Security in CAMSS

Cloud · Analytics · Mobile · Social · Security

z/OS Security
Mike Onghena
Bruce Wells
Eric Rosenfeld
Featured on the cover of z/OS Hot Topics are three of our top designers of IBM Security software: Mike Onghena (upper right), Bruce Wells (lower left), and Eric Rosenfeld (lower right). Their article, “Don’t Fall on your p@ssword”, focuses on the evolution of password controls from 2005 to the present. Their article is central to our theme: Putting Security into Cloud, Analytics, Mobile, Social, and Security (CAMSS).

We continue the security theme with the following security-related articles:

• “Secure, but not foolproof” by Mike Kasper and Mike Spiegel
• “Your order’s up! RACF client requirements satisfied in z/OS V2R2” by Bob Gensler, Laurie Ward, and Scott Woolley
• “Erasure and encryption: the yin and yang of security technologies” by Mark Nelson, John Paveza, and William C. Johnston
• “Drowning in digital certificates? Here’s a lifeline!” by Wai Choi
• “Give credit to Crypto; it gives Crypto to credit! Protecting credit card data with Visa FPE” by Steven R. Hart and Rita Beisel
• “Fortify your SMF data with digital signatures” by Anthony T. Sofia, Colin D. Chen, and Heather M. Bosko
• “Transform your data into a PDF file to share across the cloud” by Anthony Mingo, Pat Glenski, and Tariq Choudhry

While the theme of the issue is Security, we continue to focus on the other platform technologies that enable digital transformation, evolution, and expansion: Cloud, Analytics, Mobile, and Social. For example, try the following:

• Pull up “z System in a mobile world” on your smartphone from our website: ibm.co/1lgC54R. Or scan the QR code on the back cover to access our newsletters.
• Like us on Facebook: facebook.com/zosHotTopics.

And that’s not all! Take a look at the table of contents to see all of the wonderful articles in the current issue.

We hope you find this issue of Hot Topics entertaining and informative. We look forward to hearing from you!

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

4 Don’t fall on your p@ssword
BY ERIC ROSENFIELD, BRUCE WELLS, AND MIKE ONGHENA

6 Secure, but not foolproof
BY MIKE KASPER AND MIKE SPIEGEL

8 Your order’s up! RACF client requirements satisfied in z/OS V2R2
BY BOB GENSLER, LAURIE WARD, AND SCOTT WOOLLEY

10 Erasure and encryption: the yin and yang of security technologies
BY MARK NELSON, JOHN PAVEZA, AND WILLIAM C. JOHNSTON

11 Quarantine!
Improved identification of unhealthy servers with Health Based Routing
BY JESSIE YU, HORST SINRAM, AND KARLA ARNDT

14 Give credit to Crypto; it gives Crypto to credit!
Protecting credit card data with Visa FPE
BY STEVEN R. HART AND RITA BEISEL

16 Fortify your SMF data with digital signatures
BY ANTHONY SOFIA, COLIN CHEN, AND HEATHER DENNIS

18 IBM Java 8 performance on z13
Hardware and software codeign at its finest
BY ELTON DESOLUIZA, IRIS BARON, AND CLARK GOODRICH

21 Mega memory on z13
5 steps to mega value!
BY PETER SUTTON

24 z/OSMF: all about the base
Now a base element of z/OS V2R2
BY ZHEN HUA DONG AND ROLANDO PEREZ

26 REST easy on z/OS
Introducing the z/OS Client Web Enablement Toolkit
BY STEVE WARREN

28 Master the mainframe turns 10
BY KATHY PFEIFFER AND TROY CRUTCHER

30 Track down GRS requests with SMF
BY JOE GENTILE AND CHRIS BROOKER

32 How do you spell relief? (Not IEFUSI!)
Easter region management
BY BONNIE ORDONZ, SCOTT BALLENTINE, AND STEVEN JONES

34 Problems, problems
Get started on problem determination
BY RITU BHARGAVA, JOHN SHEBEY, AND BARBARA NEUMANN

36 Get your logrec moving
Dynamic logrec recording
BY NICHOLAS JONES

37 Heightened zAwareness
Introducing cutting edge analysis
BY CHRIS BROOKER, LISA J. CASE, AND YUNLI TANG

40 Quarantine!
Improved identification of unhealthy servers with Health Based Routing
BY JESSIE YU, HORST SINRAM, AND KARLA ARNDT

42 GDG: the next generation
Space, the final frontier
BY STEPHEN BRANCH

43 Sharing the wealth with DFSMSHsm
BY JAMES CAMMARATA, SNEHAL BHAKTA, AND SHANNON GALLAHER

46 Improved z/OS subsystem management
BY SCOTT BALLENTINE

47 Keeping an eye on VTOC updates
BY JOHN PAVEZA AND CARRIE VAN NOORDEN

48 Let your system do the tuning!
BY DAVID SHACKELFORD, EDUARD DIET, AND SCOTT COMPTON

51 Take the write route with z/OS zHyperWrite
BY RON BRETSCHEINDER, SCOTT COMPTON, AND TARIQ HANIF

53 Another mirror of my data? Isn’t one enough?!
BY COREY SMEATON, LISA GUNDY, AND WARREN STANLEY

56 Simplify z13 maintenance with SMP/E
BY GREG DAYNES

58 Is life in the fast lane for you? z/OS SMC-R Applicability Tools
BY DAVID HERR, GUS KASSIMIS, AND JERRY STEVENS

60 Hints, tips, and best practices for z/OS V2R2 documentation
BY SUSAN SHUMWAY, RITA BEISEL, AND TOSHIBA BURNS-JOHNSON

62 Take control with ad hoc batch scheduling
BY BRUCE TALBOTT, KEVIN KATHMANN, AND TOM WASIK

66 Share the road
SMT-2 on z13
BY KEVIN MCKENZIE, DANIEL ROSA, AND HORST SINRAM

69 Transform your data into a PDF file to share across the cloud
BY ANTHONY MINGO, PAT GLENSKI, AND TARIQ CHOUDHRY

72 Our contributors
Passwords have been used to allow system access since the early days of computing. And even though there are a number of other methods of authenticating users, passwords are still relevant in today’s cloud-oriented world. You want to avoid having to fall on your “p@ssword” — as the person whose account was hacked because of a weak password, or as a security administrator whose policy and controls allowed that to happen. Now, new password controls are available to help RACF® administrators.

RACF functions related to z/OS passwords—then and now

Over time, updates were included in RACF to enhance password security. By default, passwords rules were that a password must be 1 - 8 characters and consist of uppercase, numeric, and national characters “#”, “$” and “@”. In 2005, z/OS® V1R7 added the ability to allow lowercase characters.

The next release introduced the password phrase, which can be used as an alternative authenticator. A password phrase was defined as a 14 - 100 character string. Any character that could be entered from a keyboard for use by a command could be specified, subject to a number of built-in syntax rules. z/OS V1R9 shortened the minimum length to 9 characters when the new phrase exit, ICHPWX11, allowed it. A REXX-based sample exit was provided.

What have you done with passwords lately?

In November 2014, the PTF for APAR OA43999 introduced several updates (available on z/OS V1R12, z/OS V1R13, and z/OS V2R1) to strengthen the security of RACF passwords and phrases:

• Support for 14 additional characters in passwords. Using these characters increases the available password space.

• The ability to have a password phrase without requiring a password. You can use this feature to force users to use longer passwords.

• The addition of a new hashing algorithm that is used when storing and verifying RACF passwords—key derivation function with Advanced Encryption Standard (KDFAES, described below).

• A new ALTUSER keyword to expire a user’s password and phrase immediately. Issuing the ALTUSER keyword to expire every user who is defined to RACF might be a good first step to take if the RACF database is stolen. Users will be required to change their passwords the next time they log on.

What have the bad guys done with passwords lately?

With the expanded role of the mainframe in the connected world comes a stronger need to protect security definitions from outside eyes. The RACF database should be one of the most highly-secured resources in an installation. The theft of a RACF database is a very serious breach of an installation’s security.

A well-configured RACF database will revoke a user password if there are more than a handful of failed attempts to validate a password. This is very effective in preventing a simple password-guessing attack. Guess…no...Guess...no...Guess... revoked...done.

However, if a RACF database is stolen, the thief has copies of all the password hashes. The thief simply needs to write a program to implement the same password-hashing algorithm, then guess a password to see if it matches the hashed value from the database. If it doesn’t match, try another. Because the thief is performing the hashing and verification, the RACF revoke count no longer comes into play. This last point is very important. The thief has all the data and the ability to guess a given password without RACF being involved. The attack usually occurs on a non-z/OS system.

The source of another password threat is the use of specialized hardware to attack passwords. Decades ago, specialized hardware was something that was priced at the ‘government’ level. Over time, the prices decreased to a level that is attainable by many companies, as well as organized criminals. Today, the price is in the “kid-with-a-summer-job” range. Just add a high-end video game graphics add-on card to a PC, install some free software, and you have a potent password-cracking machine. The more graphics cards that you add, the faster you can guess passwords.

One way to counter this type of attack is to slow the speed of password guesses. You can do this by using a password-hashing algorithm that takes much longer to complete. Part of such an algorithm should incorporate iterative hashing and salting, which is the injection...
of random data into the password-hashing procedure. Salting prevents the use of rainbow tables, which are lists of pre-computed hashes that can be compared against the stolen hashes. The new KDFAES algorithm incorporates these protections.

Although RACF pays an additional password evaluation cost for each logon, a cracker incurs this cost trillions of times while attempting to guess passwords.

A reasonable goal is to make the password change interval shorter than the time required to guess passwords.

What can you do to improve password quality?

No encryption is strong enough to protect you, however, if your password is “PASSWORD”, so a complementary defense to strong encryption is to increase the quality of passwords. This approach requires the hacker to try more and more passwords before hitting the correct password. You can increase the quality of passwords by increasing the password length, increasing the characters allowed for use in passwords, and by enforcing rules that prevent the use of passwords that are easy to guess.

The PTF for OA43999 provides the ability to define phrase-only user IDs, which should be your first approach to requiring strong passwords. With password phrases, it is easier to choose a longer value that is easy to remember, but hard to guess. For example, you could start with a favorite movie quote or literary passage and then inject some misspellings or rearrangement of words to stay a step ahead of the crackers. As an administrator, you can use the REXX-based new-phrase exit, ICHPWX11, to enforce a minimum length higher than the default and to impose other rules to make these values harder to guess (see the IRRPHREX member in SYS1.SAMPLIB).

RACF also provides functions that can help you maximize the security of traditional passwords. The first option that you should consider is mixed-case passwords. Enabling mixed case increases the number of valid password characters from 39 to 65, which greatly increases the work factor of a brute-force attack.

In addition, the PTF for OA43999 adds the ability to use 14 additional special characters, bringing the total number of valid characters up to 79. The PTF for OA43999 also enhances the RACF password syntax rules with the MIXEDALL feature, which can require a mixture of up to four different character types (uppercase letters, lowercase letters, digits, and symbolic characters).

A purely brute-force attack is now the last resort for password crackers. Over the years, they have accumulated lists of modified dictionary words (with “e” replaced by “3”, and so on) including actual password lists from hacked websites. From this data, password crackers have discovered simple patterns that are used by lazy users when choosing passwords; for example, a password that starts with an English word with the first letter capitalized, then a digit or two, and ending with a symbolic (usually the exclamation mark). “Bruce01!” would satisfy such a rule, and when that value expires, “Bruce02!” will do just as well! This underscores another weakness password crackers take advantage of: discernable patterns in password history that can be used to guess your current value. Crackers essentially have an infinite amount of time to crack the history.

Never fear, there is also a REXX-based password exit, available on the RACF website (ibm.biz/BdXGW4) that provides the ability to easily restrict such values. Not only can the exit apply rules to the chosen new value, it can also compare this value with the previous value to make sure that it differs meaningfully. The goal is to train your users to choose a password that appears entirely random, but has some personal meaning that allows the user to remember it without writing it down.

How does this apply in practice?

Now that we know about all of this support, what now? In a perfect world, all users would only have password phrases at least 14 characters in length. System settings would be configured to allow all possible password characters for users of applications that don’t support phrases, and an intelligent new password exit would require strong passwords. Password and password phrase histories would be maintained with many entries, and IDs would be revoked after a low number of invalid password guesses. And last, but not least, KDFAES would be activated as the encryption method in use on the system. Passwords and password histories that are hashed with the older Data Encryption Standard (DES) algorithm would be converted to the KDFAES format using the new PWCONVERT keyword of the ALTUSER command, also introduced with the PTF for OA43999.

Password evolution

Password security has evolved over time. z/OS processing has been updated to keep up with the increasing need to make passwords more secure.


Updates in password policy can take advantage of these new features to strengthen the passwords that are used for authenticating to your system. What is in store for passwords in the future? As the computing environment changes, we will see how these authenticators, and the attacks against them, evolve.
Don’t get us wrong. We’re not saying that security mistakes are common when configuring z/OS systems. We’re just saying that we have seen security mistakes made, and we want to talk about some of the common oversights. Even though we believe that you can configure the z/OS system to be the most secure system in the world, it is not always configured that way.

TCP/IP
You might not realize that users who can send and receive data to z/OS host systems using FTP also can submit jobs to those host systems using FTP. The FTP subcommand SITE FILETYPE=JES allows an FTP user to submit a job to a z/OS system that a user has authenticated to by using an FTP logon. z/OS systems are commonly set up so that users are, at minimum, given the authority to submit batch jobs with a job name that contains their user ID.

Users can be prevented, through the JESJOBS RACF class, from submitting any jobs. However, this action is not typically done because it is expected that authenticated users should be able to submit jobs. Yet, in the case of FTP users, this might not be what your installation intended. You might only intend that your FTP users can send and receive data through FTP but not submit jobs. FTP does not provide a configuration setting that you can use to prevent people from using the SITE FILETYPE=JES subcommand. If you want to restrict FTP users, use the JESJOBS RACF class. If you were considering allowing anonymous FTP on your z/OS system, be careful how you do this so that anonymous users cannot submit jobs.

Similarly, some systems are not protected from direct access over the public Internet. The reasons for restricting direct access with a firewall, authentication, and other methods such as IP filtering should be obvious. If direct access is not restricted, your z/OS system is opened up to potential attacks.

SSL/TLS
For z/OS systems to support strong encryption, you should order the two Security Level 3 features (at no cost) and follow their configuration steps to activate them. Those features are z/OS Security Level 3 and Communications Server Security Level 3. Customers who are security-conscious are already taking these additional steps. However, if your installation is not taking those steps, data is being exposed. A growing list of publicly disclosed security vulnerabilities in various Secure Sockets Layer (SSL) protocols, for example, suggests using Transport Layer Security (TLS 1.2) or later.

RACF
On z/OS V2R1 or earlier, RACF installations without an ICHDEX01 exit could still be using a masking algorithm to verify passwords if the Data Encryption Standard (DES) encryption algorithm check fails and key derivation function with AES...
(KDFAES) is not active. Even worse, if the sample ICHDEX01 exit is used, it specifies masking, not DES, and all password hashes remain masked. Access to the RACF database would be needed to read them, but masking is not encryption. It is slightly better than plain text, but not by much.

For password checks, use encryption, not masking. The PTF for RACF APAR OA43999 allows for KDFAES 256-bit encryption of RACF passwords hashes, which offers much stronger encryption than is possible with DES encryption. Masking is no longer used for a default check as of z/OS V2R2.

**TSO/E**

You might still be using the SYS1.UADS data set, not RACF, for managing access to TSO/E. While you probably maintain UACC NONE for this data set, remember that the SYS1.UADS data set is not encrypted. Anyone who can copy this data set can see all of the user IDs and passwords in plain text. The only user IDs that are kept in the SYS1.UADS data set should be emergency user IDs that are prevented by RACF from entering the system so the only time they can log on to the system is when RACF is down.

**z/OS UNIX**

Public UNIX directories such as /tmp and /var are typically open to use by any user on a UNIX or Linux system and are commonly used as a temporary repository by users and UNIX applications. To prevent the security of users and systems from being compromised, it is important to implement some simple but sometimes overlooked controls for public directories. On z/OS UNIX, both the /tmp and /var directories are configured at system startup time to have the sticky bit set. The sticky bit prevents the files and directories that are created by a user in these public directories from being removed or deleted by another user. To prevent accidental or intentional misuse of the files and directories in public directories, ensure that all public directories on your system are configured with the sticky bit set.

The /tmp and /var directories can also contain executable programs or execs that users share with other users. To indicate that they are not secure for privileged program execution, mount the file systems for the /tmp and /var directories with the nosetuid option. Also, ensure that all public directories are in file systems mounted with the nosetuid option. This also holds true for user file systems. Ensure that the automount policy for user directories is set up this way.

Some sound practices relating to the z/OS UNIX executable programs installed on your system exist that you might overlook. Many z/OS UNIX software applications use directories that are installed specifically for the software application. Often, a new version of a UNIX software application is installed into a new directory for the latest version of the software. For example, NetView® for z/OS Version 5 Release 2 installs the software into the /usr/lpp/netview/V5R2 directory, and each newer version or release installs the software into its own directory (V5R3, V5R4, and so on).

After a new version of a product is installed and becomes the version in use on the system, the fixes that are applied to the product are typically done only for the version of the product that is being used. The executable programs that are still in the older directories do not receive the latest fixes for a product, including critical security fixes. To prevent potential exploitation of these exposures, unmount any old software product directories that are not being maintained or are for products that are no longer being used. The recommended configuration for the file systems that contain the files provided by IBM® software products and executable programs is to mount these file systems (/usr/lpp, /bin, /usr/sbin, and so on) as read-only to prevent these files from being unintentionally updated. Take the same action for all software products that you install in your z/OS UNIX file system, including your own software.

As a rule, minimize the number of users on their systems that have a UID of 0. Use the BPX.SUPERUSER profile to give select users the ability to temporarily switch to superuser authority when necessary rather than having each privileged user have a UID of 0. Additionally, use the SUPERUSER profile in the UNIXPRIV RACF class to give more selective superuser privilege.

**Security/integrity PTFs**

The described issues involve configuration settings for z/OS systems. Your security maintenance strategy is just as important. New security vulnerabilities are being found every day, and z/OS systems are not immune. IBM provides the z Systems™ Security Portal so IBM customers can be the first to know about any security vulnerability fixes for z/OS, how severe the problems are, and how to get PTFs. However, if you haven't applied maintenance in two years, knowing is only half the battle. You need a maintenance strategy for applying the security/integrity PTFs from IBM and the security vulnerability fixes for all the software vendors on your system.

You have now read about some of the most common security mistakes, but this is far from a complete list. To secure your systems even more, take advantage of a wide variety of security health checks that IBM provides. You can also use many other resources to help you find and prevent similar issues. Several security groups even provide top-ten lists of z/OS security configuration issues. IBM publishes white papers plus a series of IBM Redbooks® on the topic “Security on the IBM Mainframe”. If you are concerned about ensuring that your z/OS system is configured securely, make the following websites your next stop:

- www.redbooks.ibm.com/abstracts/sg247803.html
- ibm.com/systems/z/solutions/security_integrity.html
- web.nvd.nist.gov/view/ncp/repository/checklistDetail?id=55
Your order’s up!
RACF client requirements satisfied in z/OS V2R2

BY BOB GENSLER, LAURIE WARD, AND SCOTT WOolley

There is a large array of RACF client requirements that are satisfied in z/OS V2R2. Choose some of your favorites from our menu.

Menu

ROAUDIT: Heart-Smart AUDITOR
A Dark Suit glowers in the doorway. It’s system audit time.
The Dark Suit needs to run system utilities and reports personally to ensure the audit’s integrity. So begins the familiar drill: create a user ID with the system-wide AUDITOR attribute set, granting the Dark Suit authority to run the necessary utilities and reports.

Trouble is, the AUDITOR attribute also grants authority to actually change security settings. The last Dark Suit finger-fumbled a SETROPTS command. It took hours to repair the settings afterward.

Fear not, RACF comes to the rescue with the new system-wide ROAUDIT user attribute! ROAUDIT—Read-Only AUDITOR—authorizes a user to examine all the same RACF profiles, audit information and system controls similar to the AUDITOR attribute, but without granting the user ID the authority to alter system settings. So now, Dark Suits can perform their audits, and you can skip the antacid.

DIRSRCH: Overloaded nachos
As the RACF administrator, you carefully delegated authority to manage file ownership and permissions in z/OS UNIX System Services by giving some UNIX administrators UNIXPRIV SUPERUSER.FILESYS.CHANGEPERMS and SUPERUSER.FILESYS.CHOWN authority. But, the UNIX administrators are complaining your idea is half-baked. Those authorities do not include the directory read/search access that they need to navigate through file systems. You sure don’t want to give up the whole enchilada by granting SUPERUSER.FILESYS authority.

Skip the jalapeños - Now, give these users SUPERUSER.FILESYS.DIRSRCH authority, and watch as they search happily ever after.

RDEFINE UNIXPRIV SUPERUSER.FILESYS.DIRSRCH
UACC(NONE)
PERMIT SUPERUSER.FILESYS.DIRSRCH CLASS(UNIXPRIV)
ID(EMERIL) ACCESS(READ)

FSEEXEC: No permission on rye
Most z/OS UNIX System Services environments have a directory, such as /tmp, in which anyone can create files. It’s smart to mount these file systems with the nosetuid operand. Now RACF offers more protection. The RACF administrator can define a profile in the new FSEEXEC class to prevent any file in that file system from being run. So, if someone tries to throw a malicious program on the /tmp sandwich, they will leave hungry, being unable to run it:

RDEFINE FSEEXEC /tmp UACC(NONE)
**RRSF dynamic MAIN switch: The MAIN dish**

System programmers cringe at any suggestion about changing the MAIN system in a multisystem node for RACF Remote Sharing Facility (RRSF). The MAIN system is the one that receives all the commands from remote systems; how could all those RACF commands and RACF changes coming in and going out ever be "turned off" without losing any work?

Admitting you have a problem with your MAIN system is the first step in the existing unwieldy 11-step process to change that MAIN system. Sounds like another dreaded multi-step process, right?

Thankfully, z/OS V2R2 serves up the process for changing the MAIN system into a simple, ready-to-eat command.

For example, assume that you have the RRSF network in Figure 1. If multisystem node NODEABC is in a sysplex, you now need only one command to perform a dynamic MAIN switch, which can be issued from any system in the multisystem node (SYSA, SYSB, or SYSC):

```
TARGET PLEXNEWMAIN NODE(NODEABC) SYSNAME(SYSB)
```

Or, if multisystem node NODEABC is not in a sysplex, it's almost as simple. Issue just one command on each system in the multisystem node (SYSA, SYSB, and SYSC):

```
TARGET NEWMAIN NODE(NODEABC) SYSNAME(SYSB)
```

No commands are needed on the remote nodes (NODEXY and NODEZ) for the dynamic MAIN switch to take effect. It's as easy as pie!

**Nutrition facts**

Refer to the following publications for cooking instructions:

- Security Server RACF Command Language Reference, SA23-2292

![Figure 1. An RRSF network with two multisystem nodes and one single-system node](image-url)
Data is an interesting commodity. While the cost of its creation can vary greatly, the cost of its processing, duplication, and transmission continue to drop. As a result, more and more data is being processed, copied, stored and transmitted. Some security policies focus on access rules and the protection of data in motion and place a lesser emphasis on data at rest. Fortunately, there are two complementary data technologies that can help with protecting data at rest: erase-on-scratch and encryption.

**Erasure and Encryption: the Yin and Yang of Security Technologies**

BY MARK NELSON, JOHN PAVEZA, AND WILLIAM C. JOHNSTON

Erase-on-scratch

When a data set is deleted or space is released from the data set, the space is made available to other users. The data that was in that space is not erased unless you or your security administrator have asked that it be erased. The security administrator sets the overall erasure policy for your security product. For example, in RACF, you can use the set RACF options (SETROPTS) command to request that:

- No data set erasures are performed (SETROPTS NOERASE).
- SETROPTS ERASE indicates that the owners of profiles can specify what data sets are erased.
- SETROPTS ERASE(ALL) erases all data sets when they are deleted.
- SETROPTS ERASE(SECLEVEL(seclevel-name)) erases data sets with a specific security level or higher.

