs (CHPIDs). Managed channels can be assigned dynamically to control units at the discretion of the system. This simplifies I/O configuration planning and definition. Channel resources can be changed dynamically in response to varying workload demands and therefore used more efficiently.

DCM support for Fiber Connection (FICON) channels was introduced as part of z/OS V1R11. Now, for z/OS V2R1, DCM support for cascaded FICON channels is introduced.

Cascaded FICON
Cascaded switches are useful when there are multiple sites that need to connect to the same set of direct access storage devices (DASDs). Without cascading, you would need to purchase many more expensive, extended-distance fiber optic cables, because the processors at one site would need to be able to connect to the switches at the other site, and vice versa.

With cascaded switches, the processors at each site only need to connect to their own switches. The switches at each site are then connected to the other sites switches through inter-switch links (ISLs). Although extended-distance fiber optic cables are required for the ISLs, the number needed is much smaller. The number of channels required at each site is also smaller.

DCM concepts
To understand DCM, you should become familiar with the following concepts:

- Non-managed channel
- Managed channel
- Managed control unit
- Non-managed control unit
- I/O cluster
- Control unit group

In addition, you should be familiar with configuring the I/O configuration to define managed channels and control units. For more information about these concepts, see “Channel surfing: Give DCM the remote” in z/OS Hot Topics Newsletter Issue 22, March 2010, GA22-7501-18.

Requirements for FICON DCM with cascaded switch support
Prior to z/OS V2R1, DCM only supported adding managed channels to control units if the channel was connected to the same switch as the control unit. That is, there was a single switch between the channel and the control unit. In z/OS V2R1, DCM is extended to support cascaded switch configurations.

DCM builds tables that represent the topology of the system. To obtain topology information for each switch, the Control Unit Port (CUP) feature must be installed on each switch. For FICON DCM cascaded support, the CUP feature must be installed on the entry switch connected to the channel, the exit switch connected to the control unit, and any intermediate switches that are being used as channel extenders. This allows DCM to determine which managed channels and control units are part of the same switch fabric. DCM can only add a managed channel to a control unit if they are part of the same fabric.

Additionally, in order for DCM to be able to communicate with all of the switches in a cascaded switch topology, the CUP devices must be defined with two-byte link addresses. When a FICON CHPID is connected to a control unit using a two-byte link address, all control units using that CHPID must also specify two-byte link addresses. Therefore, you might need to update your I/O configuration for all control units that are sharing the CHPIDs used for the CUP devices to use two-byte link addresses.

z/OS V2R1 does not need to be installed on all systems to exploit DCM for cascaded fabrics. However, only DCM at the z/OS V2R1 level can make changes in a cascaded configuration where the control unit is attached to the exit switch. The control unit on prior releases of z/OS is not eligible for DCM because it is not attached to an entry switch.

p/OS V2R1 does not need to be installed on all systems to exploit DCM for cascaded fabrics. However, only DCM at the z/OS V2R1 level can make changes in a cascaded configuration where the control unit is attached to the exit switch. The control unit on prior releases of z/OS is not eligible for DCM because it is not attached to an entry switch. 

Howard Baker is a Certified Executive Project Manager for the IBM/Marist College Joint Study program. He received his MS Degree in Computer Science from Union College. He joined IBM in 1982 as a software engineer and worked on mainframe software, and managed several software development projects. For the past 12 years, he has worked on several innovative projects, including first of a kind project focused on emerging technologies.

Reginaldo Barosa joined IBM in 1973 and was one of the first Certified IT Specialists from IBM Brazil in 1995. He was first assigned to the International Electrical Engineering from the University of Michigan.

Stephen Branch is an IBM Senior Software Engineer at Silicon Valley Lab whose 25-year career has dealt with all aspects of DFSMS. Stephen specializes in ICF Catalog, IDCAMS, and VSAM and holds several patents for these components. He is the author of the Catalog Search Interface (CSI) as well as many ICF Catalog and VSAM functions. Stephen has a Master of Science Degree from Louisiana State University.

Chris Brooker is an Advisory Software Engineer for IBM zAware. He holds an M.S. degree in Software Development from Marist College in Poughkeepsie, New York. Chris has spent most of his 12-year career with IBM in z/OS Global Resource Serialization (GRS) development.

Edward W. Chencinski is a Senior Technical Staff Member and chief engineer of an I/O subsystem development team. He received his B.S. degree in electrical engineering from Lehigh University. He joined IBM in 1980 working on the ES/3090® and later the G3/ G4 CMOS cryptographic hardware processor design team, focusing on pervasive functions, simulation, timing, and modular exponentiation. He performs high-level design of numerous elements within the System z I/O subsystem.

Bryan Childs is a Senior Software Engineer in z/OS Global Resource Serialization (GRS) Development and is the GRS Development Team Lead. Bryan holds an M.S. degree in Software Development from Marist College and has been with IBM for over 20 years.

Jim Coon is an Advisory Engineer. He received his B.S. degree in Mathematics from Lyndon State College and his M.S. in Computer Science from the University of Tennessee. Jim has worked in the areas of remote job entry, retail applications, digital certificates, intrusion detection and prevention, and smart card technologies.

Jonathan Cottrell, CISSP®, is an Advisory Software Engineer on the z/OS System SSL team since 2011. Previously he worked on the z/OS LDAP team. Jon has a Bachelor of Science degree in Computer Engineering from Clarkson University.

Anuja Deedwaniya is a Senior Technical Staff Member and software architect for z/OS Simplification and System Management Initiative. She has experience in various areas of mainframe design and development and is the architect and designer for IBM z/OS Management Facility.

Robert Fairie is an Advisory Software Engineer on the z/OS System SSL team. When not developing new and improved support for System SSL, Robert brews his own beer, gardens, enjoys long walks on the beach, and flower arrangement. He also designs and builds trebuchets.

Robert Fuga is an Advisory Engineer in z/OS System Test. His expertise is in the areas of HCD, I/O configurations, coupling facilities, z/OS I/O recovery, and z/Flash. Robert has been with IBM for 31 years.

Garth Godfrey is a Senior Software Engineer in the IBM zAware development team at IBM Poughkeepsie. With 33 years of IBM development experience, Garth’s expertise includes IBM zAware, z/OS UNIX, and z/OS Ported Tools. Garth holds a bachelor’s degree in System Software Science from RIT.

Joseph Gentile is a Software Engineer in z/OS Global Resource Serialization (GRS) Function Test. He holds a B.S. degree in Computer Science from Binghamton University and is currently pursuing his M.S. in Computer Science at SUNY College at New Paltz.

Nick Jones is a Software Engineer for the BCP at IBM Poughkeepsie for the last 10 years. His work includes design, test and service for System Logger, Service Aids including IPS/IOS Basic Hyperswap®, and most recently SMF.

Steven B Jones is a Senior Software Engineer. He has been working in the Job Management area since 2001 and with IBM since 1987.

Dick Gusefski is an Advisory Engineer for IBM Systems and Technology Group at 1701 North St. Endcott, New York 13760. He received his B.S. Degree in electrical engineering from Rochester Institute of Technology. He joined IBM in 1985. He worked as a logic designer, designing cache chips for the mid-range servers. He later joined the SE/HMC team writing firmware for the channel subsystem for the Support element.

Eric Harris is a software engineer in Tucson, AZ. He provides Level 2 support for DFSMS for the Catalog and DCGAMS components. Eric holds two degrees in Business and Engineering from Northern Arizona University.

Mark Hodsman is a software engineer with IBM Software Group in Canada. He joined IBM in 2007 and works on the InfoSphere Change Data Capture for z/OS product. Mark holds a Bachelor of Science degree in Mathematics from McGill University.

David Horn is an Advisory Software Engineer for the IBM Systems & Technology Group at IBM Poughkeepsie. He has been with IBM for 24 years.

Jim Coon is an Advisory Software Engineer at IBM Poughkeepsie Development Lab and currently a member of the ITDS for z/OS development team. His expertise includes encryption and Java security. Saheem is with IBM for 11 years and holds a Master’s Degree from Binghamton University.