Only the SETROPTS ERASE(ALL) command erases everything that is deleted.

Note that SETROPTS ERASE applies to DASD data sets. Tape data sets are erased only if your tape management system is configured to perform the erasure.

The security administrator or the data set owner likely does not know all of the processing details when a data set is used. Is the data set for which erase-on-scratch was selected copied to another data set? If so, what are the erasure rules for the target data set? What if the data is copied to a temporary data set?

Data set erasure overwrites the space that is being deleted or released. For many years, not much changed with erase-on-scratch. Then, in September 2013, z/OS V2R1 provided a significant improvement in elapsed time and 90% reduction in I/Os. Instead of erasing one track at a time, z/OS erases up to 255 tracks in one command. Not satisfied with these results? In z/OS V2R2, up to 12,240 consecutive tracks in one channel program can be erased. If you use Peer to Peer Remote Copy (PPRC), enable the EOSV2 keyword of the DEVSUPxx member of SYS1.PARMLIB to take advantage of these improvements. The details are in PTF UA72887 for APAR OA46511. DOC APAR OA46511 identifies the required DS8000® microcode release levels.

Encryption

Data encryption addresses threats such as the malevolent or accidental loss of physical control of media or the movement of data within a storage device, as might be done for performance or recovery.

To encrypt data at rest, use an encryption-enabled device such as the IBM System Storage DS8000 disk storage or the IBM System Storage® TS1140 tape storage.

- The IBM DS8000 offers a self-encrypting disk that uses IBM Full Disk Encryption (FDE) to create and maintain the encryption keys. It can erases disk drives.
- The TS1140 devices encrypt tape-based data sets. When data is accessed, the drive contacts the Security Key Lifecycle Manager for an encryption key that can be used to process the data. The data is encrypted as it flows through the tape.

A second option is to use the IBM Encryption Facility. With it, you can use encryption keys that are maintained in either RACF or ICSF to encrypt specified data sets. To access the encrypted data set, you must have access to the RACF or ICSF key.

As you can see, erase-on-scratch and encryption of data at rest address two different sets of threats. When evaluating the risks associated with those threats, you need to evaluate those two technologies and, if needed, implement both.

For more information, see “Protecting data sets” in IBM Knowledge Center (ibm.co/1KnoQj6).
Digital certificates are at the heart of protecting all aspects of data communication, from websites for business and banking, to shopping and product development, to social media for interaction and collaboration.

The growing use of digital certificates
Information explosion, the adoption of cloud computing, and governmental regulation requirements make the use of digital certificates even more important than ever. However, managing digital certificates often proves to be a challenge to many organizations. The practice of setting up certificates only to make an application work, without understanding the roles that these certificates are performing in the process, further complicates the issue. As time goes by, the system can amass numerous certificates without anyone truly understanding which ones are necessary. When the application fails, the administrator hasn’t got a clue about where to begin.

Collection of certificates and keys
The most common use of certificates is to encrypt the communication through the Secure Socket Layer / Transport Level Security (SSL/TLS) handshake protocol that is used by the application. As with a “real” handshake that you perform with another person, there are two parties involved. To perform this “handshake”, each party must set up a collection of certificates and sometimes keys. This collection is called a key ring, a certificate store, a key store, or a key database, depending on the application. Such differences in terminology can potentially create confusion.

Step one
The starting point to figuring out which certificates are involved is to identify the configuration mechanism that is used to specify the name of this collection for the application. For example, the IBM Personal Communications (PCOMM) client uses its Certificate Management utility to set up its collection, which it calls a key database. For the server that communicates with the PCOMM client, the collection is indicated in the Telnet server configuration-policy file by the Keyring keyword in the TTLSKeyringParms statement.

Step two
The next step is to figure out which certificates are needed in the certificate and key collection for both sides. In the SSL/TLS handshake process, the entity that initiates the communication is the client side. The client extends its “hand” to another party with which it intends to communicate, but it (the client) might be uncertain about the identity of the other party. Unlike a physical handshake, the client does not have to identify itself first. The server side must identify itself to the client and the client must validate the server. This is called server authentication.

A simple rule: identify and verify
A common source of confusion is in determining whether the certificate’s private key is required in the collection. The public key is part of the certificate content itself, but its private key is not. Depending on the usage of the certificate, the private key might not need to be present in the same collection as the certificate. Here’s a simple rule for determining this: Identification requires the certificate and the private key of the identity; validation requires the certificate of the issuer only. If the server certificate is signed by a root certificate — that is, the Certificate Authority (CA) certificate — the simplest setup is to put the server’s certificate together with its private key in the server’s collection, and put the root certificate that signed the server certificate in the client’s collection.

In most cases, a server certificate is not signed by a root certificate directly, but rather by an intermediate CA certificate. The root certificate signs the intermediate CA certificate, which is used to sign server certificates. Multiple levels of intermediate CAs could be involved in the chain from the root certificate to the server certificate. The principle of setting up the certificate and key collection is the same. A best practice is for the server side to store the whole chain of certificates in its collection, from the identity certificate to the issuer’s certificate (the root certificate can be excluded), along with the private key of the identity certificate. The client side should store the root certificate of the server chain in its collection.

In addition to the client validating the server, the server might want to validate the client, too. This process is called client authentication, which in fact is mutual authentication. The client must also identify itself to the server so that the server can validate the client. The same rule applies: Identification requires the corresponding private key of the identity certificate; validation requires the issuer certificate of the identity certificate.

One of the incentives of using client authentication is to remove the users’ burden of managing strong passwords.
A client authentication example
Let’s apply the “identify and verify” rule to set up a new feature offered in Network Authentication Service (Kerberos on z/OS) in z/OS V2R2. This feature supports a new authentication mechanism called Public Key Cryptography for Initial Authentication (PKINIT) in the initial steps of obtaining an electronic Kerberos “ticket”. This ticket is required for logging in to a Kerberos-protected system.

The two parties involved in the authentication are the Key Distribution Center (KDC) and the client. The client can authenticate itself with a certificate instead of a user ID and password. This is the client authentication model. Do you remember that client authentication involves server authentication first? The KDC and the client must identify themselves to each other and verify each other. Each party must have its own certificate and key collection.

To simplify the example, suppose the chain for the KDC is certA => certB => certC and the chain for the client is certX => certY => certZ, where certA and certX are the root CA certificates and certC and certZ are the identity certificates. Can you figure out which certificates are required for the KDC’s collection and which certificates are required for the client’s collection? Also, which required certificates have the associated private keys present in the collection? (See the end of the article for the answers!)

Some helpful new features
If you are maintaining an existing certificate and key collection instead of creating a new one, the “identify and verify” rule helps you determine whether the current collection is valid or whether it is a result of adding certificates through trial and error.

One of the incentives of using client authentication is to remove the users’ burden of managing strong passwords.

Unnecessary certificates can make it hard to figure out which one is causing the problem. An excessive number of certificates can also affect performance: the more certificates in a collection, the longer the processing time.

RACF refers to the collection of certificates and keys as a key ring. The RACDCERT command manages the certificates and key ring. In z/OS V2R1, two RACDCERT enhancements, LISTCHAIN and CHECKCERT, were added to help understand the certificate chain and whether the key ring contains the necessary certificates.

RACDCERT LISTCHAIN is a new function. Issuing LISTCHAIN command with the label of a certificate that is stored in the RACF database shows all the certificates in the chain, along with the following information:

- The number of certificates in the displayed chain
- Whether the chain is complete or incomplete
- Whether the chain contains any NOTRUST or expired certificates
- The names of any common rings to which all of the certificates in the chain are connected.
The RACDCERT CHECKCERT function was enhanced to perform a similar function to the RACDCERT LISTCHAIN function. It shows all of the certificates that are stored in the specified data set. It also checks to see if these were installed in the RACF database. The output is similar to the LISTCHAIN command, except that it does not contain the ring information.

**Controlling the names of certificates and key rings**

Over time, you could find a large number of different types of certificates that exist in different key rings in your system. This can be very confusing. In z/OS V2R2, the RACDCERT command is enhanced to help you manage the certificates and key rings in a systematic way. You can now enforce naming conventions for the certificates and the key rings by using the new profiles in the RDATALIB class.

For example, RDATALIB profiles can enforce a rule that the labels of TCPIP certificates start with TCPIP, such as TCPIP_SYS1 or TCPIP_TEST, and that the name for server key rings start with SERVER, such as SERVER_TCPIP or SERVER_DEMO. This can help you get an idea about the intended use of a key ring and a certificate simply from the ring name and the certificate label.

In addition, you can segregate the administration of different types of certificates and key rings by using the certificate labels and key ring names.

For example, system administrators can create certificates with a label that starts with TCPIP and key rings with a name starts with SERVER, and web server administrators can only connect the TCPIP_TEST certificate to the SERVER_Demo key ring.

Another use of these profiles is to prohibit any administrator from using a specific CA certificate to sign any certificate. For example, the CA certificate of z/OS PKI Services can be restricted for use by the PKI daemon only.


**Did you get them right?**

Here are the answers to the questions about certificate setup for the Kerberos KDC and client:

- The certificates and key that are required in the KDC key ring: certA (optional), certB, certC (needs a private key), and certX.
- The certificates and key that are required in the client key ring: certX (optional), certY, certZ (needs a private key), and certA.

---

**IBM z Systems security portal**

As a best practice, IBM strongly recommends that you get access to the IBM z Systems security portal. Subscribe to the automatic notification process to get access to the latest service information about security and system integrity APARs for z/OS and z/VM®. IBM treats vulnerability information in connection with z Systems 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](http://ibm.com/systems/z/advantages/security/integrity_sub.html)
Give credit to Crypto; it gives Crypto to credit!

Protecting credit card data with Visa FPE

BY STEVEN R. HART AND RITA BEISEL

With its support for new Visa Format Preserving Encryption (VFPE) services, Enhanced Cryptographic Support for z/OS V1R13 - V2R1, also known as HCR77B0, really delivers!

Format-preserving encryption (FPE) is a method of encryption in which the resulting ciphertext has the same form as the input cleartext. The form of the text can vary according to use and application. One example is a 16-digit credit card number. After using FPE to encrypt a credit card number, the resulting cipher text is another 16-digit number. In this example, the output ciphertext is limited to numeric digits only. FPE-encrypted credit card information flows easily among point-of-sale (POS) terminals, merchant bank systems, credit card networks, and issuing banks.

Credit card transactions that are made with magnetic stripe cards and integrated circuit (IC) cards send cardholder data from the POS terminal to the payment networks in the clear. This includes the primary account number (PAN) and other data that, if intercepted by a third party, can be used to create fake credit cards and commit online credit card fraud.

IC cards, also known as chip cards or smart cards, reduce credit card fraud by using symmetric keys stored within the chip that are known by the issuing bank only. These symmetric keys are used to provide integrity and confidentiality for transactions between the credit card and the back-end issuing bank. The chips also use a unique transaction counter for each credit card transaction to provide for dynamic authentication. This means a given transaction cannot be captured and replayed at a later time to commit credit card fraud.

Europay, MasterCard, and Visa (EMV) is a global standard for interoperation of IC cards and IC-card-capable POS terminals and automated teller machines (ATMs) for authenticating credit and debit card transactions. While the intent of EMV is to reduce credit card fraud, it is only one layer of defense. Cardholder data, including the PAN, is still sent in the clear during EMV transactions.

The FPE layer of defense

FPE provides a separate layer of defense by encrypting the cardholder data, including the PAN, which is needed for magnetic-stripe and EMV transactions. This layer of defense is intended to help you satisfy requirements from the Payment Card Industry (PCI) Data Security Standard (DSS), specifically Requirement 4, which requires the encryption of cardholder data across open, public networks. This standard is mandated by the card brands and administered by the PCI Security Standards Council (SSC). According to the PCI SSC, “all entities that store, process, or transmit cardholder data must comply with the PCI DSS and may be subject to periodic audits to validate compliance”.

FPE provides a separate layer of defense by encrypting the cardholder data, including the PAN, which is needed for magnetic-stripe and EMV transactions.

The VFPE algorithm is a stream cipher. This differs from other FPE algorithms that use block algorithms. Stream ciphers differ from block ciphers because they use a generated keystream, which is a pseudorandom sequence of values equal in length to the cleartext that is to be encrypted. With VFPE, characters are extracted from each cipher block and modulo addition is then applied to the cleartext and the keystream to create the ciphertext. To decrypt, the same keystream is created and modulo subtraction is applied. No padding is needed and data of any length can be encrypted. As with any stream cipher, it is very important to never use the same keystream more than once. See Figure 1 for an illustration of how VFPE works.
Give credit to Crypto; it gives Crypto to credit!

• Field-level encipher (CSNBFLE and CSNEFLE), which encrypts payment-related database fields, preserves the format of the fields by using the VFPE algorithm. Expanding on our previous example, if you encrypt a 16-digit Extended Binary Coded Decimal Interchange Code (EBCDIC) credit card number, the resulting ciphertext will also be 16 EBCDIC digits. In this context, a database is any structured data area or repository, such as DB2®, Information Management System (IMS™), Virtual Storage Access Method (VSAM), or any column-delimited data set or file.

• Field-level decipher (CSNBFLD and CSNEFLD), which decrypts payment-related database fields that were encrypted using the field level encipher callable service.

The FPE services require some knowledge of the input cleartext character set to create the appropriate output ciphertext. The CSNBFPSEE, CSNBFPFED, and CSNBFPET callable services use the following alphabets to determine valid character sets for the cleartext input parameters:

• **Base-10 alphabet**: used when the character set consists of the numbers 0 through 9 only. The original data type of the source field can be of any type. This alphabet requires the following values to be used in the VFPE algorithm: Number of characters in alphabet(n): 10

• **Base-16 alphabet**: cards are encoded with the special ISO 7811 modified 5-bit American Standard Code for Information Interchange (ASCII) encoding for Track 2. This data type allows parity checking of the digits. Many systems require this encoding to be converted into standard data types for processing. Other data fields can use base-16 encoding and would use this same alphabet when performing VFPE. These data types support values 0 - 9 and A - F. This alphabet requires the following values to be used in the VFPE algorithm: Number of characters in alphabet(n): 16

• **Track 1 alphabet**: this alphabet requires the following values to be used in the VFPE algorithm: Number of characters in alphabet(n): 41

**Think about it!**
The FPE layer of defense can help you protect credit card information inflight and also where it resides in databases. FPE is intended to prevent fraudulent use of credit card information in EMV and non-EMV systems. Consider using this ICSF solution today to protect against credit card fraud.

**Reference**
Fortify your SMF data with digital signatures

BY ANTHONY T. SOFIA, COLIN CHEN, AND HEATHER M. DENNIS

The System Management Facilities (SMF) component of IBM z/OS manages critical organizational data that is generated from the z/OS operating system, middleware, and customer applications. This data can be valuable for long periods of time; it is not unheard of for daily, monthly, or yearly reports to be based on this data. To maintain the accuracy of these reports, the data must not be tampered with (individual records or groups of records must not be edited, deleted, or added).

With the new digital-signature technology added to the SMF component, starting in z/OS V2R2, for users of log stream recording mode, z/OS uses the Integrated Cryptographic Service Facility (ICSF) to sign and validate records. With this feature, the detection of record tampering can now be performed to verify the origin and contents of SMF records.

What is a digital signature?

A **digital signature** is an application of asymmetric encryption in which cleartext data is coupled with an encrypted hash sum of the data. The encryption of the hash sum, or signing, is performed with the private key of the asymmetric key-pair. Verifying the integrity of the data consists of re-hashing the data, decrypting the signed hash with the public key, and comparing these resulting hash sums. If they are not identical, tampering is evident.

This process provides detection of, and deterrence against, tampering. The private key should be treated as a protected resource so that only people and programs with access to the private key can originate validly sign data.

Enable record signing

To begin signing records, you must first authorize SMF to use the ICSF Public Key Cryptography Standard #11 (PKCS#11) services, obtain an appropriate pair of encryption keys, and grant appropriate access to those keys.

Next, you need to define an SMF log stream with the new RECSIGN parameter, with suboptions that identify the key-pair by their token name and your choice of hash sum and signing methods (see Figure 1). This definition will use the private key of the key-pair specified with the token.

You can use the RECSIGN parameter to enable an SMF log stream for record signing with the SMFPRMxx parmlib member or by using the SETSMF command. The RECSIGN parameter can be applied to an individual log stream or globally enabled for all log streams.

You can choose to filter any type of record into this log stream to be signed by using existing SMF log stream parameters. After these parameters are accepted, the system writes additional signature data to the log stream automatically. Signature processing is performed for each unique type and subtype. This means that the data recorded to the log stream can be separated during dump processing, but can still be verified.

```
LSNAME(IFASMF.MULTSYS.STREAM1,TYPE(1:255)),
RECSIGN(HASH(SHA1),
  SIGNATURE(RSA),
  TOKENNAME(token_name_of_asymmetric_key_pair))
```

Figure 1. Syntax of the RECSIGN parameter

Copyright © 2014 IBM Corporation. All rights reserved.
Verify the data
To verify signed SMF data, you must dump the SMF data from the log stream to a data set by using the updated IFASMFDL utility (included in z/OS V2R2). NOSIGSTRIP is a new utility parameter that you can use to cause signature data to be output, coupled with the source SMF records. The signature data is formatted as actual SMF records, but the utility removes the signature data for compatibility, by default.

After SMF records and their signatures are dumped to a data set, you can validate the data set at any time by using the updated IFASMFDP utility (included in z/OS V2R2). You must specify a specific time interval and record types to verify, along with the new SIGVALIDATE parameter (see Figure 2), in which the suboptions mirror those of the RECSIGN parameter.

```
SIGVALIDATE(HASH(SHA1),
            TOKENNAME(token _ name _ of _ asymmetric _ key _ pair))
```

Figure 2. Syntax of the SIGVALIDATE parameter

IFASMFDP’s job output will include a new validation report that describes the timespan and count of records validated, grouped by system ID (SID), type, and subtype. Validation failures terminate processing after the first error is detected and indicate a return code of 8.

SMF is now fortified!
The new digital signatures feature provides built-in support for SMF origin and data verification. This is valuable not only for existing SMF data, but also opens new opportunities for applications that need a trusted log—getting the familiar qualities of service that SMF provides, as well as data verification support by using standard encryption technology courtesy of ICSF.

Additional information
For more information about this new feature, which is available in z/OS V2R2, see z/OS MVS System Management Facilities (SMF), SA22-7630.

The new digital signatures feature provides built-in support for SMF origin and data verification.

John Hancock
IBM Java 8 performance on z13

Hardware and software codesign at its finest

BY ELTON DESOUZA, IRIS BARON, AND CLARK GOODRICH

IBM Java™, which is at the heart of more than 2000 IBM products, has become a critically important part of IBM z Systems. It is the language of choice for many transactional and data-serving workloads on the mainframe. For instance, IBM WebSphere®—along with optimized connections to such traditional system-of-record (SOR) applications as IBM CICS®, IBM IMS, and IBM DB2—enables clients to leverage the benefits of collocation while reducing system complexity. Beyond this, clients use Java in their enterprise modernization efforts to drive SOR transactions in next-generation workloads in the Cloud, Analytics, Mobile, Social, and Security (CAMSS) space. IBM Java 8 and IBM z13™ contain a number of improvements that are ready for immediate use for all of these types of workloads.

IBM Java 8 and z13: performance overview

IBM Java 8 takes advantage of such powerful new z13 features as simultaneous multithreading (SMT), vector registers and instructions—also known as single instruction, multiple data (SIMD)—and other new instructions that accelerate Java virtual machine (JVM) operations.

With SMT, z13 doubles the number of hardware threads per core. Each hardware thread uses its own set of functional units while sharing some resources on the processor, such as the cache hierarchy, which allows for higher instruction-execution throughput. Java workloads can expect to see an up-to-30% throughput improvement without any changes to application code. This new hardware feature is enabled on IBM z Integrated Information Processors (zIIPs) and for Integrated Facility for Linux (IFL) processors on z/VM.

SIMD operations on z13 are performed on 128-bit vector registers, which allow the processor to access and process up to two to 16 times more data than previous-generation mainframes from single instructions. IBM Java 8 exploits the SIMD capability for string and array operations, such as comparison, encoding conversion, and case conversion, along with elliptic curve cryptography (ECC) acceleration. In addition, IBM Java 8 supports automatic vectorization (auto-SIMD), in which loops are recognized and reduced to optimal SIMD sequences transparently.

IBM Java 8 brings noticeable performance improvements to cryptographic operations on IBM System z9® and later. It accelerates cryptography by using the CP (central processor) assist for Cryptographic Functions (CPACF) facility, which is available on IBM System z9 and later with z/OS or Linux on z Systems. Java 8 provides hardware acceleration (using CPACF) of Advanced Encryption Standard (AES), Data Encryption Standard (DES), and Triple DES (3DES)—supported modes are Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), and Electronic Codebook (ECB)—for ciphering and secure hash algorithm 1 (SHA-1) and SHA-2 for hashing. This acceleration is built into the IBM Java Cryptography Extension (IBMJCE) default provider, so no extra configuration is required to benefit from CPACF acceleration.

With z13, IBM Java 8 takes performance to new heights with SMT, SIMD, and improved CPACF. SIMD exploitation in the ECC algorithm showed up to 10 times improvement in microbenchmarks. Java enterprise applications using ECC for secure communication (through the use of AES, for example) can expect to see dramatic out-of-box improvements. The z13 CPACF is approximately twice as fast as the zEC12 CPACF, providing further acceleration.
Cloud computing

Application serving is an important component in modern enterprise clouds. IBM WebSphere Application Server and the WebSphere Application Server Liberty Profile, both Java-based, offer enhanced performance, reliability, and resiliency for building and hosting cloud applications. WebSphere Application Server, as well as other generic Java applications, see immediate improvements from SMT on z13, as shown in Figure 1. This figure depicts the performance of the DayTrader3 benchmark that is running on WebSphere 8.5.5.5 Liberty Profile with Secure Sockets Layer (SSL) enabled, using AES (with the ECC backing algorithm) in different hardware and software configurations.

From the graph, you can see that each hardware configuration has an improvement when running IBM Java 8 over Java 7.1. Comparing IBM Java 8 results on zEC12 and z13, you'll see improvements even in non-SMT configurations because of the general hardware improvements. Java 8 and Java 7.1 see 23% and 28% improvement, respectively, from SMT on z13 compared to the non-SMT mode.

![Figure 1. z/OS WebSphere Application Server 8.5.5.5 Liberty with SSL](image)

Cloud computing

For an example of a hybrid-cloud scenario connecting Bluemix™ to zSOr through a secure channel, see the YouTube video “Good Health zSOr Bluemix and IMS Explorer tooling” ([ibm.biz/BlueMixZOSConnect](http://ibm.biz/BlueMixZOSConnect)). Figure 2 shows a diagram from this video.

![Figure 2. A hybrid-cloud scenario connecting Bluemix to a z SOR through a secure channel](image)

 Analytics

IBM has several analytics products that meet enterprise business needs, ranging from business rules processing to more traditional prescriptive and predictive analytic software suites.

IBM Operational Decision Manager (ODM) is a platform for managing and executing business rules and business events to help make faster decisions, improve responsiveness, minimize risks, and seize opportunities. ODM uses string manipulations extensively, which can benefit from using the SIMD facility with IBM Java 8 on z13. IBM Java 8 on z13 showed 60% improved throughput compared to servers running the Intel Haswell architecture — 53% faster per core!

IBM SPSS® Statistics benefits from the IBM Java 8 auto-vectorization feature, which resulted in a 67% improvement over IBM Java 8 on z13 with SIMD disabled for its core matrix-multiplication code. Any application that performs operations on arrays can benefit from auto-vectorization on IBM Java 8 and z13.

 Mobile

Mobile applications can access data on z/OS by using z/OS Connect and other WebSphere solutions, which are all inherently Java-based. WebSphere Liberty z/OS Connect is a new feature in WebSphere Liberty that allows mobile and cloud applications to connect to traditional z SOR applications such as CICS, IMS, and DB2 through simple representational state transfer (REST) APIs. It can receive data in the JavaScript Object Notation (JSON) format and returns JSON output that can be read by mobile and cloud applications.
A majority of the components of the IBM MobileFirst Platform (formerly known as IBM Worklight®) are also Java-based. The IBM MobileFirst Platform provides developers with the tools to quickly deploy and extend existing SOR infrastructure to the mobile enterprise by using Java.

Figure 3 depicts the results from our performance measurements on a benchmark where a mobile and cloud traffic simulator drives a JSON workload through REST APIs to z/OS Connect. z/OS Connect transforms the data to the binary format that CICS and DB2 can consume. As the JSON payload size increases from 512 bytes to 32 KB, whole system throughput increases by 5 to 16% when using IBM Java 8 versus Java 7.1. Running this configuration on z13 results in an even better throughput improvement of up to 30% due to SMT.

Figure 3. z/OS Connect on zEC12: IBM Java 8 versus Java 7.1

Security
With the rise of mobile and cloud computing, more and more enterprise data is moving through open, unsecure environments. This drives an increasing need for efficient processing of transactions through secure channels. z Systems hardware is well-known for its many cryptographic features and security-oriented design and as such, offers unique capabilities for addressing the concerns of next-generation workloads. You can use the two popular, secure application classes: application serving and point-of-sale (POS) systems.

Figure 1 depicts results for DayTrader3 benchmark that is running on WebSphere 8.5.5.3 Liberty Profile with SSL enabled. On zEC12, IBM Java 8 outperforms Java 7.1 by 36%, which is derived largely from CPACF exploitation. The cumulative gain from IBM Java 8 on z13 versus Java 7 SR4 on zEC12 was more than two times!

The Java POS application models a worldwide supermarket company with an information technology (IT) infrastructure that handles a combination of POS requests, online purchases, and data-mining operations. This application saw a 13% improvement from optimizing SSL by using CPACF alone.

With z13, IBM Java 8 takes performance to new heights with SMT, SIMD, and improved CPACF.

The bottom line...
With workloads such as SSL-enabled application serving seeing up to two times improvement in throughput-per-core, IBM z13 and IBM Java 8 deliver outstanding performance improvements by exploiting such hardware features as cryptographic functions, single instruction, multiple data, and simultaneous multithreading.

Availability
z/OS V2R2

For SMT support
z/OS V2R1 (with the PTFs for APARs OA43366, OA43622, OA44101, and OA44439)

z/VM V6R3 (and later)

For SIMD support
z/OS V2R1 (with the PTFs for APARs OA43803 and PI12412)

Acknowledgements
We thank Marcel Mitran, Distinguished Engineer, CTO z Systems Software Performance and Linux Ecosystem, for his support and guidance.

We would also like to thank YongXin Qin and Peter Bunk from the z Systems Performance Team and Mike Everett, who leads the WebSphere Application Server on z/OS Performance team for providing empirical performance results.

Related information
Java 8 for z/OS announcement letters:

- 64-bit: ibm.biz/BdXGwi
- 31-bit: ibm.biz/BdXGwY

Figure 3. z/OS Connect on zEC12: IBM Java 8 versus Java 7.1
Your company just bought you a brand new IBM z13 loaded with 3 - 5 times more memory than your capacity planning team requested. Sounds unbelievable? Well, believe it or not, new memory pricing on the z13 makes this a common scenario. So, what are you going to do with the memory?

**Number 1: Get mind share**

You've got lots of application programmers busily enhancing your business applications, which includes spending time and effort to shrink memory footprints. But wait! What your application development team doesn’t know yet is that a gigabyte of memory on the z13 can cost less than a single person-day of application developer time!

The most important thing large memory on a z13 can do for your business is free up your application development team to deliver business value faster at lower cost and higher quality using large memory. But first you’ll need to break through years of ‘memory is costly’ training.

Make sure your solution architects know that the make-versus-buy decision just got easier—no more ruling out software with a large-memory footprint. They can deploy analytics and other advanced workloads on z/OS rather than replicating data and platforms. Large memory also enables application programmers to design applications that use traces and diagnostic footprints to simplify test verification and debug problems.

And don’t forget about development systems! Ask your capacity planning team if deploying more memory in the development environment might improve batch turnaround or transactional response time. Even a minor productivity improvement is much more valuable than saving memory especially when your z13 has 3x more memory to deploy.

**Number 2: Tune for improved availability**

Start by ensuring that your production systems have enough memory headroom to avoid outages. Now that you have 3 - 5 times more memory, you can give your performance and capacity planning teams the freedom to configure applications to handle more out-of-memory scenarios, hung devices, and similar memory intensive situations:

- Consider using tuning middleware, like IBM MQ for Multiplatforms for z/OS, V8.0, where 64-bit buffer areas can smooth workload spikes.
- Deploy enough frames to improve SVC dump performance.
• Look for cases where more memory might help. For example, the Change Data Capture (CDC) task, used by IBM DB2 Analytics Accelerator for z/OS for continuous data replication, uses a staging area in memory to hold uncommitted changes. If you have SQL statements updating or deleting millions of rows, the staging area can be a few gigabytes in size. It’s way better to configure a few extra gigabytes for CDC than to allow an application change to break replication of a table in production!

• Finally, examine your test environments. If the memory configurations of your test systems accurately mirrors the production system, you can do better stress testing.

**Number 3: Tune for response time and CPU consumption**

There are many middleware packages that run on z/OS, and each needs to be tuned to exploit memory.

**Example 1: DB2**

DB2 tuning for large memory can have astounding results for some applications. IBM has published results for IBM SAP on z Systems that show reductions in response time of up to 70%, increases in transaction rates of up to 37%, and savings in CPU time per transaction of up to 25%. See: ibm.biz/BdXGTy

Before you increase the size of DB2 buffers, use large memory to tune DB2 to operate well with its current buffer pool. There are several well-documented DB2 tuning changes you can make now that you have 3x the memory available:

• **PGFIX(YES)** your DB2 Bufferpools. If you have pageable DB2 buffer pools, you could see a 1 - 3% CPU improvement from this one change. With 3x the real memory, you now have enough available frames in your system so that there’s no reason to make your buffer pools pageable.

• **Use large pages.** Using 1 MB and 2 GB large pages gives you another potential 1 - 2% CPU gain. Set the amount of the real storage for large pages on the LFAREA keyword in the IEASYSxx parmlib member. By default, DB2 uses 1 MB large pages, if available, and the buffer pool is fixed. If your buffer pools are large enough, try 2 GB pages, and further reduce translation lookaside buffers (TLB).

• **Implement thread reuse.** Use IBM IMS or IBM CICS applications to gain CPU savings by avoiding the cost of thread allocation and deallocation. Thread reuse and the RELEASE(DEALLOCATE) bind option can boost CPU performance by avoiding package allocation.

• **Tune the global dynamic statement cache.** For CPU savings, avoid specifying a FULL PREPARE statement. The EDMSTMTC system parameter defaults to 110 MB but you can tune it to 4 GB with IBM DB2 11 for z/OS.

• **Avoid SHORT PREPARE CPU time.** Tune the local statement cache. The MAXKEEPED system parameter defaults to 5000 prepared, dynamic SQL statements to be saved past a commit point and can be tuned up to 200,000 statements with DB2 11.

• **Reduce CPU/elapsed time.** Change the in-memory data cache MXDTCACH from its 20 MB default to achieve a better access path selection in DB2 11.

Now you can use large memory to increase the size of DB2 buffer pools. Have your DB2 performance team find and remove synchronous reads with larger local buffer pools, each read avoided saves about 35 microseconds of CPU and improves response time. With the DB2 V11 buffer pool simulation support, you can try different buffer pool sizes in production so that you can see results before you make any changes. Buffer pool simulation is described in Managing DB2 Performance: ibm.biz/BdXGTu

If you are doing DB2 data sharing, you must configure large enough coupling facility (CF) structures to support your new buffer pool sizes. Use the CF Structure Sizer (CFSizer) tool to find the right size. Also, consider increasing the size of global buffer pools in the CF. The preceding reference includes results from changing both local and global buffer pools.
Example 2: Java
Large-memory tuning is a little different for Java, because unlike the DB2 buffer-pool scenario, bigger is not better and smaller heaps perform better. A customer right-sizes heaps based on the upper-bound active footprint of their application. The CPU gains from large memory come from using 1 MB and 2 GB large pages with Java. Benchmark results have shown up to 8% improved CPU performance simply by converting 4 KB pages to 1 MB pages.

Java begins consuming substantial amounts of memory as you deploy many Java virtual machines (JVMs) processing concurrently, a scenario common with both cloud Software as a Service (SaaS) patterns and Java based restructures of traditional IMS and CICS transaction processing systems. Several of the largest deployed z/OS images primarily use memory for JVMs.

Number 4: Deploy memory-hungry workloads
Example 1: Cloud applications
There are lots of advantages to the cloud programming model. Typically, the value of integrating cloud services calls into existing applications outweighs the additional network traffic and additional memory requirements incurred. With 3x more memory at your disposal, the memory consumed by transactions using cloud services waiting for responses from remote systems is not an issue.

Example 2: Analytics
More and more organizations rely on real-time insight to make important decisions and to drive customer interactions. The IBM Cognos® family is a popular suite of analytics products that runs on z/OS and Linux on z Systems, delivering substantial value. The memory required for Cognos can increase with heap sizes of tens of gigabytes. IBM Operational Decision Manager for z/OS provides a different analytics workload-capturing, automating and governing frequently occurring, repeatable business decisions. IBM Operational Decision Manager uses in-memory caching of events to provide real-time rules processing. With 3x the memory, you have the resources to deploy Cognos and IBM Operational Decision Manager.

Example 3: Linux on z Systems
Linux workloads are one of the fastest growing z/OS workloads. A substantial number of enterprises using z Systems have migrated distributed workloads from x86 servers to Linux on z Systems to gain the advantages of z Systems hardware and hypervisor technology. If you are not running Linux on z Systems, try deploying a prototype. Typical Linux implementations use substantially more memory than z/OS. With 3x more memory, you are ready to evaluate Linux without stressing your z/OS system.

Number 5: Collect the data to show the value you’re getting
Sooner or later someone will ask: “You deployed 3x more memory. Was it worthwhile?”

It is! To document the value, look for response-time gains. Many businesses have service level agreements for response time measured by external monitors. Collect before-and-after data as you tune DB2 with larger buffers. See if applications or your large objects can help you tie response-time gains to business value either anecdotally or with measured data.

Because the availability of z/OS systems is already high, it can be hard to get data on improvements in availability in a single enterprise. Check with your capacity-planning team to see if they have a point of view on the improvements in risk level. Also, look at the development and test environments for gains.

Document CPU savings as you tune your system with large memory. This can be difficult because typical application growth of 1% per month can hide the gains you made in in DB2, Java and other middleware. For DB2, if you track sync reads removed, you can show additional CPU saved by using the 35 microseconds of CPU saved per sync read rule of thumb.

Take credit for the flexibility you’ve gained by deploying new z/OS partitions and new workloads on your existing z/OS images. Talk up new applications and prototypes you couldn’t deploy with your previous memory footprint.

In the long term, the z13 memory increase can:

- Deliver business value with new capabilities delivered faster at higher availability
- Improve response time and save CPU related hardware and software costs
- Increase application productivity with simplified application design and better development and test environments.

Your enterprise will feel the value!

The most important thing large memory on a z13 can do for your business is free up your application development team to deliver business value faster at lower cost and higher quality using large memory.
Great news! Starting with z/OS V2R2, z/OS Management Facility (z/OSMF) is now included as a base element of z/OS so there's no need to order z/OSMF separately. We also improved the look and feel of z/OSMF, and made other z/OSMF enhancements.

Easier to configure
Configuration workflow provides automation scripts, system properties discovery and import functions, as well as conditional support. These functions work together to guide you in your implementation of plug-ins, such as the incident log, IBM Workload Manager (WLM), web-ISPF, capacity provisioning, and so on. In addition, z/OSMF eliminates the need to run configuration scripts to apply PTFs. Service maintenance is now integrated into the z/OSMF server process to run automatically as a separate step before the z/OSMF server process begins.

Multi-sysplex support
Systems can now be organized and viewed by the CPC in addition to sysplex and user-defined groups. A topology view is added that renders a graphical viewing of a sysplex. A set of RESTful APIs is provided that enable communication and data transfer between different systems across the sysplex.

In addition, you can log on to one z/OSMF instance and conveniently access other z/OSMF instances without having to log on again. These features are intended to allow you to manage all systems from a single point and drive actions consistently across appropriate groups. As the first exploiter, you can use the Incident Log task to obtain an aggregated display of incidents across sysplexes within your enterprise, as shown in Figure 1.

More powerful workflow engine
Workflow authors can create structured and automatic workflows like middleware provisioning, daily operations, system migration, and so on. The following functions automate complex tasks and can help you set up test environments quickly:

- Workflow-to-workflow execution allows a workflow to call another workflow.
- Workflows can be placed in a canceled state, whereby they are no longer active, but the data within the workflow continues to be accessible.
- REST APIs can manage a workflow programmatically, which allows exploiters to create, perform, query, monitor, and terminate workflows.
- Workflow authors can provide a default job card for workflow level or step level, while users can customize the job card for a different level. You can also specify variable substitution in the job card.
- Ability to update workflow instances by creating, editing, and removing steps.
- Migration of a prior version of a workflow so that the status of completed tasks is preserved when it is customized as a new workflow.

Software management improvements
REST interface services APIs are provided that enable applications to list all of the software instances defined in z/OSMF, read the properties for a single software instance, and define a new instance of installed software. This allows a client application to interact with the software management task in z/OSMF. In addition, you have the ability to print and export data in key tables, including tables that list software instances, deployments, and products.

z/OS continues to improve simplification and usability. In z/OS V2R2, z/OSMF simplifies your system administration and provides more external application support. For more information about z/OSMF, see:

ibm.com/systems/z/os/zos/features/zosmf.

Acknowledgments
We would like to thank Kirsten Brunner, Barbara Sannerud, Xiao Zhen Zhu, Bo GL Li, and Dan Qing Huang for their contributions to this article.
All fun and games

The solution will be on the z/OS Hot Topics website and on the z/OS Hot Topics Facebook page.
The number of web service applications on the internet has increased significantly in recent years. RESTful applications that use HTTP or HTTPS as a means of communication and send JSON or XML data is as common as it gets in the mobile, client/server world.

Wouldn't it be cool if your existing z/OS applications running in a traditional environment could easily ramp up to play in the game as well as through a set of base z/OS services available to most programs on z/OS?

If you’re excited about these options, welcome to the new z/OS Client Web Enablement Toolkit! Built into the base of the z/OS operating system, the toolkit provides a lightweight solution to enable these applications to more easily participate in this client/server space by providing the following built-in features:

• A z/OS JSON parser that can be used to parse JSON text that comes from any source and create new JSON text or add to existing JSON text.
• A z/OS HTTP/HTTPS protocol enabler which uses interfaces similar to other industry-standard APIs.

Just about all environments on z/OS can avail themselves of these new services. Traditional z/OS programs that run in native z/OS have little or no options that they can easily use to participate in web services applications. Programs running as a batch job, as a started procedure or in almost any address space on a z/OS system now have APIs that can be used in a similar manner to any standard z/OS APIs provided by the operating system.

Furthermore, programs can use these APIs in the programming language of their choice. You can use C/C++, COBOL, PL/I, and Assembler languages, and samples are provided for C/C++, COBOL, and PL/I.

Would you like to hear more about the parts of the toolkit and get a small taste of what you can do?

z/OS JSON parser

Suppose that you would like to be able to make sense of a large JSON text file that was sent to you from a web server that you are communicating with. The new z/OS JSON parser can do the heavy lifting for you.

The following questions can help you decide which style of parsing is best for you:

Welcome to the new z/OS Client Web Enablement Toolkit!

• Do you know the format of the data that is being returned?
• Are you looking for specific fields in a particular format?
• Do you need to learn about all the data that is returned?

Based on your answer, you can choose the “search” style, the “traversal” style, or a combination of both. The “search” style looks for specific key values in the text stream and then finds the values that are associated with those key values. The “traversal” style starts the traversal parser services to recursively move through the text stream until it learns what was sent.

In either case, a program that uses the JSON parsing services follows this format:

1. Start the JSON parser initialize service (HWTJINIT) to create a parsing instance.
   **Tip:** You can go wild here and create as many parser instances as your application requires. Each parser instance allows the z/OS JSON parser the ability to separately manage the parsing of a JSON text stream. The more instances you have, the more concurrent JSON text streams you can parse.

2. Call the JSON Parse service (HWTJPARS) to have the parser validate the syntax of the text stream and create an internal representation of the JSON text data. Once the data is parsed, it allows all subsequent services to run faster because the data is usually instantly available.
3. Get the data that is needed:
   a. If you use the “search” style, the JSON Search service (HWTJSRCH) is likely to be called here to find a specific key in the data (a name in a JSON name/value pair). If that key is found, the value can be retrieved by using one of the parsers “get” services. You can also scope search to a particular JSON object if desired.
   b. If you use the “traversal” style, you can simply reuse the sample code that is in one of the many programming languages that is provided by the toolkit in samplib. The sample code shows a simple way of parsing through the JSON text stream. The data that is returned causes the code to iterate through entries in an object or array to call the appropriate JSON parsing services until all the data is learned.

4. When the process is finished, the parser instance (HWTJTERM) is used to free the storage that is being used by the parser.

   It’s that simple! Oh, by the way, do you want to create JSON text or add entries to the existing text stream you received? If so, you’re covered here! You can easily create JSON text and rebuild the entire JSON text stream with the new entries with the JSON Create service (HWTJCREN) and the JSON Serialize service (HWTJSERI).

z/OS HTTP enabler

Wouldn’t it be nice to see an existing z/OS application webify itself by using an industry-standard method for communicating to a web server, and sending and receiving REST API calls and replies? The HTTP/HTTPS enabler portion of the toolkit allows:

- An application to make simple API calls to connect to the server
- An application to build an HTTP request, send it to the server, and receive a response.

SSL (including TLS), cookies, proxies, and redirects are supported in the toolkit.

How would the application use the toolkit to execute a REST API interaction? Here is an application blueprint:

1. Initialize a connection (HWTHINIT) to get the work area ready for the toolkit to process the connection.
2. Set the necessary connection options (HWTHSET). In this step, the URI of the web server, SSL options, cookie processing behavior, and other options can be set, one option at a time.
3. Connect to the server (HWTHCONN).
4. Initialize a request (also HWTHINIT) to get another work area necessary for the toolkit to process the request.
5. Set the necessary HTTP request options (HWTHSET). What HTTP request headers do you need to send to the server? What about a request body? What application code should receive control to process the data that is returned from the web server? Again, it’s setting one option at a time.
6. Send the request (HWTHREQST) to join the request to a particular connection and send it over that connection. The responses are sent to user callback routines (exits).
7. Do house cleaning. You can stop a request to free its work area storage, disconnect from a web server, and end a connection. Issue whatever services make sense in your application.

Don’t feel overwhelmed. A sample is worth a thousand words! Samples are provided in many z/OS programming languages to show just how simple this stuff is.

How can you get your hands on this toolkit? Both pieces of this amazing toolkit are available in the base of z/OS V2R2. The z/OS JSON parser portion of the toolkit was available in April 2015 in V2R1 and the HTTP enabler is now also available in V2R1.

You can find more information about the toolkit in z/OS MVS Programming: Callable Services for High-Level Languages, SA23-1377. Also, see z/OS Introduction and Release Guide, GA32-0887 to learn how easy it is to webify your application!
This year marks the tenth anniversary of the Master the Mainframe student contest in the U.S. and Canada. When the contest made its debut in 2005, it drew in over 700 college and university participants. Since that time, the contest has seen significant growth with an average of over 5,000 participants annually. In 2007, the contest was expanded to include high school participation. Throughout its ten-year history, five high school students have claimed some of the top five prize spots. This year was no exception, with two of the top five winners being high school students.

In 2014, IBM held its first Master the Mainframe World Championship, and invited 43 students from 23 countries to compete in a global challenge.

Each year the contest offers new challenges which expose students to new enterprise system technologies, highlighting the modern capabilities of Big Data, Cloud, Security, and Mobile workloads on the mainframe. This year’s challenges gave students the opportunity to work with a variety of operating systems (z/OS, z/VM, z/TPF, and Linux) and programming languages such as Assembler, C, COBOL, and Java. The final project included a real-world challenge in which competitors wrote a credit card application that sorts and manipulates data for reporting and analysis. The top five winners were flown to Poughkeepsie, New York in March and were honored at an awards ceremony. The awards were presented by Ross Mauri, IBM General Manager, z Systems and Maria Boonie, VP, z Systems Software and Firmware Development.
Throughout the years, the one constant that has remained the same is: no experience is necessary to participate. In fact, the contest is designed for students with little or no mainframe experience. Students just need to bring drive and competitive spirit, and they will be ready to compete. The contest increases with difficulty as it progresses.

**Let’s hear from our winners**

**First Place: Kevin Matesi, Northern Illinois University**

Kevin is a second-year graduate student at Northern Illinois University, and plans to graduate in May.

“I participated in the 2014 Master the Mainframe Contest, in which I finished with an honorable mention for Part 3. This year I was shocked to see I had finished in 1st Place. It was an invaluable experience, and I was honored to be a part of it. It was amazing to meet the other extremely talented winners. I’m excited for the opportunities that this contest has made available to me.”

**Second Place: Joseph Bloom, Deerfield (Illinois) High School**

Joseph is a senior at Deerfield High School in Deerfield, Illinois.

“The best part of the contest for me was discovering this kind of hidden world that is the mainframe. I had never heard of a mainframe before the contest. But even going through just Part 1, I was really intrigued by this totally foreign environment. I’ve learned so much about how mainframes are used, and how I use one every day without even realizing it. My contest experience has been really amazing.”

**Third Place: Jeremy Krach, University of Maryland**

Jeremy is a sophomore in computer science at the University of Maryland, College Park.

“I really enjoyed how the contest walks its competitors through the challenges. At each stage, you get enough advice to figure things out, although it is never enough to make the answer totally obvious. That design is partially what kept me hooked throughout. I was learning new skills from the challenges, and none of them left me on my own. I always had enough resources to piece together the solution.”

**Fourth Place: Hongzhe (Henry) Liu, Algonquin (Massachusetts) Regional High School**

Henry will graduate in 2016 and plans on studying artificial intelligence in college.

“This was my first year in the Master the Mainframe Contest. Once I got going through the challenges, I just couldn’t stop. The challenges are set up in a way that introduces learning, but with fun instructions and great prizes. I was honored to be placed within the top 5 students from the contest. I wasn’t quite sure, being a high school student, if I would have enough skills to be able to get all the way through.”

**Fifth Place: Steven Hoover, Syracuse University**

Steven is a graduate student studying information management at Syracuse University’s School of Information Studies.

“I had heard of the contest before, but had never participated. If you enjoy programming challenges, want to gain mainframe-related skills and knowledge, and have a bit of free time on your hands, then this contest is for you! The prizes are nice incentives, but your experience in the contest can lead to even more exciting rewards and opportunities. Employers need people who are knowledgeable and excited about mainframes, and participation in this contest is a great talking point when looking for a job!”
Track down GRS requests with SMF

**BY JOSEPH GENTILE AND CHRIS BROOKER**

People monitor global resource serialization requests to diagnose problems such as system/application hangs or when planning for GRS RNL migrations. This proactive monitoring yields insights that prevent problems during a migration. Currently, everyone tends to use global resource serialization ENQ/DEQ Monitor (ISGAUDIT) to do this monitoring. But although ISGAUDIT provides great function, it also has some drawbacks:

- It causes an increased ENQ/DEQ path length, even when no request is being monitored.
- Has a complicated filtering scheme.
- Does not really support GQSCAN activity or ENQs at the step level. And, I bet you know how much that stings!