Mark Hodsman is a Senior Software Engineer in z/OS Product Portability at IBM Poughkeepsie. He has 12 years of experience with IBM in z/OS product development. Mark holds a Bachelor of Science degree in Mathematics from McGill University.

Garth Godfrey is a Senior Software Engineer in the IBM zAware development team at IBM Poughkeepsie. With 33 years of IBM development experience, Garth’s expertise includes IBM zAware, z/OS UNIX, and z/OS Ported Tools. Garth holds a bachelor’s degree in System Software Science from RIT.

Joseph Gentile is a Software Engineer in z/OS Global Resource Serialization (GRS) Function Test. He holds a B.S. degree in Computer Science from Binghamton University and is currently pursuing his M.S. in Computer Science at SUNY College at New Paltz.

Dick Gusefski is an Advisory Engineer for IBM Systems and Technology Group at 1701 North St. Endcott, New York 13760. He received his B.S. Degree in electrical engineering from Rochester Institute of Technology. He joined IBM in 1985. He worked as a logic designer, designing cache chips for the mid-range servers. He later joined the SE/HMC team writing firmware for the channel subsystem for the Support element.

Eric Harris is a software engineer in Tucson, AZ. He provides Level 2 support for DFSMS for the Catalog and DCGAMS components. Eric holds two degrees in Business and Engineering from Northern Arizona University.

Steven Hart, CISSP®, is an Advisory Software Engineer for z/OS Cryptographic Services in Poughkeepsie, New York. In his ten years with IBM, he has worked on Integrated Cryptographic Services Facility (ICSF), Trusted Key Entry (TKE), Encryption Facility for z/OS, Tivoli Key Lifecycle Manager (TKLM) for z/OS, and on Cryptographic Coprocessor firmware. Currently he works on ICSF strategy, design, and development.

Mark Hodsman is a software engineer with IBM Software Group in Canada. He joined IBM in 2007 and works on the InfoSphere Change Data Capture for z/OS product. Mark holds a Bachelor of Science degree in Mathematics from McGill University.

Dick Horn is an Advisory Software Engineer for the IBM Systems & Technology Group at IBM Poughkeepsie. He has been with IBM for 24 years.
Kevin Kelley is a very well regarded software engineer and student pilot. He has been with IBM since 1974.

Shigeki Kimura is the technical lead for z/OS migration at IBM Japan. He has expertise in and deep knowledge of z/OS release-to-release migration and has participated in key reviews of z/OS Migration, GA22-7409, since 2006. Shigeki has been with IBM for 27 years.

Stephen M. Kocik is an Advisory Software Engineer for I/O Supervisor at IBM Poughkeepsie, NY. Stephen is the lead developer for FCICS DCM. He has been with IBM for ten years.

Juergen Kuhn has been with IBM at the Boeblingen Laboratory for 23 years. During his IBM career, he has worked for development projects in z/OS, z/VSE®, and Cloud Computing. He is currently leading the z/OS Resource Measurement Facility (RMF) development team.

Jim LaBarge is a System Tester for Systems and Technology Group at IBM Poughkeepsie. His expertise includes service aids, zAware, and Predictive Failure Analysis. Jim holds a computer science degree from SUNY Potsdam. Jim has been with IBM for 14 years, first on the AT&T account with Global Business Services, and now in his current position in STG.

Marie R Laser is an Advisory Software Engineer in the z/OS development group. Since joining IBM in 2005 her focus has been on Language Environment and user interface development.

Jason C Lee is an Advisory Software Engineer, z/OS Catalog Development. He holds a B.S. degree in Electrical Engineering and Computer Science from University of California, Berkeley. Jason has been with IBM since 2004 in both DFSMS RLS and ICF Catalog.

Cecilia Carranza Lewis is a Senior Technical Staff Member for z/OS DFSMS at IBM Silicon Valley Lab. Cecilia has been with DFSMS for 29 years. She drives technical design/development in the area of z/OS data/storage management. She has led various projects such as EAV, sequential striping, and compression.