**Enter global resource serialization**

Starting with APAR OA42221 for z/OS V2R1, GRS began cutting an SMF 87 record for each super-expensive global generic GQSCAN. Now with z/OS V2R2, we have started recording ENQ/DEQ activity in SMF records as well. This new monitoring technique contains significantly less overhead on the ENQ/DEQ thread! (You can find the same function in the PTF for APAR OA42221 for z/OS V2R1.)

---

**Frustration-free filtering**

In tandem with SMF record 87, you can use new parmlib member GRSMONxx for quick and easy changes to your filtering without losing any ISGAUDIT filtering function. The powerful, highly customizable GRSMONxx parmlib member lets you define inclusion and exclusion criteria, similar to what you would do with the GRSRNLLxx parmlib member. GRSMONxx contains a series of FILTER statements, each of which can be set to INCLUDE (add to the inclusion list) or EXCLUDE (add to the exclusion list). A FILTER statement contains subkeywords that let you define the filter criteria (see Figure 1).

```c
/* ---------------------------------------------------------------- */
FILTER INCLUDE QSCAN GENERIC(YES) XSYS(YES) /* IN THIS FILTER, WE 
FOLLOW OA42221 LOGIC: TO TRACE 
QUEUE SCANS THAT ARE BOTH GENERIC 
AND CROSS-SYSTEM. */
/* ---------------------------------------------------------------- */
FILTER INCLUDE ENQDEQ QNAME(*) RNAME(*) RNLMATCH(YES) /* IF PLANNING 
TO CHANGE RNLS, IT MAY BE USEFUL 
TO MONITOR FOR ENQ/DEQ REQUESTS 
WHICH MATCHED THE CURRENT RNLS */
FILTER EXCLUDE ENQDEQ QNAME(SYSDSN) RNAME(*) /* IT MAY BE HELPFUL TO 
FILTER OUT KNOWN RNL MATCHES, 
SUCH AS SYSDSN ENQS, TO REDUCE 
THE DATA */
/* ---------------------------------------------------------------- */
FILTER INCLUDE ENQ QNAME(SYSZIBM) RNAME(CHRIS) /* TRACE ENQ/RESERVE/
ISGENQ REQUEST=OBTAIN FOR 
QNAME=SYSZIBM, RNAME=CHRIS */
FILTER INCLUDE DEQ QNAME(SYSZIBM) RNAME(JOE) /* TRACE DEQ/ISGENQ 
REQUEST=RELEASE FOR 
QNAME=SYSZIBM 
RNAME=JOE */
/* ---------------------------------------------------------------- */
FILTER INCLUDE ENQ QNAME(*) RNAME(*) RESERVE(YES) /* Trace ENQs which 
resulted in a hardware 
RESERVE after RNL and 
Exit processing. */
/* ---------------------------------------------------------------- */
FILTER INCLUDE ENQ QNAME(SYSDSN) RNAME(THE.DATA.SET.NAME) 
JOBNAME(MYJOB) /* Trace ENQs for a particular data 
set coming from MYJOB */
/* ---------------------------------------------------------------- */
```

*Figure 1. Sample GRSMONxx parmlib member filter statements*
We added the RNLMATCH(YES|NO|ANY) keyword to help you plan RNL changes, and the WAITER(YES|NO|ANY) keyword to help diagnose contention. Keywords are logically ‘AND’ed together in the FILTER statement. Keywords JOBNAME, QNAME, and RNAME accept wildcard characters.

Think big!
These GRSMONxx filters are flexible and powerful, but if you’re not careful with your FILTER statements, you can end up with a large number of requests. Unlike other SMF records, the system doesn’t just cut SMF type 87 on SMF intervals. SMF type 87 records are also cut whenever there are 30 matching requests. Ensure that you have enough SMF resource available for these records. We also suggest you route type 87 records to their own log stream so they don’t steal SMF space from other record types.

Watch your STEPs
With all of the focus on wearable devices these days, you often hear people talking about their steps. We’re no different, but the steps we’re talking about are not quite the same. While ISGAUDIT can’t trace ENQ/DEQ requests with a scope of STEP, the SMF type 87 support is fully capable of keeping track of your STEP level requests. Let’s see who can get to 10,000 first!

A couple of use scenarios...
We can think of many uses for this highly flexible function:

- Let’s say you’re moving to go to a GDPS® environment, where hardware reserves are a big no-no. Use SMF type 87 records to trace all ENQs resulting in hardware reserves.
- Plan for RNL migrations using SMF type 87 records.
- Use type 87 records to find tasks holding important ENQs for too long, causing traffic jams leading to system or sysplex slow down. Sure, you could issue the D GRS,C command during the slowdown to try catching the offender, but by the time the command runs, the ENQ might be released and the trail cold.

SMF 87 recording is a powerful tool in your systems programming arsenal. These records bring to light system activity that was previously almost invisible. And they do all this without long-time contention or having to use ISGAUDIT with its accompanying increase in ENQ/DEQ overhead!
Looking to ease the task of job-step region management? Need a way to request above- and below-the-line region amounts in job control language (JCL)? Is maintaining your IEFUSI region-management exit becoming arduous? New with IBM z/OS V2R2, the REGIONX JCL keyword and the SMFLIMxx parmlib member can help!

When specified on the JOB or EXEC JCL statements, the single-parameter REGION value is applied to above- and below-the-line storage, with below-the-line storage applied first. As such, users can wind up with more below-the-line storage than they want, and less above-the-line storage than they need. The new REGIONX JCL keyword resolves this limitation because it allows each below- and above-the-line value to be specified separately. REGIONX syntax is as follows:

```
REGIONX=(below,above)
```

where:

- `below` is less than 16384K or 16M, specified in K or M units
- `above` is less than 2G, specified in K, M, or G units
- 0K, 0M, or 0G means “maximum.”

You can specify REGION and REGIONX in the same job, but you cannot specify them on the same statement. Also, they are mutually exclusive when one is specified on the JOB statement and the other is specified on the EXEC statement. In contrast with the way REGION works, REGIONX values that are specified on the EXEC statement override REGIONX values that are specified on the JOB statement. This allows for job-level default region settings and any step can receive a different requested amount. Default values are applied for either of the REGIONX values that are not specified. In the case of REGIONX on an EXEC statement, either or both values can be defaulted to the REGIONX values on the JOB statement, if specified.

REGIONX provides increased granularity because you can request above- and below-the-line storage for a single job or job step. But what about improvements in installation-wide region management?

Clients use the IEFUSI installation exit to set REGION and MEMLIMIT values for jobs to ensure that proper amounts of working storage are allocated for user programs and for system usage within an address space. This exit is also used to ensure that programs do not exhaust system resources and to enforce job cancellation rules. With a long list of input and output parameters, it can become time-consuming and error-prone to update the exit to meet growing region-management requirements.

The new SMFLIMxx parmlib member allows installations to implement customized region and MEMLIMIT appropriations, as well as job cancellation criteria based on the job and its environment. All of this is done without the additional work of changing the exit code. Here’s how it works: SMFLIMxx allows policies to be set using “rules”. A rule consists of filters and actions or values to be set when the filters match. The full set of filters follows:

- `JOBNAME(1 - 8 values)`
- `JOBACCT(1 - 8 values)`
- `STEPNAME(1 - 8 values)`
- `STEPACCT(1 - 8 values)`
- `PGMNAME(1 - 8 values)`
- `USER(1 - 8 values)`
- `JOBCLASS(1 - 8 values)`
- `SUBSYS(1 - 8 values)`
- `SYSNAME(1 - 8 values).`
SMFLIMxx allows for some special filtering of accounting data. For example:

```plaintext
REGION JOBACCT((D2404P,POK,*),
               (,FSH,*),
               (*,STL,HERMAN))
REGIONBELOW(4M)
REGIONABOVE(1G)
MEMLIMIT(4T)
```

The first job accounting set would match a job with JOB (D2404P,POK,JONES). The second set would match JOB (,FSH,SMITH). Note that in this case, the first field must be null. The third set would match JOB (B7VB,STL,HERMAN). When extra parentheses are not used, a match would be on the first account field only. Use the percent sign (%) to indicate “no matching on remaining fields,” for example:

```plaintext
REGION JOBACCT((D2404*,%)) ...
```

This would match JOB (D2404P,STL,’Extra data added by user’) as well as JOB (D2404,FSH).

Multiple rules are applied in succession, so that the most recent rule for a set of criteria “wins.” A use case for this would be a rule that specifies EXECUTE(CANCEL) for a general case and a second rule that specifies EXECUTE(YES) for some specific job. SMFLIMxx does not replace IEFUSI. Its policies are applied after IEFUSI runs. Implications of this are:

- EFUSI can be left as-is and SMFLIMxx will override its decisions as conditions warrant.
- Optionally, you can update IEFUSI to override SMFLIMxx in all or some cases. A new parameter bit (sub-word 1, bit 4, pointed to from word 5 of the parameter list) will bypass SMFLIMxx.
- You can update IEFUSI to remove all REGION and MEMLIMIT decisions.
- Leave other processing in IEFUSI, such as processing required for other exits (values passed to other IEFUTL or IEFUSO, for example) in IEFUSI.

You can update IEFUSI at your convenience.

**SMFLIMxx and REGIONX spell relief!**

SMFLIMxx provides a powerful way to specify policy-based region and job-step accounting management using “English-like” decision points. You can use this parmlib member in concert with IEFUSI to simplify the enforcing of operational requirements. REGIONX enhances the way a job step can request storage. These two z/OS V2R2 enhancements simplify system management and ensure optimal use of system region resources.
anyone remember the Everly Brothers song, “Problems, problems, problems all day long”? Well, IBM z/OS problem determination is an area of interest for system programmers, both new and experienced. A message or abend code can be a clue to find out where to begin in solving a problem. What steps do you take? And where do you go next?

Where to start
When you encounter a message or an abend, what do you do? The first step is to identify the component that issued it. For z/OS V1R13, the LookAt tool is a good place to start. LookAt brings you to the documentation for the message, including the component issuing the message or abend, what it means, and sometimes, how to get around the underlying problem. The z/OS V1R13 LookAt tool is located at:
ibm.com/systems/z/os/zos/bkserv/lookat

• For z/OS V2R1 and later, search IBM Knowledge Center to help identify a component associated with a message or abend. You’ll find IBM Knowledge Center for z/OS at:
ibm.com/support/knowledgecenter/SSLTBW/welcome

Some messages are accompanied by additional diagnostic information, such as the errnojr reason code common in z/OS UNIX System Services messages. In this case, using BPXMTEXT from IPCS, TSO/E or the z/OS UNIX shell, provides even more specific information about the failure.

Where to look for resources
• The software support handbook is another good resource to search for known issues or to request fixes. If you don’t find enough information to address the underlying problem, you might need to gather more documentation and report the problem to IBM for analysis. You’ll find the software support handbook at: ibm.biz/BdXGwG

• z/OS Problem Management, SC23-6844, describes how to perform operational problem determination (PD) for component-specific problems including loops, hangs, and critical messages. Each component section documents problem symptoms and describes basic commands and functions that can be used to perform PD, including how to investigate the problem, what doc to collect, and recovery actions. Some of the chapters for component-specific problems include RRS, XCF/XES, System Logger, Catalog, JES2, PDSE, SDM, VSAM, and VSAM RLS. And we’re hoping to add more!
Hints and tips for finding your product in the IBM Support portal

- Access the IBM Support Portal:  
  [ibm.com/support/entry/portal/support](http://ibm.com/support/entry/portal/support)

- Start typing! You won’t see the list of products until you start typing search terms.

- Sometimes less is more. Enter a single letter to see all the products that start with that letter. Or, if you're typing in a long phrase, like *WebSphere Application Server*, start by typing *WebSphere*.

- Sometimes more is more. Try avoiding abbreviations. For example, *WAS* will not yield results (it is not an approved abbreviation), where *WebSphere* will.

- If you know the nine character component ID, enter it directly into the search field.

- If you can’t resolve or find a fix for a problem, you can initiate a service request to IBM in one of the following ways:

  - Link to the IBM Support Portal from the support handbook
  - Access the IBM Support Portal directly:  
    [ibm.com/support/entry/portal/support](http://ibm.com/support/entry/portal/support)

When you initiate a service request, you must select the product or component where the problem is to route the new service request to the correct support group within IBM. It can be hard to find z/OS products, because they might be listed at the bottom of alphabetically-sorted lists. We suggest that when you report a problem against a new component, you then add it to your preferred products list to make it easier to find. PMRs are routed to component queues based on the associated 9-character component ID. For information about associating a component with a component ID, see *z/OS MVS Diagnosis: Reference*, GA32-0904: [ibm.biz/BdXGwu](http://ibm.biz/BdXGwu).
Logrec has a long and venerable history as the sometimes overlooked system repository of error data.

**A little history...**
Back in 1994 in IBM MVS™/ESA V5R1, the LOGSTREAM=IEASYxx parameter allowed any data set to be the logrec data set when specified at IPL. Then in 1995, with MVS/ESA V5R2, log stream support was added to logrec. This allowed system error records to be written to a log stream rather than the traditional logrec data set, as well as filling the need for a sysplex source of error records. But there still wasn't as much flexibility as we wanted. For example, when users moved from data sets to log streams, they found that the logrec data set was never freed. Although it's possible to choose a different data set name at IPL time, the logrec data set is stuck in use for the life of the system. Naming for a log stream was even more restrictive. The only name that was allowed for the log stream was SYSPLEX.LOGREC.ALLRECS. Further, clients who IPLed the system with log stream recording turned on could only switch to data set recording after re-IPLing the system.

**And a dynamic future...**
Now, in z/OS V2R2, logrec is dynamic! We added support to the SETLOGRC command to:

- Dynamically allocate and deallocate logrec data sets on the fly.
- Specify a new logrec data set or log stream name. Yes, you can use the SETLOGRC command (see Figure 1) to specify a log stream name other than SYSPLEX.LOGREC.ALLRECS! Users can specify log stream logrec recording at IPL time by using the new LOGREC=LOGSTREAM=my.log.stream parameter of IEASYSxx.

These dynamic options open up a whole range of great possibilities. Previously, logrec users had to IPL with data set recording and then switch to log stream recording after the system started. This meant that if a system logger error occurred, the system might be unable to record logrec errors and data might be lost. Now, installations can IPL with their intended logrec recording method and switch between data sets and log streams at any time.

**How bad is it? Better logrec messages**
We've spiffed up the way logrec handles temporary log stream conditions to clarify when immediate action is needed. Previously, any logrec log stream error resulted in system message IFB100E—the error could be catastrophic (like the system logger address space failed), or it could be a temporary glitch (like a CF structure rebuild in progress).

Now, in z/OS V2R2, the system issues different messages for different types of problems:

- Message IFB102E (see Figure 2) indicates that the CF structure or staging data set is full. In this case, you might want to do some log stream tuning to prevent future full conditions, but no immediate action is required. As always, logrec still writes to the log stream when the resources are available after a log stream offload.

**References**
- For more information on the SETLOGRC command, see z/OS MVS System Commands, SA38-0666.
- For more information on the LOGREC IEASYSxx parameter, see z/OS MVS Initialization and Tuning Reference, SA23-1380.
IBM z/Advanced Workload Analysis Reporter (IBM zAware) is a machine-learning solution first introduced with the IBM zEnterprise® EC12 (zEC12). It uses cutting-edge pattern recognition techniques to analyze message logs and pinpoint deviations from normal operational patterns. In the first release, IBM zAware allowed for the real-time analysis of z/OS OPERLOG. Available on the IBM z13 (z13), Version 2 introduces the support to analyze the syslog that is generated by Linux running on z Systems. This support results in a new set of advanced analytics and user interface enhancements.

Linux—analyze this

The introduction of new cutting-edge analysis provides a solution for Linux syslog messages. Adding new technologies usually presents new challenges. One of those challenges occurred when Linux support was added to IBM zAware. There is a difference between Linux and z/OS message traffic. While the z/OS message stream consistently yields a large quantity of messages, the Linux message stream is much quieter. As an example, from a z/OS system 1,000 messages can be received within a minute but from a Linux system it might take a week to produce the same amount. As a result of these differences, the analytics team changed the analysis interval size. On a Linux system, IBM zAware requires 60 minutes of current data to produce an accurate anomaly score, while on a z/OS system, IBM zAware requires 10 minutes of current data. Because snapshots are taken every 10 minutes (and updated every 2 minutes) to provide analytical information to you, there are overlaps in Linux snapshots due to the interval size. These time intervals are maintained to allow IBM zAware to produce more granular results and an immediate view of the systems.

Periodicity scorer

Although there were differences in message traffic, it was not too large of a feat for IBM zAware to overcome. Not only did we tweak existing top-notch features to account for these differences, we also incorporated a new scorer, the periodicity scorer. Systems are repetitive in nature; automated scheduling of tasks occurs at regular intervals and human activities are often periodic. The periodicity scorer is able to detect which messages are generated at a regular interval and capture time-of-week activity. This automation is useful in failure prediction; when a system deviates from its normal periodicity, the periodicity scorer is able to detect this deviation and note which message is not in sync. When you use the graphical user interface (GUI), you notice new periodicity information, score, status, and last issued as shown in Figure 1. The periodicity score is determined based on how regular a particular message occurs on given intervals. A high score indicates that the message is regular and the Periodicity Status column indicates this message as “IN_SYNC”, otherwise it is “NOT_IN_SYNC.” And finally, the Last Issued (UTC) column gives the time when this message was last seen. Ultimately, the new periodicity scorer can reveal which messages are periodic. This can be useful when predicting unforeseen events.

Heightened zAwareness

Introducing cutting-edge analysis

BY CHRIS BROOKER, LISA J. CASE, AND YUNLI TANG
sizes are included so that more messages can be compared to reduce the amount of data that you have to analyze. The analysis for that system is summarized into hour-long segments that are displayed instead of individual systems. For z/OS, a system group is equivalent to a sysplex. For Linux systems, the model group is displayed. Additionally, the scores are summarized into hour-long segments to reduce the amount of data that you have to use. All the underlying details and drill-down capabilities are still available.

**Creating model groups**

Systems that run similar workloads often follow a naming convention. An example would be having systems that are named “webserver001”, “webserver002”... and so on. To create a model group that contains the previous web servers, you can write a rule such as “webserver”.” All of the systems whose names begin with “webserver” would then become members of the model group. Suppose a new system, “webserver003” connects to IBM zAware. This system is automatically added to the model group, and analysis on “webserver003” would begin immediately.

**z/OS improvements**

We continue to make further improvements to our z/OS anomaly detection. In addition to the periodicity scorer, new enhancements were made to cluster analysis, specifically to the formation of clusters. Dynamic frame sizes are included so that more messages can be considered for clustering. These features prove to be wonderful additions that have further enhanced the already powerful analytic capabilities.

**UI makeover**

Along with all the great new analytics already outlined, IBM zAware received a facelift in Version 2. The first thing that you notice is that the landing page is no longer the “classic bar chart” view. We added a consolidated view that is called the “Heat Map Table” view as shown in Figure 2. On the Heat Map Table view, groups are displayed instead of individual systems. For z/OS, a system group is equivalent to a sysplex. For Linux systems, the model group is displayed. Additionally, the scores are summarized into hour-long segments to reduce the amount of data that you have to use. All the underlying details and drill-down capabilities are still available.

The main benefit of the Heat Map Table view is to bring more data to the window without the need for a row per system. However, if you are not monitoring many systems with IBM zAware, the Heat Map Table view might not be the best solution for you. You can use the toolbar button to switch back to the classic bar view.

In addition, the tabular analysis view changed. Originally conceived as an accessible version of the classic bar view, now you can use it to put two systems side by side to compare the number of unique messages and anomaly scores for each interval.

The color and number that is displayed in each square represents the maximum anomaly score for any system in the group over that hour-long period. In this way, you can quickly focus on the group and time of the first sign of trouble. If you click one of the squares, the Heat Map Table view is redrawn, and shows you all of the systems in the group. It also sorts the data, putting the system with the highest score for that hour at the top.

When the Heat Map Table view displays systems, clicking one of the squares shows the classic bar view in a detail pane at the bottom of the window. You can drill down into 10-minute intervals as you did in the previous version of IBM zAware.
Sifting through the data
One of the biggest benefits of IBM zAware is that it can help you find the right needle in the right haystack. Still, the amount of data that is presented on the IBM zAware windows can be a bit overwhelming. In several places, powerful filtering was added. For example, from the Interval page, you can view the messages that occur inside that interval. This display can be a list of hundreds, if not thousands, of messages. On the upper right of the grid is a quick-filter box. You can start typing and the grid will narrow down to contain only rows with matching data. You can also use the drop-down menu next to the filter area to build a more powerful filter. This filter can reduce the data that is displayed by values in specific columns, and even apply logic rather than straight matching. For example, the Build Filter dialog can be used to show only those messages that occurred more than two times and contained the letters IEA in the message ID field.

Finishing touches
In Version 1, IBM zAware had plenty of white space. Now the interface is upgraded to use as much space as available in the browser. Additionally, some static elements have been hidden, like page description and the anomaly score key to allow for the important content.

In Version 2, “hashing” support is added so that the browser Back and Forward buttons return you to the previous view or page.

IBM zAware continues to be a powerful tool with the addition of Linux support, improved analytics, and enhanced GUI functions. It is a tool you can use to help reduce the downtime that you might experience in your environment.

Find out more
Learn more about all the new IBM zAware features in the IBM z Advanced Workload Analysis Reporter (IBM zAware) Guide—Version 2.0, SC27-2632.

Version 2 introduces the support to analyze the syslog that is generated by Linux running on z Systems.

IBM zAware is going social!
To better engage our growing set of users, IBM zAware is venturing into the social media world! We set up an IBM developerWorks® community that you can find at ibm.biz/IBM_zAware. The community includes blog entries, a forum for posting questions and comments, as well as other informative topics. Our blog focuses on providing users with a steady stream of content regarding all things IBM zAware.

Additionally, you can monitor our twitter account, which contains up-to-the-minute developments. Follow us @IBM_zAware to stay in touch with trending topics in the world of zAnalytics. Use #zAware to share stories about how zAware helped you.
Improved identification of unhealthy servers with Health Based Routing

BY JESSIE YU, HORST SINRAM, AND KARLA ARNDT

IBM z/OS V2R2 Health Based Routing (HBR) enhancements to dynamic workload routing reduce the business impact of middleware-server health issues, by quarantining a limping (“sick but not dead”) server, and rerouting work to a healthy back-end server.

In the past, individual subsystems have done significant work to avoid storm drain scenarios, where an unhealthy server doesn’t process work, leading to an appearance of lots of capacity, which, if work is routed based only on capacity consideration, attracts even more work to a failing server! Building on these improvements, HBR now goes a step further to provide a broader solution that dynamically routes sysplex work quickly and intelligently to avoid unhealthy or unresponsive servers.

You can set up dynamic work routing in a sysplex by using Workload Management (WLM) services to obtain information and act on recommendations about which systems and server address spaces to send work to. WLM returns a list of servers eligible for work, which includes a relative weighting that indicates the relative number of requests to send to each server. With advanced services IWM4SRSC and IWSMRSRS, you can specify FUNCTION=SPECIFIC to return weight values that are based on factors such as CPU capacity, goal achievement, and abnormal transaction rate.

The recommendations returned by these services can be further refined by a server health value returned by each service. A server health value less than 100 (perfect health) reduces the routing recommendation weight. Using the server health value allows servers to provide health status feedback to WLM, so that less healthy servers get less work. For example, IBM DB2 Distributed Data Facility and Sysplex Distributor can use these services to influence workload routing.

WLM health services

Prior to z/OS V2R2, the programming model for the WLM health services was that only one component, usually the server itself, would provide the health for a given address space. In z/OS V2R2, we introduced the notion of finger-pointing, which allows multiple internal and external components to report on the health of a server. For example, the IWM4HLTH service is enhanced to enable multiple callers to report on a server’s health, identifying themselves and providing reasons for their health ratings. Health ratings from different callers are tracked separately, and for workload routing, the minimum of all current health ratings take effect.

XCF reports on sysplex member health

Most middleware products exploit cross system coupling facility (XCF) and cross-system extended services (XES) services, which already detect problems with their exploiters, making them the perfect candidates for the new finger-pointing function.

For example, XCF signaling is widely used by subsystems, z/OS internal components, and vendor products to communicate within a sysplex. An XCF group member can send and receive messages to other group members. Each message consumes buffer storage, which isn’t released until it’s processed by the receiving member. But if the receiving member fails to process incoming messages quickly, an out-of-buffer condition can occur, impacting the receiving system, the sending system, and potentially the entire sysplex. XCF already detects this condition and takes appropriate action to resolve it. But now, with z/OS V2R2, XCF also uses the enhanced IWM4HLTH service to reduce the health value of the address space that is associated with a sick member. If there are multiple members in the address space, XCF reports the average health score across all members in the address space to WLM.

XCF bases the health score for an address space on data from XCF services used by the subject member. For each service, XCF considers factors such as:

- Whether the relevant user exit routines are stalled
- The depth of the work queue
- Duration of the stall
- How recently progress was made.

As XCF detects problems, it deducts points from the health value of the address space. The longer the problems persist, the more points XCF deducts. XCF invokes IWM4HLTH only when there is a change in the health score.

EVENT 01: HIGH - SERVERHEALTH - SYSTEM: SY1 2014/08/14 - 10:39:18
JOB NAME: IRLM9 ASID: 0150 CURRENT HEALTH VALUE: 0
CURRENT LOWEST HEALTH VALUES:

<table>
<thead>
<tr>
<th>SUBSYSTEM</th>
<th>HEALTH</th>
<th>REPORTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCF</td>
<td>80</td>
<td>0000000000000001</td>
</tr>
<tr>
<td>XCF IRLM9</td>
<td>0</td>
<td>0000000000000000</td>
</tr>
</tbody>
</table>

ERROR: ADDRESS SPACE SERVER CURRENT HEALTH VALUE LESS THAN 100.
ERROR: THIS VALUE MAY IMPACT YOUR SYSTEM OR SYSPLEX TRANSACTION ERROR: PROCESSING.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID AND TO ACTION: DETERMINE THE IMPACT OF THE HEALTH OF THE ADDRESS SPACE TO ACTION: OVERALL TRANSACTION PROCESSING.
XES reports on connector health

XES services allow subsystems, system products, and authorized applications running in a Parallel Sysplex® to use coupling facility (CF) structures for data sharing. To read or write to data in a structure, an application must first connect to the structure.

A few CF structure-related processes, such as structure rebuild and user synchronization points, require coordinated processing across all connectors to a CF structure. XES communicates each event to the connectors, expecting a response. If a connector fails to respond in a timely manner, the process hangs while waiting for the response. This delay can make the structure unavailable, which results in sysplex-wide sympathy sickness. XES already detects and communicates these connector-hang conditions.

And now, with z/OS V2R2, XES also uses the enhanced IWM4HLTH service to reduce the health value of the address space that is associated with a sick connector. If there are multiple connectors in the address space, XES sets the health value to the average health score from all connectors in the address space. XES invokes IWM4HLTH only when there is a change in the health score.

A connector might have multiple outstanding responses to monitored events. Each outstanding response is represented by a monitor block and has its own health score based on how long the response has been outstanding. The connector’s score is the lowest score for all the monitor blocks associated with the connector.

The multifaceted approach to server health is complemented by WLM service IWM4HQLT, which allows applications to retrieve health information for one or more servers, returning both the current and recent health ratings from each of the providers.

Server Health Integration with Runtime Diagnostics (RTD) and Predictive Failure Analysis (PFA)

Runtime Diagnostics is a point-in-time diagnostics tool you can use when you think there is a problem on your system. It provides diagnostic information for eight types of analysis, and is enhanced in z/OS V2R2 to externalize server health problems.

On z/OS V2R2, when you issue the fhzr,analyze command, Runtime Diagnostics invokes the IWM4QHLT service. If any servers have a current health value less than 100, it displays a SERVERHEALTH event in its output along with any other events it finds.

As an additional feature, some of the Predictive Failure Analysis (PFA) checks invoke Runtime Diagnostics when PFA determines that the check’s current value is too low. The new SERVERHEALTH event is returned to PFA by Runtime Diagnostics for any servers with a current health value less than 100 and the event is included in the PFA check’s exception report.

With the new HBR features in z/OS V2R2, XCF and XES report stalled members and hung connectors to WLM. This new health data helps WLM assign lower weights to the sick servers, which in turn helps routers like Sysplex Distributor to route work away from the sick servers using WLM weights. In addition, operators can use the Runtime Diagnostics results to see which servers are sick, who set their health values, and the reasons they were set. These new HBR features are just few of the many proposed solutions that improve dynamic workload routing decisions. We will continue to present new solutions to increase stack-wide resiliency. So stay tuned!
Well, not space exactly, but a generation data group (GDG) with more capacity.

z/OS V2R2 introduces a new extended GDG (GDGE) which can contain up to 999 generation data sets (GDS). Aye Captain, if ye generate one new GDS per day that would be enough for 2 years and a wee bit more. A new IDCAMS parameter, EXTENDED, is added to the DEFINE GDG command that allows the LIMIT to be up to 999. Only the number of managed GDSs is increased, the upper generation number remains 9999 (G9999V00) before wrapping around to 1 (G0001V00).

The catalog search interface field name GDGLIMIT returns only one byte of information, thus is unable to report GDG limit values greater than 255. No information is returned for a GDGE when GDGLIMIT is specified.

A new z/OS V2R2 field name, GDGLIMTE, returns the GDG limit as a two-byte binary number and is valid for both classic GDGs and GDGEs.

Bit 5 (x'08') returned in the byte using field name GDGATTR can be tested to determine if the GDG is extended. If it is on, the GDG is extended. An application that also runs on prior z/OS releases can test this bit to determine whether to use GDGLIMIT or GDGLIMTE.

You can use z/OS Generic Tracker to identify GDGLIMIT field name usage in applications. The usage is reported whenever the field name GDGLIMIT is present, but GDGLIMTE is not. When GDZTRACK is active, you can use the DISPLAY GTZ,TRACKDATA command to show when GDGLIMIT is used.

**Engage!**

To use the EXTENDED parameter, you must enable it. Toleration for systems prior to z/OS V2R2 is limited to failing any attempt to access a GDGE. Have all systems at z/OS V2R2 or later with no expectation of roll back to a prior release before enabling the DEFINE GDG EXTENDED parameter. You must also set the new SYSLPARMLIB(IGGCATxx) parameter GDGEXTENDED(YES) to enable the IDCAMS DEFINE GDG command EXTENDED parameter.

Captain, all of our systems are z/OS V2R2, and we need to enable GDGEs. Make it so, Number One!

A quick way to convert an existing classic GDG to a GDGE is with the following IDCAMS commands:

```
DEFINE GDG (NAME(SMS.TEMP) EXTENDED LIMIT(999))
ALTER SMS.GDGC.* NEWNAME(SMS.TEMP.*)
DELETE SMS.GDGC GDG
DEFINE GDG (NAME(SMS.GDGC) EXTENDED LIMIT(999) SCRATCH PURGE)
ALTER SMS.TEMP.* NEWNAME(SMS.GDGC.*)
DELETE SMS.TEMP GDG
```

**Set your phasers to kill, er scratch!**

Two new options affect GDG SCRATCH on roll-off or EMPTY processing. Occasionally, when users define a GDS, they might intentionally or unwittingly set an expiration date causing SCRATCH to fail when the GDS is rolled off or emptied. A new PURGE parameter is available on z/OS V2R2 DEFINE and ALTER GDG commands that overrides the expiration date when scratching the GDS. The default is NOPURGE on DEFINE, but the default can be changed by adding the GDGPURGE(YES) parameter in the SYSLPARMLIB member (IGGCATxx). The PURGE parameter is only valid with the GDGSCRATCH setting. The PURGE parameter is valid for both classic GDGs and GDGEs.

Similarly, when users define a GDG, they might take the default of NOSCRATCH. The default does not scratch the GDSes during roll off or EMPTY processing. The GDGSCRATCH(YES) parameter in IGGCATxx causes the default to be SCRATCH when defining a GDG.

Both parameters GDGPURGE and GDGSCRATCH are only set when the GDG is defined. Changing the IGGCATxx parameter value after the GDG is defined does not change the GDG setting. You must use the ALTER command to make this change.

Use the MODIFY CATALOG,REPORT command to display the enablement status of GDGEXTENDED, GDGPURGE and GDGSCRATCH.

With z/OS V2R2, GDGs can boldly go where no GDG has gone before!
Would you like better utilization of dump tasks in an HSMplex, along with improved diagnostics for fast replication dumps, and better dump stacking? With facilities provided by IBM z/OS DFSMShsm in z/OS V2R2, you can have them all.

Overview
DFSMShsm in z/OS V2R2 provides new facilities to help with dump processing and problem diagnostics. These new facilities are:

- **Common dump queue.** Previously, a dump request (from a command, auto dump, or fast replication) had to be processed on the DFSMShsm host that initiated the request. That host might not be able to quickly process all requests, due to the number of requests and task restrictions. The solution? The common dump queue (CDQ). With a CDQ, a request can be processed by any system in a group of DFSMShsm hosts. This can increase the number of tasks that are available to perform the work, and improve throughput by distributing the workload instead of concentrating it on a single host’s address space.

- **Fast replication message simplification.** With DFSMShsm fast replication, finding messages can be difficult — it requires searching for the messages for a specific copy pool among all of the other messages in the log data set. DFSMShsm simplifies this task, and makes problem diagnosis easier, by dynamically allocating a unique data set that collects all of the DFSMShsm and DFSMSdss messages for a specific FRBACKUP request or automatic dump of a copy pool.

- **Improvements to dump stacking.** Prior to z/OS V2R2, dump stacking always took precedence over utilizing dump tasks. For example, with the dump class option STACK(10), if there were 10 volumes and 10 dump tasks, all of the volumes were always copied by using a single task and tape. With z/OS V2R2, you can use the new MINSTACK dump class option to indicate that distributing work is more important than stacking. In addition, you can now stack volumes from different copy pools with the same dump class. With these improvements, more volumes are stacked and more tasks and tapes are utilized.

Common dump queue explained
The CDQ is shared by multiple DFSMShsm hosts, managed by a master scheduler (MS), and implemented through the use of the cross-system coupling facility (XCF) for host-to-host communication between an XCF-defined group and its members. The goal of the CDQ is to balance dump processing across the resources that are available in all of the hosts, and return results to the host where the request originated.

Some key attributes of a CDQ environment are:

- A CDQ cannot span an HSMplex.
- You cannot use system affinity.
- You can set a priority of requests.
- A host can be connected to only one CDQ.

Manage a CDQ with the QUERY, CANCEL, HOLD, RELEASE, and ALTERPRI commands, along with a new QUERY CQ(DUMP) option.

XCF commands apply to the CDQ group.
A CDQ allows for flexible configurations. It provides the capability to define multiple queues, and to specify that group members can both receive and process requests, only process requests, or receive requests but not process them. Results are returned to the host that received the request, but progress and status messages are recorded on the host processing the request.

The submitting host receives requests from commands and sends them to the MS host. The submitting host also notifies the user when a command has completed.

The MS is the DFSMShsm host that manages all of the dump requests in the CDQ. It accepts requests from a submitting host and itself. The MS assigns the requests to eligible hosts (called processing hosts), including itself, that have tasks to process the work, while it balances the utilization of the dump tasks among the hosts in the group. The MS also manages the interaction between the processing host for stacking and the submitting host for command completion.

The processing host receives assigned work requests from the MS, completes the work, and interacts with the MS to manage stacking.
A host in the CDQ can be any or all of the host types. See Figure 1.

**Selecting CDQ hosts**

The MS host selects hosts for CDQ work based on the current utilization of dump tasks in each host relative to each other.

A host’s dump tasks are not available if the HOLD DUMP, SETSYS EMERGENCY or SETSYS MAXDUMPTASKS(0) command has been issued. With XCF, the MS becomes aware of these states.

XCF messaging is used to assign requests to remote hosts unless the MS host is selected. In that case, the request is placed on its local queue. The MS assigns work in units of the minimum stack requirements (specified with the new MINSTACK option).

To transition MS responsibilities to another host, use the SETSYS CQ(D(MSC(N))) command.

**Processing CDQ requests**

The assigned processing host selects an available task to process the request.

Status messages and diagnostic trace data are recorded on the processing host. A single dump request might generate many individual volume dumps, with each one processed on a different host. This might make it difficult to put together the message flow of an individual dump command. To simplify the process, use the fast replication message simplification support. The processing host sends fast replication messages to the MS, which consolidates messages for the copy pool in a single unique data set.

As volumes complete processing, the host interacts with the MS to manage stacking up to the maximum (STACK) number of volumes.

**Disconnecting from the CDQ**

The command SETSYS COMMONQUEUE(DUMP(DCONNECT)) removes a member from the CDQ group, making its dump resources unavailable to the CDQ.

- Remote requests that are being processed on this host, and requests that originated from this host, are waited upon for completion before the host is disconnected from the CDQ.
- A HOLD DUMP command that is issued before the disconnect command prevents queued requests from being selected. After the disconnect operation has completed, a RELEASE DUMP command allows the queued requests to be processed locally.
- A disconnection request from the MS host waits for all CDQ active and queued work to complete before disconnecting the MS from the CDQ. After the MS is disconnected, another member in the CDQ group that was defined as an MSC automatically takes over the MS responsibilities.
- For shutdown procedures, first issue a disconnect command on each host that is being removed from the HSMplex.

**Connecting to the CDQ**

The command SETSYS COMMONQUEUE(DUMP(CONNECT(basename))) connects a host to a CDQ. The first host that issues the command defines the XCF group and connects itself. Subsequent hosts are only connected. The first member to join the group that is a master scheduler candidate (MSC) becomes the MS for the CDQ.

You can define the MSC state with the CONNECT option of the SETSYS COMMONQUEUE(DUMP) command. It defaults to YES. Or, you can preset or change the MSC state separately from the CONNECT option with the command SETSYS COMMONQUEUE(DUMP(MSC(Y|N))).

For redundancy, add multiple hosts as MSCs. This ensures that a host takes on MS responsibilities when the current MS is removed due to a disconnection or a system error. You can define hosts that might not have processing capacity for the MS responsibilities with SETSYS COMMONQUEUE(DUMP(MSC(N))).

**Placing requests on the CDQ**

A host that is participating in CDQ processing places its dump requests onto the CDQ for processing. Be aware that:

- HOLD DUMP does not fail WAIT-type requests if another host in the CDQ is not held. This is because this host’s resources are those of the group.
- The SETSYS MAXDUMPTASKS(0) command should be issued if you do not want dump tasks on a host to be used by the CDQ group. This command has the same effect as HOLD DUMP, and if issued from an MS host, does not affect the host’s responsibilities. A HOLD DUMP command that is issued on the MS prevents it from assigning and processing requests for the CDQ and fail WAIT type requests.
- A HOLD DUMP command that is issued on all hosts causes a CDQ to become DISABLED. In this case, new and currently queued WAIT-type requests are failed.

**Figure 1. Hosts in a common dump queue**

- HSM 1 is a submitting and processing host
- HSM 2 is the designated Master Scheduler. It receives all requests, managing them on a single queue, and distributes the requests among the eligible hosts connected to the common queue, through XCF Messaging.
- HSM 3 is a processing host

**Connecting to the CDQ**

The command SETSYS COMMONQUEUE(DUMP(CONNECT(basename))) connects a host to a CDQ. The first host that issues the command defines the XCF group and connects itself. Subsequent hosts are only connected. The first member to join the group that is a master scheduler candidate (MSC) becomes the MS for the CDQ.

You can define the MSC state with the CONNECT option of the SETSYS COMMONQUEUE(DUMP) command. It defaults to YES. Or, you can preset or change the MSC state separately from the CONNECT option with the command SETSYS COMMONQUEUE(DUMP(MSC(Y|N))).

For redundancy, add multiple hosts as MSCs. This ensures that a host takes on MS responsibilities when the current MS is removed due to a disconnection or a system error. You can define hosts that might not have processing capacity for the MS responsibilities with SETSYS COMMONQUEUE(DUMP(MSC(N))).

**Placing requests on the CDQ**

A host that is participating in CDQ processing places its dump requests onto the CDQ for processing. Be aware that:

- HOLD DUMP does not fail WAIT-type requests if another host in the CDQ is not held. This is because this host’s resources are those of the group.
- The SETSYS MAXDUMPTASKS(0) command should be issued if you do not want dump tasks on a host to be used by the CDQ group. This command has the same effect as HOLD DUMP, and if issued from an MS host, does not affect the host’s responsibilities. A HOLD DUMP command that is issued on the MS prevents it from assigning and processing requests for the CDQ and fail WAIT type requests.
- A HOLD DUMP command that is issued on all hosts causes a CDQ to become DISABLED. In this case, new and currently queued WAIT-type requests are failed.
Fast replication message simplification explained
The SETSYS FASTREPLICATION(MESSAGEDATASET(YES HLQ(hlq))) command enables fast replication message simplification. When FRBACKUP requests or auto dump requests with copy pools are processed, DFSMShsm allocates an SMS-managed data set with the prefix hlq. The data set name identifies the copy pool name, location qualifier for a DB2 copy pool, date-time stamp of when the copy pool started, and the status (S,F, or I, to indicate whether the operation was successful, failed, or is in progress).

Message simplification works with or without a CDQ, but message simplification is a must with CDQ processing because the messages are distributed. With a CDQ, copies of the messages are consolidated, by copy pool, to a unique data set for viewing.

In a non-CDQ environment, the host that manages this data set is the host that submitted the fast replication request. With a CDQ, the MS manages the collection of messages from each of the processing hosts.

Figure 2 illustrates the data set naming convention and message format.

Dump stacking improvements explained
The dump stacking improvements in z/OS V2R2 apply to both CDQ and non-CDQ environments.

The MINSTACK option and the new MAXSTACK alias for STACK have been added to the DEFINE DUMPCLASS command. You can override them with equivalent options on the BACKVOL DUMP(DUMPCLASS) command. When you do not specify the MINSTACK option, the command defaults to STACK.

Faster, simpler, and more efficient
With the new function in z/OS V2R2 DFSMShsm, you can improve the throughput of dump tasks, simplify working with fast replication dumps, and better utilize tasks and tapes when dump stacking. What more could you ask for!

Downloading the z/OS Collection Kit
To download the z/OS Collection Kit, follow these steps:
1. Go to the IBM Publications Center: ibm.com/shop/publications/order
2. Select your country, then click “Search for publications.”
3. Type SK4T-4949 in the “Publication number” field and press Enter. SK4T-4949 is the order number for the electronic-only z/OS Collection Kit and matches what was previously available in physical media for base and features.
4. Use a modern zip app such as 7-zip to unzip the file. See the appropriate download pages at: ibm.com/systems/z/os/zos/library/bkserv/index.html

Every “shelf” has its own order number, so you can pick and choose to download only the products you need from the IBM Publications Center. For example, z/OS HCM, SK2T-1377, or z/OS Messages and Codes, SK5T-9251.

Otherwise, to download the entire collection kit:
- z/OS V2R1 SK4T-4949
- z/OS V1R13 SK3T-4271
Improved z/OS subsystem management

BY SCOTT BALLENTINE

The subsystem interface (SSI) is an integral part of the z/OS operating system. The SSI just plugs along in the background, routing information to and from the subsystems that make z/OS work. However, when something breaks down in one of these subsystems, the impact to the system can be severe. In z/OS V2R2, we have some new functions to help ease the pain of managing a broken subsystem.

Initialization routine behavior
Many subsystems are dependent on a subsystem initialization routine to build all of the “stuff” that the subsystem will eventually need. This initialization routine can be specified in several ways, but is most often specified by using the INITRTN parameter in the IEFSSNxx parmlib member or the SETSSI ADD command. Obviously, if the initialization routine does not run, the subsystem will not be able to function.

A common error when installing a new subsystem is that the initialization routine was not found. This happens most often because the initialization routine name was specified incorrectly. Another common situation is that the initialization routine was not installed in an APF-authorized library.

In the past, if the initialization routine could not be found, the subsystem would be defined but because it was not initialized, the subsystem would be inactive. But with z/OS V2R2, the subsystem is no longer defined if the initialization routine was not found. In many cases, this means that you can now issue a SETSSI ADD command to dynamically add the subsystem.

Be warned, though: the system still defines the subsystem for some subsystem initialization errors. The system still defines the subsystem even if you specify incorrect initialization parameters or the subsystem initialization routine ABENDs.

Subsystem deletion
After a subsystem is defined, you’re stuck with it, right? No more. In z/OS V2R2, a new command, SETSSI DELETE, is available to delete a subsystem.

The syntax for the new command is:

SETSSI DELETE, SUBNAME=ssss, FORCE

Although this is a very powerful command that can bail you out of trouble, this is also a very powerful command that can do a lot of damage. The SSI does not know about dependencies that other processes have on a subsystem. Any jobs, products, or other subsystems that are dependent on a deleted subsystem might fail. Also, any resources that are owned by the subsystem, such as storage obtained by the subsystem, will persist after the subsystem is deleted. Finally, not every subsystem can be added dynamically, so even though you can remove a subsystem, you might not be able to add it again. Use caution!

Final words
There are a lot of details that we did not cover here, including the authorization needed for the SETSSI command, updates to the DISPLAY SSI command, and the services available to dynamic subsystems. A few good starting places for more information are z/OS MVS Using the Subsystem Interface, SA38-0679, and z/OS MVS System Commands, SA38-0666.

The ability to easily resolve a missing subsystem initialization routine without deleting the subsystem will eliminate many of the headaches associated with installing a subsystem. But the biggest subsystem interface change in z/OS V2R2 is the ability to delete subsystems. Hopefully, it’s a tool you’ll never need to use, but if you do, you’ll be glad it’s there.
A new SMF record captures the changes to a data set from the time a data set is created until a data set is deleted. This SMF record is used when you want to know how a data set changed over time.

What is a VTOC? VTOC is the volume table of contents and contains data set control block (DSCB) records for each data set on a volume. Each time a data set is manipulated, the DSCB that is associated with this data set is updated.

In z/OS V2R2, changes to a DSCB are recorded in the SMF record 42, new subtype 27 as described in z/OS MVS System Management Facilities, SA38-0667. These new SMF records provide an audit log after the VTOC records for a data set are no longer available. If you select which SMF records to include, the system might not automatically include these new records.

Life of a data set as seen through the VTOC record
When a data set is allocated, one or more DSCB records are created and written to the VTOC. Updates to these records occur when the data set is extended, closed, partially released, or renamed. DSS Defrag/Consolidate and vendor products can also modify VTOC records. When the data set is deleted, the DSCB records that describe this data set are no longer available. Before z/OS V2R2, you were unable to retrieve the history of this data set. With VTOC audit logging, a comprehensive footprint of this data set’s life is available.

Capturing writes to the VTOC
EXCP I/O requests are examined by the EXCP scan routine. It is here that a channel program is scanned and writes to the VTOC are detected. When this occurs, an SMF record is built that includes the DSCB along with information that identifies the system, date and time, job name and ID, and user security token.