Jeff Magdall is a Project Executive in IBM Systems and Technology Group and is currently the z/OS z/TPF Leader for System z, in Poughkeepsie, NY.

Peter Mailand is a software engineer at IBM System and Technology Germany. He joined IBM in 1996 as a member of the R/3 development team and has 17 years of experience in the performance monitoring area.

Daniel Martin is a software engineer at IBM Software Group in Germany. He joined IBM in 2004 and works for the IBM DB2 Analytics Accelerator team since 2010. Daniel holds a university doctoral degree in computer science from the University of Stuttgart.

Charles Mari is an Advisory Software Engineer in the z/OS Core Technologies at IBM Poughkeepsie. He currently works in the Storage Management area of z/OS.

Sean McMillen is an Advisory Software Engineer who has been with DFSM/SHsm for over 10 years. Sean is a frequent contributor to z/OS Hot Topics. He resides in San Diego and enjoys spending time with his son Jacob and the beach.

Terri Menendez is Senior Technical Staff Member, z/OS VSAM/Catalog Architect. Terri has a Bachelor of Science degree in Mathematics with a concentration in Computer Science from San Jose State University. Terri has worked for IBM since 1984. He was the chief architect for the RLS for Catalog enhancement in z/OS V2R1.

Lucy Miller is a Senior Solution Architect for the Customized Offerings (ServerPac and CBPDO) for IBM BT/TIO CIO in Poughkeepsie, NY. She has been involved with ordering, manufacturing, delivery, and installation of z/OS software, including requirements in these areas, for many years.

Robert Miller is an Advisory Software Engineer in z/OS Storage Management Test at IBM Poughkeepsie. He has been with IBM for 23 years.

John Monti is a Senior Software Engineer for z/OS Language Environment has worked for IBM for over 20 years and is a frequent speaker at SHARE.

Ken Oakes is a Senior Technical Staff Member in the Systems and Technology Group. He has held various technical and technical leadership positions in the zSeries I/O design area. He is currently involved in I/O subsystem design for the next-generation zSeries server.

Steve Partlow is an Advisory Software Engineer in the z/OS Core Technologies at IBM Poughkeepsie. He currently works in the Storage Management area of z/OS.

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.

Kathy Pfeiffer is a Senior Software Engineer at IBM in Poughkeepsie, New York. She is the Academic Initiative System z program manager.

Dale Riedy is a Senior Technical Staff Member in the System z Operating Systems Development organization in Poughkeepsie, New York. He is an expert in the z/OS I/O Supervisor. Dale has been with IBM for over 30 years.

Don Resnik is a Senior Software Engineer at IBM in Poughkeepsie, New York. He is the System z Client Skills and Academic Initiative lead.

Mary Rezeski is an Advisory Software Engineer, Mary has worked on the z/OS SSL Team since mid-2009. Previously, she worked on the Java team. Mary holds a BS Degree in Mathematics and an MS Degree in Systems Science.

Anthony Sofia is a Software Engineer at IBM Poughkeepsie. His expertise includes the new zEnterprise Data Compression offering, the z/OS I/O Subsystem and the z/OS System Management Facilities. He has been with IBM for 9 years and enjoys spending his free time with his wife and son.

Nat Stevenson has worked for 10 years, providing z/OS L2/L3 defect technical support in Poughkeepsie, NY and Gaithersburg MD. He has spent the last 20 years working providing “z/OS Core” support at the Washington Systems Center (WSC) in Gaithersburg, MD.

James Sweeney is the Design Lead for the ICSF Enterprise PKCS #11 services on z/OS. He has worked on numerous security and middleware components, including Integrated Cryptographic Services Facility (ICSF), Resource Access Control Facility (RACF), and Public Key Infrastructure Services (PKI) for z/OS.

Peter Szved is a Senior Engineer in the Systems and Technology Group. He has held a wide variety of development roles including ASIC designer for IBM’s supercomputer offerings, architecture author for System z and System p, and I/O firmware developer for System z PCI. He was a member of the system design team for Flash Express and is currently team leader for Parallel Sysplex firmware development.