Who wrote that VTOC record?
The z/OS operating system components that update DSCBs specify a unique token to identify their activity when they update DSCB records. This four-character field is named SMF42RACT. Diagnostic data that is included in the SMF record helps to identify “unknown writers”. The fields that are located in the new SMF record that can help provide clues to identify the writer are:

• Job name (SMF42RJOB) and ID (SMF42JNO)
• Activity that caused the DSCB update (SMF42RACT)
• Channel program command codes that are used to update the DSCB (SMF42RCMDS)
• Virtual address of the EXCP caller (SMF42RUPSW).

Can anyone identify their activity?
Yes, they can. If you have programs of your own that modify DSCBs, you can identify them, and ISVs can, too. You can do this in one of two ways:

• For calls to CVAFDIR, use the 4-byte CVCLID field.
• For other bare-metal interfaces such as EXCP or XDAP, use the IOBEUSER field in the IOB Extension.

IBM programs use a prefix that begins with A-I. Vendors can include their component ID within the name. Avoid using “SYS.”

Are all DSCB writes recorded?
The answer is no. There are a set of VTOC updates that will not be recorded. This includes:

• Temporary data sets of the form ‘SYSyyddd.Thhmmss.*’ where yy is the last two digits of the year, ddd is the day of the year, hh is the hour, mm is the minutes, and ss is the seconds.
• Data sets on a system residence volume
• I/O to an offline volume
• Updates to DSCB Format 4, 5, and 7
• When multiple DSCBs are written in the same channel program, only the first write is recorded.

Can I disable these records?
You can disable SMF subtypes when SMF data is written to SYS1.MAN data sets. Use parmlib member SMFPROMxx keyword NOTYPE to identify record subtypes that should not be recorded. Specify NOTYPE(42(27)) to disable VTOC audit logging.
Let your system do the tuning!

Workload-based XRC write pacing

BY DAVID SHACKELFORD, EDUARD DIEL, AND SCOTT COMPTON

In most XRC (officially named z/OS Global Mirror, or zGM) environments, write pacing is an essential part of achieving balance between production response times and the recovery point objective (RPO). However, because write pacing is controlled at a volume level, a new set of challenges is introduced. This is especially true in SMS-managed environments, where data placement is left to SMS. If a high importance workload is migrated to a highly paced volume, application response time can suffer. If a low importance (but high-update rate) application is allocated to a low pacing volume, the XRC session can be affected. Hence, keeping XRC tuned by using write pacing to regulate workloads for volumes can be a perpetual battle of tracking down poorly allocated data sets and moving them to give better application and mirroring performance.

Enter workload-based write pacing. This new function uses the importance levels that are assigned by Workload Manager (WLM) in production to control write pacing in the XRC environment. For z/OS workloads, high and low importance workloads can be on the same volume, and XRC will now pace the lower importance workloads first. Now storage management is much easier because you no longer have to segregate workloads by volume or hunt down the offending data sets when allocations change. (Other workloads such as z/VM and Linux on z Systems are still managed on a volume basis.)

Planning for workload-based write pacing

Two measurements of data center success are key in planning to use a disaster recovery (DR) solution such as XRC. The recovery point objective (RPO) measures how much data loss can be tolerated in a disaster and still have acceptable business recovery. In an asynchronous mirroring solution like XRC, the RPO is typically limited to seconds. The mirror will be recoverable to a time somewhat earlier than the actual time, and the amount it is behind is called exposure time.

The other key measurement for DR planning is the application response time and, in particular, the perceived business response time. Keep in mind that an increase in I/O response time might not result in a difference in perceived response time. This difference is key to balancing the needs of the business. Also, different applications have different response-time needs.

XRC write pacing keeps the mirror’s exposure time below the business’s RPO by injecting small delays in I/O response time when the application’s update rate exceeds XRC’s capacity for mirroring the data. Read I/Os are not affected. Use the setting of the write pacing level to control the maximum injected delay.

Workload-based write pacing can differentiate up to six workload categories. Multiple applications or workloads might fit in a category, and some categories might be unused.

Classify application workloads according to how much I/O response time impact they can accept without affecting the perceived response time, using broad categories. Avoid focusing on the details of disk performance.

Enabling workload write pacing

To enable workload-based write pacing, you must change the settings on both the application LPARs and the system data mover LPARs.

Application setup:
WLM Constructs
IEAOPTxx STORAGESERVERMGT=YES

System data mover setup:
SHADOW WorkloadWritePacing(a,b,c,d,e,f)

Figure 1. Workload write pacing
Application system setup

Ideally, on the application LPARs, you already have a satisfying workload manager setup that assigns workloads to service classes. If you don't, you can check out z/OS MVS Planning: Workload Management, SC34-2662.

Workload-based write pacing is controlled on the application system using the same method used to control the I/O Priority Manager function. The IBM Redbooks publication, DS8000 I/O Priority Manager, describes how to set up I/O Priority Manager. You use the same process to set up workload-based write pacing. The service class parameter used to control both functions is importance. Align the importance level with the maximum I/O delay values that were determined during planning. As an example, the category System would get the “System” importance level, the category High Priority OLTP would be importance 1, the Low Priority OLTP category would be importance 2, and so forth.

After you have Workload Manager set up, it's simply a matter of adding the following statement to your existing IEAOPTxx member in SYS1.PARMLIB:

```
STORAGESERVERMGT=YES
```

DR site system setup

You also have to configure the LPARs for the system data mover. Your existing write pacing parameters are still valid, and they complement the new SHADOW WorkloadWritePacing parameter in your XRC parmlib member. The WorkloadWritePacing parameter specifies to XRC which pacing levels to assign each of the importance levels, starting with the highest importance level and ending with the discretionary importance level.

Workload-based write pacing uses the importance value on each write I/O to determine the write pacing level to be used in the calculation for how much write pacing delay to inject, if any. For more information, see DFSMS Advanced Copy Services, SC23-6847.

You must set six values on WorkloadWritePacing parameter, with each value corresponding to importance levels 1 - 6. If you use fewer than six importance levels, set the unused importance levels equal to the value of the previous importance level. For example, if importance level 6 is not in use, use the same value as importance 5. The levels must be increasing in value:

```
SHADOW WorkloadWritePacing(1,5,A,C,E,E)
```

Verifying function availability and activation

A Y in the first column under the WP heading in the new XQUERY STORAGECONTROL XFEATURES report means that the primary controller is capable. In the following example, the N in the second column means that software support for the WorkloadWritePacing parameter was not enabled in the parmlib the last time an XSTART was done.

```
XQUERY STORAGECONTROL _ XFEATURES REPORT - 001
  T LIC FEATURES
  SSID Y LEVEL FU D W SL ER IR EX WP
  -----------------------------------------------------------
  DAA8 U 7.7.41.34 YY Y Y NN YY Y ED YN
  UTL=1 SUTL=0
  DATA CONSISTENT(2015.080 01:17:15.592775) DELAY(00:00:00.76)
  XQUERY STO _ XFEATURES REPORT COMPLETE FOR SESSION(S1)
```

By doing a quick XSUSPEND and restart, you are in the exciting new world of workload-based write pacing.
Verifying actual injected pacing

To verify that everything is working correctly with workload-based write pacing, you have to run some workloads. The following example demonstrates an XQUERY PACE report for a single volume (MYD201) that has three workloads running simultaneously on it. All of the workloads are equal in this example, but the most important workload (designated with WLM1) gets significantly more attention than the discretionary workload (designated with WLM6).

<table>
<thead>
<tr>
<th>XQUERY VOLUME _ PACE REPORT - 003</th>
</tr>
</thead>
<tbody>
<tr>
<td>WLM PACING WLM1(1) WLM2(5) WLM3(A) WLM4(C) WLM5(E) WLM6(E)</td>
</tr>
<tr>
<td>PRIM   SEC       SC SC RES   THD RES WRT PACE</td>
</tr>
<tr>
<td>VOL    VOL STA SSID SN ID CNT CNT RATE RATE MS</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>MYD201 MYD202 DUP DAA8 -- 01 00008E8 1400@F 0AEF 0AEF 0000.3</td>
</tr>
<tr>
<td>WLM1 1400:1 0530 0530 0000.0</td>
</tr>
<tr>
<td>WLM2 1400:5 0441 0441 0000.1</td>
</tr>
<tr>
<td>WLM6 1400:E 017C 017C 0001.7</td>
</tr>
</tbody>
</table>

1 VOLUME(S) MEET REQUESTED SPECIFICATION
TOTAL=2 DUP=1 CPY=0 PND=0 SUS=0 SEQ=0 UTL=1
PAV=0
MONITOR INTERVAL STATISTICS AS OF 2015.080 00:09:25 UTC
XQUERY VOLUME _ PACE REPORT COMPLETE FOR SESSION(S1)

HyperPAV and write pacing

Workload-based write pacing of I/Os introduces an additional need within the system to manage alias devices for the parallel access volume (PAV). HyperPAV aliases are managed on an as-needed basis. If an alias is required to start an I/O, one is quickly assigned to the base device and the I/Os are started. You do not want all your aliases to be consumed by write I/Os that are being paced because this could adversely affect your overall I/O throughput.

This new function uses the importance levels that are assigned by Workload Manager (WLM) in production to control write pacing in the XRC environment.

A new enhancement in the IBM System Storage DS8870 storage controller allows host operating systems to better understand when workload importance-based write pacing is in effect. With this new function, the I/O supervisor automatically takes into account workload importance when it assigns aliases for write I/Os that are subject to these pacing delays. No extra configuration is required. The I/O supervisor can then ensure that aliases are more readily available for higher priority work and read I/Os that are not subject to pacing delays.

This new alias management scheme utilizes a graduated scale for required alias availability based on workload importance. Higher importance work might continue to utilize aliases for write I/Os when pacing delays are in effect at lower alias availability levels, while lower importance work’s write I/Os might be delayed before getting started because too few aliases are available. In this way, the important write I/Os take precedence, helping z/OS meet your overall system workload goals.

The benefit to you

Using workload-based write pacing makes your DR solution easier to manage. By differentiating the impact of mirroring on the application, it helps you meet the DR site RPO, while still protecting higher importance applications from excessive impact due to pacing. By moving the point of control from the volume level to the workload level, it removes the need to segregate workloads by volume. And by providing more detailed information in the XRC pacing report, it makes the impact of mirroring easier for the business to understand.
When you depend on having your data available for transactions and processes, multiple copies of the data are typically required. Two products that help provide continuous availability and disaster recovery to help you manage this aspect of the IT environment are:

- Geographically Dispersed Parallel Sysplex (GDPS)
- IBM Tivoli® Storage Productivity Center for Replication for z Systems Basic HyperSwap® (a copy services solution for z/OS).

These solutions provide the capability for you to establish relationships between storage controllers for devices that need to be mirrored. And, when HyperSwap is used, to communicate with z/OS to load and monitor sets of devices for HyperSwap events.

Today, control unit-based synchronous data replication (also known as Metro Mirror) is accomplished by using synchronous Peer-to-Peer Remote Copy (PPRC). Pairs of devices are configured such that the PPRC relationship is created between a primary volume and a secondary volume. Device pairs can be at the same site or at a limited distance (up to 100 km, or greater distances with a line cord feature code RPQ 8A1871).

The added latency is from the synchronous mirroring management (that can be up to 330 microseconds) and the distance (at least 10 microseconds per km) when performing the replication. While physics dictates the latency due to distance, there's an opportunity to reduce the latency that gets added when managing the synchronous relationship.

Enter zHyperWrite
IBM zHyperWrite™ combines concurrent control unit-based synchronous mirroring and software mirroring to provide substantial improvements in IBM DB2 log-write latency. DB2 performs both asynchronous and synchronous log writes. The synchronous log writes have a direct effect on the performance of transactions and can serialize other functions as DB2 locks are held during this process. With zHyperWrite enabled, DB2 synchronous log writes to a primary volume (item 1) are mirrored to fully duplexed secondary volumes by DFSMS Media Manager (item 3), bypassing PPRC (item 2), and eliminating a significant portion of the I/O latency. For any secondary volume that is not in full duplex state, the data replication will be handled by control unit mirroring. Additionally, any write I/O that isn't appropriate for zHyperWrite, (for example, non-log writes) continues to use control unit synchronous mirroring (items 1 and item 2).

Eligible?
For zHyperWrite eligibility, a primary and secondary volume pair must be in a Metro Mirror relationship in full duplex status, and must be managed by Hyper Swap by either Tivoli Storage Productivity Center for Replication for z Systems or GDPS.

When the requirements are met for DB2 log writes, z/OS Data Facility Storage Management Subsystem (DFSMS) Media Manager can initiate write I/Os to the primary log volume and any secondary log volume in parallel. z/OS instructs the control units that are involved to avoid doing the synchronous mirroring for these types of I/Os. Data mirroring for secondary volumes that are not in a Metro Mirror relationship in full duplex status are handled by the control unit. Assuming that all the secondary volumes are eligible, zHyperWrite I/O results in total elapsed time that is equivalent to the maximum response time for any of the I/O operations.

If the IBM Systems Storage DS8870 detects zHyperWrite I/O to a secondary volume that is not in full duplex state, the I/O fails. If a zHyperWrite I/O to a primary or any of the secondary volumes fails for any reason, Media Manager redrives the I/O to the primary volume without zHyperWrite and allows PPRC to perform the data mirroring to the secondary volumes.

Figure 1 shows an example HyperSwap environment. When synchronous replication is used and a write I/O is performed (item1), there is added time for the write I/O while the control unit mirrors the data to peer devices (item 2). The added latency is from the synchronous mirroring management (that can be up to 330 microseconds) and the distance (at least 10 microseconds per km) when performing the replication. While physics dictates the latency due to distance, there’s an opportunity to reduce the latency that gets added when managing the synchronous relationship.
Secondary devices that are accessed by using zHyperWrite are monitored in RMF Monitor I and RMF Monitor II in the same way as online devices except that the volume serial number is blank, which indicates that the device is offline. The data is saved in SMF 74 subtype 1 record. If RMF Monitor II records are requested, data is saved in SMF 79 subtype 9 record.

DFSMS counts the number of I/Os that occur for a data set including DB2 log data sets. You might see an increase in the number of I/Os that occur for DB2 log data sets when Media Manager is performing software mirroring. To help distinguish this, z/OS provides a new field (S42DSIOS) in an SMF 42 subtype 6 record that contains the total number of software initiated writes to Metro Mirror secondary devices. The SMF data can help you understand and account for the additional I/O activity.

Test results
Synchronous DB2 log writes are critical to the nature of DB2 performance, especially when using Metro Mirror to replicate logs from one storage controller to another. This can affect online transaction processing (OLTP) workloads and batch DB2 insert jobs. OLTP transactions typically perform frequent commit operations that cause synchronous DB2 log write operations.

IBM system test
The IBM system verification team measured performance improvements for zHyperWrite technology, and compared the results with and without zHyperWrite in effect. The team used:

• z/OS V2R1
• IBM DB2 for z/OS 10.0.0
• z/OS hosts on an IBM zEnterprise EC12 (zEC12)
• FICON Express8S channels
• DS8870 with microcode that included zHyperWrite.

To manage the HyperSwap Metro Mirror configuration, the team used Basic HyperSwap in Tivoli Storage Productivity Center for Replication for z Systems, Version 5.2.3. The tests were conducted with a configuration of short (zero) distance between z/OS hosts and storage controllers, and short (zero) distance between primary and auxiliary storage controllers.

The team used a test program simulating OLTP to study the performance effects of zHyperWrite feature. The team studied the commit response times as a function of the number of log control intervals (CI) created divided by the number of commits where the number of CIs per commit was limited to just a few.

The numbers
The left graph in Figure 2 shows the average commit time for a single log page write, with results that show that zHyperWrite saves 300 microseconds per commit. The right graph in Figure 2 shows the average commit time with and without zHyperWrite, which indicates a 43% reduction in the DB2 log write latency and corresponding commit response time.

By enabling zHyperWrite:

• Synchronous DB2 log writes are performed faster
• Overall transaction throughput is improved
• DB2 locks held during log writes are released more quickly.

Along with overall improved resilience for workload spikes, there's additional capacity for growth, and potential cost savings from workload consolidation.

Time to accelerate
zHyperWrite is provided in z/OS V2R2 and the support is enabled by default. If support was previously disabled, you can enable it by using the IECIOSxx SYS1.PARMLIB member.

Turn on the software acceleration feature for DB2 to use zHyperWrite. To turn it on, set the REMOTE_COPY_SW_ACCEL parameter on the DSN6LOGP ZPARM macro to YES (the default is NO). You can turn the parameter option on (and off) without restarting DB2. The DB2 DISPLAY LOG command is updated to indicate that zHyperWrite is active.
Hardware can fail. Humans are imperfect. Disasters happen. You need protection. The IBM System Storage DS8000 series provides various replication technologies that can help protect your business. Now, with the IBM Multiple Target Peer-to-Peer Remote Copy (Multiple Target PPRC), your protection can go further.

A loss of a disaster recovery site can severely affect your business. If local production fails, swapping to a Metro Mirror target allows applications to continue running. However, without another target to act as a backup for disaster protection, you are left unprotected from a subsequent failure. The solution to this problem is Multiple Target PPRC.

Before the availability of Multiple Target PPRC, it was possible for a primary volume to mirror its data to only one secondary volume. Now, the same primary volume can have more than one target. You can enable data to be mirrored from a single primary volume to two different secondary volumes, which can even be at two different target geographic locations! By allowing an extra replication target, a Multiple Target PPRC environment can keep your applications continuously available and maintain disaster recovery even after a site failure.

IBM PPRC replication technologies
You can combine the following replication technologies to provide advantages in unlimited configurations and deployment scenarios. Scenarios include continuous availability, disaster recovery, and data migration.

• Metro Mirror is a synchronous mirror that provides the potential for a zero Recovery Point Objective (RPO).
• Global Copy is an asynchronous copy of data and is suitable for unlimited distances.
• Global Mirror is an asynchronous mirror of data with logical data consistency.

Continuous availability
A standard single Metro Mirror pair topology is often used to provide continuous availability within metro distances at a single site or in a two-site topology. Referring to Figure 1, if a problem is detected at your primary site (H1) a Hyper-Swap operation to your secondary location (H2) can seamlessly keep your applications in operation.

In a two-way Metro Mirror pair topology, data can be synchronously mirrored to different secondary locations at the same time H1:H2 and H1:H3. You can create this environment by establishing links between all sites, and then establish the two Metro Mirror pairs. Because these pairs are independent of each other, if either secondary site (H2 or H3) is lost, the remaining site will continue to mirror the primary (H1).

Figure 1. Metro Mirror pair topology
If a disaster occurs at H1, applications can continue running, thanks to H2’s mirror. You can quickly establish a Metro Mirror pair H2:H3 to maintain HyperSwap capability. You can establish this pair without requiring a full copy of the data. This capability is called Multiple Target Incremental Resynchronization (MTIR).

When two PPRC relationships are created on a primary volume, the DS8870 detects that it is now part of a Multiple Target PPRC configuration. The DS8870 automatically sends commands to each of the secondary volumes and instructs them to create an MTIR pair that points to the other secondary volume. These pairs provide a change-recording mechanism to track which data is potentially different between the two secondary volumes. This mechanism is what allows for the resynchronization to be an incremental copy rather than a full copy. The MTIR pairs are created in a suspended state. When there’s a loss of H1 they can be quickly converted to an active Metro Mirror and maintain the HyperSwap capability.
Metro Mirror plus Global Mirror

With Multiple Target PPRC, you do not have to choose between a high-availability local copy and a long-distance disaster recovery copy. You can have them both. Furthermore, if an outage occurs at any one of the three sites, replication can continue between the remaining two sites. No single site failure will cause the loss of replication.

With a Metro Mirror plus Global Mirror topology, the Metro Mirror relationship provides a local, HyperSwap capability. The additional Global Mirror relationship provides a long-distance disaster recovery capability. Multiple Target PPRC allows for both of these capabilities at the same time.

Referring to Figure 1, a Metro Mirror plus Global Mirror topology uses Metro Mirror for H1:H2 and Global Mirror for H1:H3. An outage event at either of the secondary sites, H1 or H2, does not impact the mirror to the other secondary site. For example, the H2 Metro Mirror secondary site might be taken down for maintenance or fail due to some error event without impacting the Global Mirror disaster recovery capability. Similarly, if an event causes the Global Mirror relationship to H3 to stop, Metro Mirror still provides an addition copy of data at H2.

Referring to Figure 2, if there’s a failure at the primary H1 site, a HyperSwap replication to the Metro Mirror secondary at H2 allows applications to seamlessly continue operation without impact. You can quickly restart Global Mirror from H2 to H3 by using the Incremental Resynchronization feature of Multiple Target PPRC.

Cascaded MGM

Cascaded Metro/Global Mirror (MGM) refers to a topology where data is first mirrored synchronously by Metro Mirror, from H1 to H2, and then from the H2 site to a remote disaster recovery site at H3 by using Global Mirror. Like Multiple Target PPRC with Metro Mirror and Global Mirror, this topology provides both a local high-availability capability and a long-distance disaster recovery capability at the same time. Multiple Target PPRC provides several benefits and simplifications in such an MGM topology.

With a cascaded MGM configuration, there might be an outage at the primary H1 site. If so, a HyperSwap operation to the intermediate H2 site allows applications to continue to run while also retaining the disaster recovery protection of Global Mirror to the remote H3 site. After a recovery of the original local site, Multiple Target PPRC provides the capability to resume Metro Mirror directly from H2 back to H1. This configuration creates a Multiple Target MGM configuration and restores the high-availability capabilities while never losing the disaster recovery capability. If necessary, you can start another HyperSwap operation to move the application I/O back to the original H1 site. You can do it all without ever pausing the Global Mirror operation.

Migration made simpler

Without Multiple Target PPRC, when the time comes for a technology upgrade, data center move, or a replacement of a primary or secondary disk, the process can be long and complicated. Often, the upgrade requires some kind of application impact.
Multiple Target PPRC makes the task of migrating your mirrored disk easier. It no longer needs to be an excruciating process that includes quiescing applications to make the final move. Using the Multiple Target PPRC feature, you can introduce a new primary or secondary disk while your existing mirror continues unimpeded.

Replacing peer-to-peer remote copy primary

Whether you are using Metro Mirror, Global Copy, or Global Mirror, you can now replace a PPRC primary device without interruption to your replication. The goal of a PPRC primary migration is to convert from an H1:H2 topology to an H3:H2 topology.

Because Multiple Target PPRC allows a PPRC Primary to have more than one secondary, you can initiate the migration by establishing H1:H3. Production continues accessing the H1 devices, and mirroring H1:H2 is maintained. You can run the initial copy to H3 synchronously with Metro Mirror or you can run it asynchronously with Global Copy, and then convert it to Metro Mirror when the copy is near completion. After the status of H1:H3 reaches DUPLEX state, you can start a HyperSwap operation and convert the MTIR relationship to an active pair, which results in the final H3:H2 topology.

If you are using Global Mirror, you can define the Global Mirror session for H3 and after the swap, add H3 to the Global Mirror session. You can do these steps while the H1:H3 copy is in progress, so that H3 begins participating in the Global Mirror session as soon as the swap completes.

Replacing peer-to-peer remote copy secondary

When you replace a PPRC secondary you are starting with an H1:H2 topology, and ending with an H1:H3 topology. You can run this migration process without interrupting your replication.

Multiple Target PPRC allows a PPRC primary to have more than one secondary. This means you can initiate the migration by establishing H1:H3 while production continues accessing the H1 devices, and mirroring to H2. After H1:H3 reaches DUPLEX state, remove the H1:H2 pairs and you have an H1:H3 topology.