Ann Totten is a senior software engineer for z/OS System Verification Test at Poughkeepsie. She has expertise in z/OS UNIX file systems, zFS, and shells and utilities. Ann has a Masters of Science degree from Union College. She has been with IBM for 24 years.

Scott Tuttle has been with IBM for over 35 years and is currently a software developer at the IBM Poughkeepsie Lab. He has worked in many of the z/OS components, including GRIS, XCF/XES, consoles, allocation, supervisor, and RSM.

Elpida Tzortzatos is a Senior Technical Staff Member in the z/OS Core Technologies department at IBM in Poughkeepsie, New York where she leads the Storage Management Design and Development teams (RSM, VSM, IASM). She has been instrumental in the delivery of several key enhancements in the memory management area, such as support for 64-bit, large page, and flash memory.

Girija Varanasi is currently a programmer in the BCP Service Aids team. Prior to that, she worked in MSYS for Setup and Job Management areas. She has been with IBM since 2001.

Marna Wallister is an IBM Senior Technical Staff Member in the z/OS Build and Installation department at IBM Poughkeepsie. Her current responsibilities include z/OS release migration and new installation technologies of the z/OS platform. Marna is a frequent speaker on z/OS installation topics at EXPO, SHARE, and other customer conferences.

Steve Warren is a Senior Hardware Designer and Developer for the BCPi component of z/OS in IBM Poughkeepsie. Steve has been with IBM for 23 years.

David Whitney is a z/OS developer for Timer Supervisor at IBM Poughkeepsie. His expertise includes ETR, STP, and Context Services. Dave holds a PhD degree from the University of Delaware.

Glenn Wilcock is a Senior Technical Staff Member with IBM Systems and Technology Group, in Tucson, Arizona. He is the lead architect for z/OS DFSM/SHsm and has over 20 years of experience developing storage software solutions on System z.

Stefan Wirag is a senior software engineer at the IBM lab in Boeblingen. His expertise includes z/OS workload management, capacity provisioning, and Web 2.0 technology. Stefan joined IBM in 1999.

Ryan Wisniewski is an advisory software engineer in DFSM/SHsm Level 2 Support at the Tucson, Arizona, lab specializing in Catalog and IDCAMS. Ryan holds a Computer Science and Accountancy Degree from Northern Illinois University.
Be the master of your fate
Over 4,000 students and experienced professionals from 40 countries have taken the IBM System z mastery test. Have you?

The IBM System z and z/OS Fundamentals Mastery Test (Test #000-Z05) is another indicator that the enterprise systems ecosystem continues to grow with more mainframe skills. If you're new to the mainframe or know of others at your business who are learning the mainframe, the IBM System z Mastery Test is a great way to validate and gain official recognition of your technical knowledge and expertise.

The test, developed by IBM, IBM Clients, and IT professors is recognized, worldwide, as a reliable, valid, and fair method of assessing entry level criteria for those new to the mainframe.

The System z mastery test is a proctored test and is available worldwide at over 5,000 authorized Prometric testing locations. The format of the test is multiple choice. For more test details visit the following site:

[ibm.com/certify/mastery_tests/ovrZ05.shtml](http://ibm.com/certify/mastery_tests/ovrZ05.shtml)

Passing a mastery exam is a tremendous accomplishment. What are you waiting for? Master the Mainframe!

Clarifications, corrections, and announcements

Visit the Hot Topics website for clarifications and corrections to articles from our past issues:

[ibm.com/systems/z/OS/zos/bkserv/hot_topics.html](http://ibm.com/systems/z/OS/zos/bkserv/hot_topics.html)
Glossary for network administrators

Starting with z/OS V2R1, z/OS Communications Server provides you with a comprehensive glossary that network administrators will find indispensable! This information integrates glossaries from z/OS Networking, VTAM, and TCP/IP. The glossary consists of approximately 4,000 terms to help you find the meaning of almost any term in the z/OS Communications Server documentation.

Find a copy of our new glossary in the z/OS V2R1 Information center at: ibm.com/systems/z/os/zos/bkserv/

Can you find a term that we missed? Write us and tell us what you find at: comsvrcf@us.ibm.com