Thanks to Multiple Target PPRC in both of the preceding migration scenarios, migration is accomplished without quiescing production activity, or going through the round-robin cascade of device switching. Multiple Target PPRC opens the door to all kinds of possibilities, which include:

- Mirroring a primary device simultaneously to more than one secondary device
- Choosing from several combinations of synchronous and asynchronous mirroring from the same device
- Taking advantage of simplified high-availability and disaster recovery steps
- Using the simplified disk-migration capability, with little to no impact to production.

zFavorites

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

ibm.com/systems/z/os/zos/library/zfavorites

Scan with your phone’s QR code reader
Simplify z/13 maintenance with SMP/E

BY GREG DAYNES

The z/OS Management Facility (z/OSMF) was released in 2009 as a web-based interface for several z/OS functions. The "new face of z/OS" has grown over the last five years to encompass a host of new features, including cutting-edge cloud application programming interfaces. Along the way, z/OSMF moved from the WebSphere Application Server OEM Edition to a much smaller and customized WebSphere Liberty Profile server.

The IBM z13 server is the newest IBM z Systems server. While the z13 is based on earlier IBM zEnterprise System and other servers, it is distinct in its support of newer workloads, such as analytics and cloud computing, as well as traditional mission-critical applications.

As with prior generations of servers, most z/OS software support is provided by service in the form of PTFs, which are identified by one or more SMP/E fix categories. SMP/E fix categories are specified in FIXCAT HOLDDATA. SMP/E uses these categories to identify PTFs that are not installed on your system. This provides an easy way to verify that you have all of the PTFs that are needed for z13 support installed on your system.

**Hold that data**

To get started, you need to acquire the latest 2-year HOLDDATA file and receive it using the SMP/E RECEIVE command. This file is included in all IBM service orders. It can also be requested specifically in a RECEIVE ORDER CONTENT(HOLDDATA) command or transferred from this Enhanced HOLDDATA for z/OS web page:

ftp://service.boulder.ibm.com/s390/holddata/full.txt

If you use the File Transfer Protocol (FTP) site, you must select the 2-year file.

After you receive the HOLDDATA file, you can use the REPORT MISSINGFIX command to determine whether PTFs in any fix category that you are interested in are missing from your system (that is, the PTFs are not installed). The key is to specify all of the SMP/E fix categories that you are interested in in your REPORT MISSINGFIX command. This isn't as straightforward as you might think.

**Service with a smile**

IBM has created three fix categories to identify required service for the z13, PTFs needed to exploit z13 capabilities, and PTFs recommended to fix known problems:

- IBM.Device.Server.z13-2964.RequiredService
- IBM.Device.Server.z13-2964.Exploitation
- IBM.Device.Server.z13-2964.RecommendedService

However, it is likely that these fix categories are not sufficient. It is always a good practice to include the fix categories for the server that you are on, as well as any generation of servers that you are skipping. This means that you would likely include one or more of the following fix categories:

- IBM.Device.Server.zBC12-2828*
- IBM.Device.Server.zEC12-2827*
- IBM.Device.Server.z114-2818*
- IBM.Device.Server.z196-2817*
- IBM.Device.Server.z10-BC-2098*
- IBM.Device.Server.z10-EC-2097*

Depending on which hardware and software functions you use in your installation, it is also likely that you need to include one or more of the following fix categories:

- IBM.Device.Server.*.ParallelSysplexInfiniBandCoupling
- IBM.Device.Server.*.ServerTimeProtocol
- IBM.Device.Server.*.UnifiedResourceManager
- IBM.Device.Server.*.zHighPerformanceFICON
- IBM.DB2.AnalyticsAccelerator*
- IBM.Function.zEDC

Note the use of the generic wildcard character (*) in each of the last two sets of categories. This ensures that SMP/E processes all of the appropriate fix categories with the least amount of typing on your part. For a complete list of all IBM fix categories, see:

ibm.com/systems/z/os/zos/features/smp/e/fix-category.html
Your next stop: The target zone

After you have the complete list of fix categories that you are interested in, it’s a snap to determine whether PTFs associated with those categories are installed. A single SMP/E REPORT MISSINGFIX command can scan your GLOBAL zone for FIXCAT HOLDDATA, matching the FIXCAT values specified on the command, and then compare the APARs identified in the FIXCAT HOLDDATA with the PTFs installed in the specified target zones to produce a report to identify any unresolved APARs. In other words, the command reports which PTFs are missing (that is, not installed) for the specified fix categories. Furthermore, it produces a customized job that is used to obtain any PTFs that have not already been received using the RECEIVE ORDER command, and then install any missing service using the APPLY CHECK command.

The following sample SMP/E command identifies any missing service for a z13, including all service for an IBM zEnterprise EC12 (the server that is in use), all service for an IBM zEnterprise BC12 (a server skipped), as well as all hardware-related categories that are not specific to the z13. The command checks target zones for three different z/OS environments: z/OS V1R12 (target zone TGT112), z/OS V1R13 (target zone TGT113), and z/OS V2R1 (target zone TGT21).

```
SET BDY(GLOBAL)
REPORT MISSINGFIX ZONES(TGT112,TGT113,TGT21)
FIXCAT(IBM.Device.Server.z13-2964.RequiredService,
IBM.Device.Server.z13-2964.Exploitation,
IBM.Device.Server.z13-2964.RecommendedService,
IBM.Device.Server.zEC12*,
IBM.Device.Server.zBC12*,
IBM.Device.Server.*.ParallelSysplexInfiniBandCoupling,
IBM.Device.Server.*.ServerTimeProtocol,
IBM.Device.Server.*.zHighPerformanceFICON,
IBM.Device.Server.*.UnifiedResourceManager,
IBM.Function.zEDC,
IBM.DB2.AnalyticsAccelerator*)
```

---

In this output, you can easily see all of the required, recommended, and exploitation PTFs that are required on the z/OS V1R12 system for the z13, including some High Performance FICON for z Systems (zHPF) fixes.

So, with as few as two simple steps, you can use SMP/E to verify that all of the service required on z/OS to run on, and even exploit, a z13 server is installed, as well as identify any PTFs that are not installed.
No doubt by now you’ve heard of Shared Memory Communications over Remote Direct Memory Access (SMC-R) or RDMA over Converged Ethernet (RoCE). These new technologies promise fast and efficient communications across z/OS systems (see the article “Get started with SMC-R on z/OS V2R1 for fast and efficient network communication” in z/OS Hot Topics Newsletter Issue 28, August 2014, GA32-0892-01).

SMC-R can provide significant latency reductions and CPU savings for TCP/IP communications across z/OS systems. However, you wonder how much, if any, benefit SMC-R can provide in your environment. You can gain some insight by reviewing SMF records, analyzing Netstat displays, and crawling through traces.

To assist in that effort, the SMC Applicability Tool (SMCAT) is now available. When activated, SMCAT monitors the TCP workload for a list of configured destination IP addresses or subnets and produces a report that can be used to project the potential benefits of SMC-R. It does not require the SMC-R function to be activated or any special hardware, and is provided in the following PTFs:

- z/OS V1R13: PTF UI24872
- z/OS V2R1: PTF UI24762.

Configuring and activating SMCAT monitoring

SMCAT is activated by the new VARY TCPIP command SMCAT: VARY TCPIP,,SMCAT,,user.tcpparms(smcatconfig) where user.tcpparms(smcatconfig) is your configured data set that specifies the monitoring interval and destination IP addresses or subnets to monitor. Figure 1 shows a simple SMCAT configuration data set for one IP address and one subnet. In this case, SMCAT monitors and reports on the TCP workload to one real IP address and a VIPA subnet for a two-hour period.

```
; SMCAT CONFIGURATION DATA SET.
; MONITOR INTERVAL = 2 HOURS (120 MINUTES).
; MONITOR WORKLOAD TO ONE REAL IP DESTINATION ADDRESS AND
; ONE VIPA SUBNET.
SMCATCFG INTERVAL 120
IPADDR
; TCP DESTINATION - REAL IP ADDRESS
10.20.110.1
; TCP DESTINATION - VIPA SUBNET
10.12.10.20/16
```

Figure 1. SMCAT data set configuration example
**Tips:** You can determine what IP addresses and subnets to specify in the configuration file by following these steps:

1. Select one or more remote z/OS systems you want to monitor TCP/IP communications from the current system (the one SMCAT is running on).

2. If network traffic patterns between the current system and other z/OS systems is similar, choose one of them and extrapolate the results to include communications to the other systems.

3. Identify all destination IP addresses of the remote z/OS systems that might be used by workloads on the current system.

4. If virtual IP addresses (VIPAs) are being used (either dynamic or static) and the VIPAs are allocated from a dedicated subnet, specify a subnet instead of individual IP addresses to simplify the configuration.

5. If destination VIPAs are not in the same subnet as the remote system’s physical connectivity, also specify the host IP addresses of the OSAs that are used on the remote system to access the VIPAs.

6. If real IP addresses are used for communications, you can specify a subnet only if the subnet is entirely contained on z/OS systems that you want to monitor. If other hosts that are not z/OS reside on the same subnet, their traffic will be incorrectly counted.

**The SMCAT monitoring report**

At the end of the monitoring interval, SMCAT produces a report (see Figure 2) in the TCP/IP job log that contains several useful statistics:

- The amount of TCP traffic between this z/OS system and the identified remote systems
- Whether the traffic patterns are well suited to SMC-R protocol
- Traffic pattern statistics that provide insights on whether the benefits for SMC-R enablement will be primarily in lower latency, CPU reduction benefits, or both
- Whether the SMC-R benefits can be realized by using the existing IP network topology (all systems in the same subnet) or whether network configuration changes are required to enable SMC-R (to place the systems in the same subnet).

**SMC Applicability Tool Sample Report**

**TCP SMC-R traffic analysis for matching direct connections**

---

Connections meeting direct connectivity requirements

25% of connections can use SMC-R (eligible)
10% of eligible connections are well-suited for SMC-R
14% of total traffic (segments) is well-suited for SMC-R
2% of inbound traffic (segments) is well-suited for SMC-R
16% of outbound traffic (segments) is well-suited for SMC-R

**Internal Details:**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total TCP Connections:</td>
<td>120</td>
</tr>
<tr>
<td>Total SMC-R eligible connections:</td>
<td>24</td>
</tr>
<tr>
<td>Total SMC-R well-suited connections:</td>
<td>22</td>
</tr>
<tr>
<td>Total outbound traffic (in segments)</td>
<td>110000</td>
</tr>
<tr>
<td>SMC-R well-suited outbound traffic (in segments)</td>
<td>22000</td>
</tr>
<tr>
<td>Total inbound traffic (in segments)</td>
<td>100000</td>
</tr>
<tr>
<td>SMC-R well-suited inbound traffic (in segments)</td>
<td>14000</td>
</tr>
</tbody>
</table>

**Application send sizes used for well-suited connections:**

<table>
<thead>
<tr>
<th>Size</th>
<th># sends</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=1500</td>
<td>200</td>
<td>35%</td>
</tr>
<tr>
<td>&gt;1500 and &lt;=4000</td>
<td>130</td>
<td>24%</td>
</tr>
<tr>
<td>&gt;4000 and &lt;=6400</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>&gt;6400 and &lt;=16K</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>&gt;16K and &lt;=32K</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>&gt;32K and &lt;=64K</td>
<td>50</td>
<td>10%</td>
</tr>
<tr>
<td>&gt;64K and &lt;=256K</td>
<td>109</td>
<td>22%</td>
</tr>
<tr>
<td>&gt;256K</td>
<td>0</td>
<td>0%</td>
</tr>
</tbody>
</table>

**Interval Details:**

- Total TCP Connections: 120
- Total SMC-R eligible connections: 24
- Total SMC-R well-suited connections: 22
- Total outbound traffic (in segments): 110000
- SMC-R well-suited outbound traffic (in segments): 22000
- Total inbound traffic (in segments): 100000
- SMC-R well-suited inbound traffic (in segments): 14000

For more information about SMC-R and SMCAT, including FAQs, best practices, common problems, and reference documents, see:

ibm.com/software/network/commserver/SMCR.

For more guidance on configuring and using SMCAT, see z/OS Communications Server: IP System Administrator’s Commands, SC27-3661 in IBM Knowledge Center:

ibm.com/support/knowledgecenter/SSLTBW_2.2.0.
Hints, tips, and best practices for z/OS V2R2 documentation

BY SUSAN SHUMWAY, RITA BEISEL, AND TOSHIBA BURNS-JOHNSON

IBM Knowledge Center is the new single point of access for IBM product documentation. It’s colossal!! It’s stupendous!! It’s… perhaps a bit daunting? Don’t fret. We’re here to show you how to find the z/OS information you need.

Step right up! IBM Knowledge Center basics

Logging in to IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSLTBW/welcome) with your ibm.com user ID reveals a world of delights including the ability to create your own collections and to save search criteria.

Select topics or sections from anywhere in IBM Knowledge Center and organize them into collections. Use collections to quickly view relevant content, or to print the whole menagerie as one PDF document.

The following search features help you locate relevant content:

• Search filters: Set a search filter to narrow your search to a specific product release. The z/OS releases are under IBM Operating Systems > z Systems Operating Systems > z/OS. Keep in mind that the search filter stays set until you reset it or click within another product’s navigation tree.

• Search operators: Use search operators including intitle, intext, and inurl to further narrow your search results. For example, to search for ICSF in z/OS Migration, type ICSF inurl:e0zm100 & inurl:v2r2 in the search field. Voila! Your search is scoped to just one book.

• Search results URL: Hover over the topic title link in the search results to determine if the listed book ID for the topic matches that of the book you want. For example, book ID e0zm100 is part of the URL for topics that reside in z/OS Migration.

• Saved search criteria: Save up to ten search criteria to find relevant content even faster in the future.

Tip: When you read content, open linked topics in new browser tabs so that each original page stays scrolled to the correct spot.

Juggle multiple tools! More ways of searching

Some of your fellow z/OS documentation users are combining IBM Knowledge Center with the following tools to achieve even better search results:

• BookManager®: Search the z/OS V1R13 content in BookManager format at ibm.com/systems/z/os/zos/library/bkserv/v1r13books.html. After you find the content you want, click the picture frame icon to determine your location in the library. Then, using the table of contents in IBM Knowledge Center, navigate to the same book and section to see the current information.

• Information Centers: Download the z/OS V1R13 ISO file (SK5T-7089-03) from the IBM Publications Center at ibm.com/shop/publications/order. Burn a DVD from the ISO file, then use the DVD on your workstation to install the z/OS V1R13 information center. For the z/OS V2R1 content, go to the IBM Publications Center to download SC27-8400. After downloading, unzip the package and substitute the V2R1 documentation plugins for the V1R13 plugins in the Information Center plugins directory.
**New**

**IDCAMS SHCDS LISTSTAT**

Frustrated that you have to close your files just to see the latest catalog stats? Wondering when that next opportunity will be? Well, stop wondering, because the time is NOW! Introducing the new IDCAMS SHCDS LISTSTAT command, where file statistics you could only dream about are now at your fingertips!

Whether your files are open or not, the new LISTSTAT command displays sysplex-wide catalog statistics for your VSAM record-level sharing (RLS) data sets. Interested? Here’s how:

In z/OS V2R2 (or z/OS V2R13 and later with the PTF for APAR OA42435), issue the following AMS command:

```
SHCDS LISTSTAT ('<cluster name>')
```

The essential file statistics such as totals for records, inserts, deletes, updates, retrieves, CA/CI splits, extents information, free space, index levels, and high-used/allocated RBA are immediately returned to you.

Enjoy!


A n age-old problem with batch jobs has always been how to run them faster. Many solutions have been created over the years. This included concepts such as piping where the output of one job is input to another job to reduce I/O and breaking a single multistep job into multiple single-step jobs that can run in parallel. Sometimes these parallel jobs have a set of dependencies that you must resolve before or after executing the job.

Products exist that support scheduling jobs with scripts, and so on, which control when jobs are submitted and executed. These products require an effort separate from the job description (JCL) itself to program the control script. Job execution control (JEC) does not replace those products. Instead, JES2 is providing controls that can facilitate breaking down jobs into their constituent pieces. That is, taking a multistep job and breaking it into multiple independent, but related, jobs. These jobs can then be submitted such that JES2 executes them in the correct order, and also allows simultaneous execution of two or more jobs on the same JES2 member.

New JCL statements used to manage job groups
JEC provides native support, by using JCL, for a job scheduling scheme within JES2. The intent is to support the interstep relationships when a multistep job is decomposed into a group of single (or few) step jobs. The governing JEC object is a job group that is created by using a JOBGROUP JCL statement. Ten new JCL statements comprise the JES2 JEC:

- **JOBGROUP** creates a job group.
- **ENDGROUP** denotes the end of a job group definition, and causes that definition to be written to the checkpoint.
- **GJOB** defines a job within a group.
- **JOBSET** defines and references a set of jobs.
- **SJOB** defines a single job within a JOBSET (which can include several jobs).
- **ENDSET** denotes end of a JOBSET.
- **BEFORE** defines jobs or job sets that the current job must run before.
- **AFTER** defines jobs or job sets that the current job must run after.
- **CONCURRENT** defines jobs or job sets that must execute simultaneously on the same JES2 multi-access spool (MAS) member.
- **SCHEDULE** associates a job with a job group.

**JOBGROUP** entity
JOBGROUP is a new entity that is stored in the JES2 checkpoint that describes jobs within the group and the various relationships between those jobs. JOBGROUP only defines the structure of the dependency relationships. When it is initially instantiated, JOBGROUP does not reference any jobs. The definition is static. You cannot add jobs or change dependencies after you create the JOBGROUP definition.

This new feature also introduces the concept of a logging job. A logging job is a JES2 job that acts as the “front end” for the JOBGROUP entity. A logging job has these characteristics:

- Has two data sets:
  - The JESJCLIN data set includes statements that are used to define the job group.
  - The JESMSGGLG data set contains event information for JOBGROUP and all the jobs that belong to it. For instance, a message is logged when JOBGROUP state changes, when each job starts, completes, or is skipped.
- Is used as a front end for commands that act on the group (for example, hold, cancel, and purge).
- Enables filtering performed by the extended status and job modify SSIs.

After JOBGROUP is instantiated, jobs can then be bound, or registered to it by using a new JCL SCHEDULE statement. Any JES2 batch job can be registered to JOBGROUP. Job groups have these types of dependencies:

- Before
- After
- Concurrent.
The following sample job group illustrates these concepts.

```bash
//*---------------------------
//*  DEPENDENCY NETWORK:
//*                    JOBC
//*                     |
//*                JOBE===JOBA
//*                     |
//*                    JOBX
//*---------------------------
//MYGROUP  JOBGROUP
//*JOBA     GJOB
//*       CONCURRENT NAME=JOBA
//*JOBC     GJOB
//  BEFORE NAME=JOBA
//*JOBE     GJOB
//  AFTER  NAME=JOBA
//*JOBX     GJOB
//MYGROUP  ENDGROUP
```

When you submit this, JES2 will instantiate JOBGROUP MYGROUP into the JES2 checkpoint. A MYGROUP logging job will also be created. No jobs are registered to the group yet. When jobs are registered, the following processing takes place:

1. JOBC will run first because it includes the BEFORE NAME=JOBA statement
2. After JOBC finishes running, JOBE/JOBA will run simultaneously on the same JES2 member because of the CONCURRENT statement that ties JOBE and JOBA together
3. After JOBE/JOBA finishes running, JOBX will run because of the JOBX statement that specifies AFTER JOBA
4. After JOBX finishes running, the job group has a status of COMPLETE. JOBGROUP will continue to exist until purged, or all registered jobs are purged at which time the job group is purged
5. The logging job data set (JESMSGLG) records key job group transitions.

The SCHEDULE statement is used to associate jobs with the job group. After the JCL for a job with a SCHEDULE statement completes input processing, the job is registered to JOBGROUP. You can submit jobs that are defined as part of a job group in any order but you must submit the jobs after the JCL of JOBGROUP is processed and JOBGROUP definition is committed to the JES2 checkpoint. For example, consider the following JCL. The jobs that are defined in job group MYGROUP (from the preceding sample) are submitted in an order that is different from the expected execution order:

```bash
//*===JOBA===================================
//JOBA      JOB TIME=NOLIMIT,REGION=0K,MSGCLASS=A,CLASS=B
//          SCHEDULE JOBGROUP=MYGROUP
//STEP1     EXEC PGM=IEFBR14
//*===JOBC===================================
//JOBC      JOB TIME=NOLIMIT,REGION=0K,MSGCLASS=A,CLASS=A
//          SCHEDULE JOBGROUP=MYGROUP
//STEP1     EXEC PGM=IEFBR14
//*===JOBE===================================
//JOBE      JOB TIME=NOLIMIT,REGION=0K,MSGCLASS=A,CLASS=B
//          SCHEDULE JOBGROUP=MYGROUP
//STEP1     EXEC PGM=IEFBR14
//*===JOBX===================================
//JOBX      JOB TIME=NOLIMIT,REGION=0K,MSGCLASS=A,CLASS=A
//          SCHEDULE JOBGROUP=MYGROUP
//STEP1     EXEC PGM=IEFBR14
```

JES2 is providing controls that can facilitate breaking down jobs into their constituent pieces.
In the preceding example, the concurrent set did not run until JOBC ended.

After JOBGROUP is instantiated, you can run various commands on the group, for example:

- $CG'MYGROUP': cancel job group MYGROUP
- $PG'MYGROUP': purge job group MYGROUP (if completed)
- $HG'MYGROUP': hold job group MYGROUP
- $AG'MYGROUP': release job group MYGROUP
- $TG'MYGROUP': modify job group MYGROUP's attributes
- $DG'MYGROUP',SUMMARY: display a complete summary of job group MYGROUP
- $DG'MYGROUP',JOBS: display only job information.

The logging job (JESMSGLG) is the central place where all job group and constituent job transitions are recorded. These transitions include job registration, execution, skipped, and return codes. The preceding sample shows the complete JESMSGLG for MYGROUP.

The preceding job log contains the following information:

- The job identifier for a logging job starts with a “G” (for example, G0000062). You can use this job ID in commands just like batch job IDs. For example, the command $DG62 would display the status of job group MYGROUP. You can also use the job group name in commands such as $DG'MYGROUP'.
- A job is recognized by the job group when it is registered.
- Two CONCURRENT statements might have an intersecting job. If so, jobs in both statements must run together, and the combined set of jobs is referred to as a concurrent set. Jobs in the concurrent set will run simultaneously on the same JES2 member after JOBC runs. In the preceding example, JOBE and JOBA are in a concurrent set.
- Message $HASP1201 is issued when a concurrent set (JOBE and JOBA) enter execution. WLM extended batch initiator support to assure the set is only run on a member where capacity is available.
The following sample shows the messages that are issued by the commands:

```
$DG'MYGROUP',JOBS
G0000262 $HASP890 JOB(MYGROUP)
$HASP890 JOB(MYGROUP) JOB GROUP JOB LIST
$HASP890 JOB NAME JOBID JOB STAT COMP STAT
$HASP890 JOBY JOB00269 Q=HARDCPY COMPLETE
$HASP890 JOBX JOB00268 Q=HARDCPY COMPLETE
$HASP890 JOBC JOB00265 Q=HARDCPY COMPLETE
$HASP890 JOBE JOB00267 Q=HARDCPY COMPLETE
$HASP890 JOBB JOB00264 Q=HARDCPY COMPLETE
$HASP890 JOBA JOB00263 Q=HARDCPY COMPLETE
$HASP890 JOBD JOB00266 Q=HARDCPY COMPLETE
```

Messages are issued by the commands to indicate which actions are taken on what objects (job or job group). The $DG command now includes various keywords to provide a robust view of job group information. For individual jobs, the $DJ command has added extensions to indicate whether a job is associated with a job group, has any BEFORE/AFTER dependencies with other jobs in the group, or has delays that are associated with the job group or concurrent set.

---

**Final words**

We barely scratched the surface on what JEC enables for defining job flow. The new JCL statements provide a base that you can use to model extremely complex job relationships. The job log data set for the logging job shows step-by-step information on the execution flow of the jobs in the job group. The commands provide the control and monitoring functions that you need to maintain a smoothly running job group.

Stay tuned for more information on JEC.

---

**Introducing IBM Spectrum Scale for Linux on z Systems**

IBM Spectrum Scale for Linux on z Systems 4.1.1 is a proven, scalable, high-performance data and file management solution. Based on IBM General Parallel File System (GPFS™) technology, IBM Spectrum Scale for Linux on z Systems is a powerful, clustered file system for big data, analytics, and cloud environments. With IBM Spectrum Scale for Linux on z Systems, you can manage files through Red Hat Enterprise Linux or SUSE Linux Enterprise Server (SLES) instances. The Linux on z Systems instances can run:

- In logical partition (LPAR) mode or on z/VM as guest operating systems
- On the same or different z Systems.

IBM Spectrum Scale for Linux on z Systems supports two different models of cluster configuration: the shared-disk model and the network shared disk (NSD) client/server model.

Ready to learn more? Download the Getting Started white paper today:
As processor speed increases, ready instructions run faster, but the speed to resolve cache misses isn’t keeping up at a proportional pace. Generally, as the clock frequency increases, a cache miss results in more unproductive processor cycles. Beginning with IBM System z10™, HiperDispatch steers work units to processors that share common cache, which minimizes the duration to resolve cache misses. Even with HiperDispatch, cache misses can still occur, and the processor is unproductive while it is resolving cache misses (see CPU 0 in Figure 1).

The IBM z13 supports simultaneous multithreading (SMT) to increase processor (core in SMT terminology) throughput amid the cache misses that a workload experiences. A core supports multiple threads (instruction streams) that compete to run ready instructions on the core’s shared execution resources. The z13 supports SMT-2 (two threads per core). When only one thread has a ready instruction (and the other thread is resolving a miss or is waiting), it wins the competition and runs its instruction. When two threads each have a ready instruction, the core hardware runs both instructions in parallel if possible. Otherwise, it arbitrates which thread’s ready instruction gets to run and defers the other thread’s ready instruction to compete again next cycle (see thread 0 and thread 1 in Figure 1).

SMT tends to:

- Provide faster access to execution resources (SMT-2 can double the number of threads)
- Increase core instruction throughput (more likely to have a ready instruction to run)
- Result in each thread running slower (for sharing the core resource).

The core instruction throughput gain and thread speed slowdown depend on the workload. In general, as the frequency and duration of cache misses increases, there is more opportunity for SMT-2 to improve throughput. Online Transaction Processing workloads tend to have these characteristics at varying degrees and tend to experience the most benefit from SMT-2. Computational workloads that rarely take cache misses tend to receive smaller throughput gains from SMT-2.
The response to the DISPLAY MATRIX=CORE command is more informative than the DISPLAY MATRIX=CPU command response. For example, here’s the sample output for DISPLAY MATRIX=CORE:

```
IEE174I 22.34.15 DISPLAY M 226
CORE STATUS: HD=Y MT=2 MODE: CP=1 zIIP=2
ID ST ID RANGE VP ISCM CPU THREAD STATUS
0000 + 0000-0001 H FC00 +N
0001 + 0002-0003 M FC00 +N
0002 + 0004-0005 M FC00 +N
0003 + 0006-0007 LP FC00 +N
0004 + 0008-0009 L FC00 +N
0005 +I 000A-000B M 0200 ++
0006 +I 000C-000D M 0200 ++
0007 +I 000E-000F LP 0200 ++
0008 +I 0010-0011 L 0200 ++
```

Get the message

When compared to the DISPLAY MATRIX=CPU command output, the DISPLAY MATRIX=CORE command adds the following information:

- HiperDispatch state: HD=Y
- Maximum MT mode that is supported by the hardware and the IPL of z/OS: MT=2
- MT mode by core type: CP=1, zIIP=2
- Core polarity: H=High, M=Medium, L=Low
- State of core: Park (P) or Unpark (blank)
- Core interrupt subclass mask
- Status of each thread: + online, - offline, N offline and expected offline due to MT Mode.

Automation products that use the commands must be aware of the effects in the PROCVIEW CPU and PROCVIEW CORE statements and make updates as needed.
Capacity and accounting considerations
Before you enable SMT-2 in a production environment, it’s important to understand the capacity, accounting, and WLM Policy implications.

The z13 and z/OS deliver meaningful runtime capacity metrics relative to an MT Mode=1 core capacity that includes the following:

• MT Mode=2 core capacity when two threads are running
• MT Mode=2 core capacity when one thread is running (which is identical to MT Mode=1 core capacity)
• MT Mode=2 core capacity free when one thread is running.

As shown in Figure 2, the capacity information allows z/OS to calculate the MT capacity gain between MT Mode=2 and MT Mode=1 at run time for any workload.

| Capacity        | MT Mode = 2   | MT Mode = 2   | MT Mode = 1
|-----------------|---------------|---------------|---------------
| In use          | Core executing | Core executing | Core         |
| Free            | 2 threads     | 1 thread      |               |

Figure 2. Core capacity by MT mode and running threads

z/OS system programmers expect the same job to use a similar amount of capacity each time it runs regardless of the MT Mode. On the z13, z/OS is designed to charge each job according to the amount of single-thread (MT Mode=1) equivalent capacity use to provide consistent accounting in different MT modes. This design extends throughout z/OS, such that all accounting metrics are presented in terms of single-thread (MT Mode=1) equivalent capacity use.

WLM considerations
MT Mode=2 can positively or negatively affect the rate that work moves through your system. Velocity goals tend to be more sensitive with MT Mode=2 because the number of processors (thread) that are using the samples doubles. After the transition to MT Mode=2, monitor response times and WLM goals (especially velocity goals), and adjust the goals as necessary.

If you use the Workload Manager (WLM) Sysplex routing capabilities through Sysplex Distributor or DB2 for z/OS Distributed Data Facility (DDF), you might observe that the zIIP capacity gain by MT Mode=2 is reflected in the workload distribution. Naturally, capacity gain primarily affects systems that have a more significant portion of zIIP capacity.

The MT Mode=2 core instruction throughput gain and thread speed loss are workload-dependent. The IBM Lab observed the following results when it compared benchmarks that used MT Mode=1 against MT Mode=2:

• Core capacity gain between +10% and +40%
• Thread speed loss between -30% and -40%.

MT Mode=2 capacity and accounting information are accurate between 20%-90% zIIP usage because z/OS receives a sufficient workload representative sample, with one running thread and with two. When the zIIP usage is outside these bounds, z/OS can receive an insufficient or non-workload representative sample and produce less accurate results.

Transaction response time includes execution queue delay and actual execution time. MT Mode=2 tends to result in:

• Lower execution queue delay by doubling the number of thread execution locations.
• Longer execution time for threads that are sharing core execution resources. When each z13 thread operates 30% - 40% slower than an MT Mode=1 z13 core, each thread runs at a speed similar to a z196 CPU.

Depending upon the workload, MT Mode=2 can have a positive, neutral, or negative affect on response time. On the whole, we expect you to be in for a pleasant surprise when you enable SMT!

Enabling service
The enabling service that is required for z/OS V2R1 includes the following PTF (and APAR) numbers:

• UA90753 (OA43366) for BCP
• UA90762 (OA43622) for SRM / WLM
• UA76154 (OA44439) for XES / XCF
• Optionally UA76026 (OA44101) for Resource Management Facility requires and UA90772 (OA44624) for z/OS UNIX System Services.

See the IBM Redbooks publication z Systems Simultaneous Multithreading Revolution at: redbooks.ibm.com/abstracts/redp5144.html.
Transform your data into a PDF file to share across the cloud

BY ANTHONY MINGO, PAT GLENSKI, AND TARIQ CHOUDHRY

One of the common ways to exchange information today is by using a PDF file. The PDF format is extremely portable and, the receiver will be able to see what you intended regardless of whether they view it on a workstation or on a mobile device. A PDF file also supports protection and encryption. This article provides an overview about how to create, encrypt, protect, and email a PDF file using your system.

The Infoprint Server optional feature of IBM z/OS and the IBM Print Transform from AFP for Infoprint Server AFP to PDF for z/OS products enable this support.

Together, these tools provide the functions that are needed to:

- Process input data from diverse environments, such as Microsoft Windows, IBM AIX®, or Linux platforms.
- Automatically detect input data types and convert the data from line mode or AFP format to a PDF file by using the AFPXPDF transform function.
- Convert AFP documents or simple line data (including CICS and IMS transactions) into a PDF file for easy availability across diverse platforms, including a cloud.
- Allow the PDF output to be saved either in a z/OS UNIX file or an MVS data set.
- Create a PDF as a protected document with encryption and password protection.
- Allow these documents to be sent as email attachments (using Infoprint Server IP PrintWay™ Extended mode).
- Optimize options to help reduce PDF document size, if needed.

Scenarios for using the transforms

Two common scenarios below help show you how you can use Infoprint Server with the AFPXPDF transform interface.

Scenario 1: Converting data from JES spool, encrypting it, and emailing to a user.

1. An application places data on a JES spool.
2. The application JCL directs the output to Infoprint Server to be sent as an email.
3. Infoprint Server automatically determines that the data needs to be transformed to a PDF file and invokes the AFPXPDF transform function.
4. The AFPXPDF transform process determines that the PDF document must be encrypted with a password and invokes a user-provided password exit to obtain the user and owner password for encryption. It encrypts the input by using a standard encryption algorithm to produce encrypted PDF output.
5. Infoprint Server sends an email and includes the encrypted and password-protected PDF file as an attachment. The encryption helps prevent data snooping while transmitting the PDF file over open TCP/IP networks, and the recipient can open the PDF attachment by using their pre-established password.

Scenario 2: Converting data, encrypting it, and storing for later access.

1. You invoke AFPXPDF transform processing directly by using the Infoprint Server AOPBATCH program.
2. You can convert the input to a password-protected and encrypted PDF output by using the AFPXPDF transform, which you can save as a z/OS UNIX file or an MVS data set for later use.

When you save the PDF output file, you can make such files available in an environment of your choice so they are sharable with a diverse user community like mobile users. This is especially useful when users need to share a large number of PDF files or very large PDF files with either an individual or a group in a protected manner.
Example JCL to call the AFPXPDF transform function:

```
//TRANSFRM EXEC PGM=AOPBATCH,
// PARAM="/afpxpdf -cSecPDF -j attr=/DD:AT -o//
// DD:OUT //DD:IN"
//IN DD DSN=input_transaction_data,DISP=SHR
//OUT DD DSN=output_PDF_file,DISP=(NEW,CATLG,DELETE),
//  DCB=(RECFM=VB,LRECL=1024,BLKSIZE=1028),SPACE=(CYL,(1,1),RLSE)
//ATTRS DD *
pdf-owner-identifier='Nurse-Lee'
pdf-user-identifier='Dr-Smith'
/*
//STDOUT DD SYSOUT=* 
//STDERR DD SYSOUT=* 
//STDDO DD SYSOUT=* 
```

Example partial transform configuration class:

```
transform afpxpdf_SECPDF
    start-command = afpxpdfd
    environment={AOP_PASSWORD_EXIT -> "/usr/lpp/Printsrv/lib/aokpdfexit.dll/
                /usr/lpp/Printsrv/samples/aokpdfexit.db"}
AOP_ENCRYPT -> yes
AOP_PROTECT -> "copy print"
... 
```

If you want to view the security values of a PDF document, select File, Document Properties, and Security.

**Protected PDF documents options**

PDF documents protected with encryption can be produced with or without a password. You might enable encryption without a password if you are sending a PDF document to a user without needing passwords. By using this option, the PDF document is encrypted so that packet sniffers do not see your data in the clear, yet it does not require the user to have a password to view the PDF document. You can implement encryption without a password by using the AOP_ENCRYPT variable in the transform configuration file.

Encryption with a password requires a password to open the PDF document and you can also limit available actions. By limiting the available actions a user can perform, you can protect the data further. For example, you can set the PDF to not allow copying or printing of the data.

If a user tries to open a PDF that has been protected with a password, they are prompted to provide the password before they can see the contents of the document or perform allowed actions.

To produce PDF documents with a password, you must set up a password exit and a password database that contains the passwords for user and owner identifiers. In addition, you must provide a transform configuration class, which contains the settings that you want to use. A sample password exit code and sample password database are included with the transform function and a user guide provides details for further customization.

**Optimizing PDF documents**

For some people, managing the size of PDF documents might be important. The AFPXPDF transform function provides optimization options that can help you reduce the PDF file size, which sometimes results in significantly smaller files. The transform process, by default, performs the standard flate compression function to compress text, graphics, and images that are included in the PDF output. You can control the function by using the AOP_FLATE configuration option. Two common configuration options are image optimization and font optimization.

**Image optimization**

When you process PDF documents that include images, you can manage the size of images that are included in the PDF document by using additional configuration options. For JPEG images, the output can be scaled by using the AOP_JPEGQUALITY environment variable. For raster images (raster images are resolution dependent), use the AOP_MAXIMAGERESOLUTION environment variable. For example, you can set AOP_JPEGQUALITY -> 10, which scales the JPEG images to 10% of the original quality (a 90% reduction in image quality). For raster images, if you specify AOP_MAXIMAGERESOLUTION -> 300, the image resolution is reduced to 300DPI.

For most cases, these options yield a viewable image with a reduction in PDF file size.

Create, encrypt, protect, and email a PDF file using your system.
Font optimization

There are two options available to tune the amount and type of font information that is included in the PDF document.

With the first option, AOP_FONTMAP_TABLE, you can convert raster fonts to corresponding outline fonts. Outline fonts typically provide a superior viewing experience.

With the second option, AOP_OUTLINES, you can specify how outline font information is included in the PDF documents. With the default settings, full font information for outline fonts is embedded in the PDF document for best document fidelity. In some cases, this can result in significant font information being embedded into the PDF document. This depends on several factors, such as number of fonts, document size, and font types in use. This can result in a large PDF document. If you use standard Adobe Base 14 outline fonts, and you choose the value base14 for this option, it can result in significant file size reduction. This is achieved by including the font information by name only for the base14 outline fonts, without a loss in document fidelity.

For example, a sample font-mapping table maps the common raster character set to the corresponding outline character set:

\[\text{AOP_FONTMAP_TABLE} \rightarrow \text{"/usr/lpp/Printsrv/samples/aokfontmap.samp"}\]

This option specifies that standard Adobe Base 14 outline fonts are to be included by name, and to embed any other outline fonts:

\[\text{AOP_OUTLINES} \rightarrow \text{base14}\]

Combined, these options provide security and optimization capabilities to convert line data or AFP input and either email it as a PDF attachment or share it by using the platform of choice knowing that the data is secure and protected.

For additional information about the AFPXPDF transform, see the print transforms documentation in *AFP in IBM Print Transform from AFP for Infoprint Server for z/OS*, G325-2634. For additional information about how to use the Infoprint Server feature, see the *z/OS Infoprint Server User’s Guide*, SA38-0695.
Karla Arndt is a Senior Software Engineer and the team leader for Predictive Failure Analysis and Runtime Diagnostics. She has published several articles and holds several patents in the areas of predictive technologies, autonomic computing, and referential integrity.

Scott Ballentine is a software engineer with the z/OS Device Allocation team in IBM Poughkeepsie. He has been part of the z/OS team since joining IBM in 1997 with a B.S. in Computer Science from Carnegie Mellon University.

Iris Baron is the team lead for IBM’s Java JIT compiler on z Systems in Toronto. She has more than 10 years experience in the IBM JIT compiler area focusing on compiler optimization and Java performance enhancements.

Rita Beisel is a technical writer at IBM. She is currently writing for Integrated Cryptographic Services Facility (ICSF), Trusted Key Entry (TKE), Encryption Facility for z/OS, and Cryptographic Services System Secure Sockets Layer Programming (SSL). She has written for RACF, z/VM, DB2, DFSMS, and MVS in the past. She also likes to do interface design, usability testing, and surveys.

Snehal Bhakta is a software engineer for DFSMS/hsm development in the IBM Systems and Technology Group.

Ritu Bhargava is an Advisory Software Engineer at IBM Poughkeepsie, who works in the Binder / Language Environment Level 2 Support team. Ritu has been with IBM for fourteen years and has a Masters in Computer Science from City University of New York.

Heather Bosko is a Software Engineer at IBM in Poughkeepsie, New York. Heather has over 16 years of experience in z/OS including the areas of GRS, System REXX and SMF.

Stephen Branch (aka Mr. Catalog) is an IBM Senior Software Engineer at Almaden Research Center whose 32 year career has dealt with all aspects of DFSMS. Stephen specializes in ICF Catalog, IDCAMS, and VSAM. He holds several patents for these components and 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.

Ronald Bretschneider is a Senior Software Engineer for DFSMS at IBM San Jose, CA. Ronald has 31 years experience in DFSMS and holds a B.S. degree in Mathematics and an MS degree in Computer Science from San Jose State University.

Chris Brooker is a Senior Software Engineer with over 14 years experience on the mainframe and z/OS. He is currently the Development Team Lead for IBM zAware and is an expert in Global Resource Serialization.

Toshiba Burns-Johnson is a User Experience Professional at IBM. She has extensive experience in the areas of information development, user experience design, usability, and accessibility. She helps bridge the communication gap that exists between users and developers, and advocates for design changes that improve the user experience.

James Cammarata is a software engineer for DFSMS/hsm development in the IBM Systems and Technology Group. Jim has held various DFSMS development positions over the 31 years he has been at IBM.

Lisa Case is an advanced software engineer with over 30 years experience at IBM. During her vast career she has worked on a variety of different platforms and has held a many different positions. Lisa is currently a member of the IBM zAware team.

Pierre Cassier is a senior certified IT specialist, based in Montpellier, France, working for the GDPS test team. He has worked with z Systems for 40 years, as system programmer for a large customer, and then for IBM as z/OS performance expert. He has worked with GDPS Active/Active since the inception of the product.

Eric Chabert is a senior certified IT specialist, based in Montpellier, France, working for the GDPS/Active-Active Test team. He has worked with z Systems for over 25 years, with a variety of products, but principally with CICS, DB2, Problem Determination Tools and with GDPS Active/Active.

Colin D. Chen is a Staff Software Engineer at IBM Poughkeepsie. His expertise includes development of several z/OS components with a focus on IOS. He holds a B.S. in Computer Science from Pennsylvania State University and has been with IBM since 2007.

Wai Choi, CISSP®, is a Senior Software Engineer for IBM in Poughkeepsie. She works on digital certificate support in RACF, PKI services and Kerberos. Wai actively participates in conferences and forum on certificate-related topics.

Tariq Choudhry is an Advisory Software Engineer in z/OS Print organization. He has been most recently involved in Infoprint and transform products for Infoprint Server for z/OS and development work and has worked on JES3 test and development prior to that. Tariq has been with IBM for 23 years.

Scott Compton is a Senior Technical Staff Member currently working in the I/O Supervisor area. He joined IBM in 1986 after receiving his B.S Degree in Computer and Information Science from The Ohio State University.

Ross Cooper is a Software Engineer for the RACF, Kerberos and PKI products at the IBM Poughkeepsie lab.

Troy Crutcher is the Academic Initiative program manager.

Greg Daynes is a Senior Technical Staff Member and the z/OS Installation Deployment Architect in IBM at Poughkeepsie, New York. His current responsibilities include z/OS software acquisition, installation, migration, deployment, and maintenance strategies. Greg is a frequent speaker at SHARE and the IBM z Systems Technical University (formerly zExpo).

Elton De Souza is a Software Developer at the IBM Toronto lab working on the Just-In-Time compiler technology for Java on z. His focus is on Cloud and Mobile workload acceleration on z Systems.

Eduard Diehl has been working at IBM for over 20 years. He started out in mainframe performance analysis focusing initially on XRC, but has since moved to z/OS System Data Mover development spending most of his career in XRC.

Shannon Gallaher is a software engineer for DFSMS/hsm development in the IBM Systems and Technology Group.

Bob Gensler is a security software developer for z/OS RACF and PKI Services. His 28 years of experience include: operating system and software infrastructure development for AIX and the pSeries HPC platforms.

Joseph Gentile is a Software Engineer in IBM z/OS Core Development. He is currently the GRS lead and also specialized in System Logger. Joe holds a B.S. in Computer Science from Binghamton University and is pursuing his M.S. in Computer Science at SUNY New Paltz, specializing in machine learning.

Pat Glenski is a Software Engineer in z/OS Development at IBM Rochester, MN. He has worked for IBM for over 20 years in different areas. He has several patents, and has presented at several conferences for IBM.

Clark Goodrich is a Senior Software Engineer for z Systems Performance Organization at IBM Systems & Technology Group in Poughkeepsie, New York. Clark's focus is performance analysis of Java on z Systems.
Lisa Gundy is a Senior Software Engineer in the z/OS DFSMS development organization. She joined IBM in 1982 and is currently the lead architect and strategist for z/OS DFSMS Disk Copy Services. Lisa is also the DFSMS System Data Mover component lead architect and development team leader.

Tariq Hanif is a Senior Software Engineer in the z/OS System Verification Test group where he specializes in z/OS HyperSwap, Parallel Sysplex, Disaster Recovery and High Availability testing for the z/OS operating system.

Steven R. Hart, CISSP®, works on design, development and strategy for z/OS Cryptographic Services at IBM Poughkeepsie, New York. In his twelve years with IBM, he has made significant contributions to Integrated Cryptographic Services Facility (ICSF), Trusted Key Entry (TKE), Encryption Facility for z/OS, Tivoli® Key Lifecycle Manager (TKLM) for z/OS, and Cryptographic Coprocessor firmware.

David Herr is a Senior Software Engineer in IBM Systems Unit Enterprise Networking Solutions development team. He is the performance analyst and a developer for z/OS Communications Server, focusing on TCP/IP technologies, High Availability functions and zEnterprise networking.

Zhen Hua Dong is an Advisory Software Engineer, comes from z Systems Brand Team, IBM. He joined IBM in 2006. He is working as the role of technical and technology enablement and his working focus now is z/OSMF, z Systems HA/DR and Open Cryptographic infrastructure.

William (Craig) Johnston, CISSP®, is a Senior Managing Consultant for STG Lab Services at IBM Poughkeepsie, New York. His expertise includes RACF, ICSF, and other z/OS security components. He is recognized industry-wide as a subject matter expert on tape and disk encryption. He has been with IBM since 1982.

Nicholas Jones is a Software Engineer for the BCP at IBM Poughkeepsie for the last 12 years. His work includes design, test and service for Service Aids including IPCS, PDUU, and Logrec.

Steven Jones is a lead software engineer for the z/OS Job Management team, which includes the Allocation, Job Scheduler, and SMF components. Steve joined IBM in 1987 and has a Masters degree in computer science from RPI.

Mike Kasper is an Advisory Software Engineer for the z/OS System Integrity Competency Center at IBM Poughkeepsie. Previously, he worked in TSO/E Development. He holds a Bachelors degree in Computer Engineering from the University of Illinois at Urbana-Champaign and has been with IBM for 15 years.

Gus Kassimis is a Technical Staff Member with IBM Systems Unit Enterprise Networking Solutions architecture, strategy and design team. He is a designer for the z/OS Communications Server, focusing on TCP/IP technologies, High Availability functions and zEnterprise networking.

Kevin Kathmann is an Advisory Software Engineer. Kevin has been working in JES2 as a developer since 2008. Before joining JES2, Kevin’s 20 years with IBM have included various positions in IBM System i® database.

Kevin McKenzie is a Software Test Engineer for z/OS. He was the test architect for SMT-2 and HiperDispatch. He is a Software Engineer for z/OS serving as a lead z/OS Supervisor programmer with expertise in HiperDispatch and SMT-2.

Anthony Mingo is an Advisory Software Engineer in z/OS Print at Rochester, MN. His expertise includes Print Transforms from AFP for Infoprint Server for z/OS and Infoprint Server development. Anthony has been with IBM for 17 years.

Mark Nelson, CISSP®, CSSLP**, has spent the last 27+ years of his IBM career with the RACF Design and Development Team. He is the author of such popular RACF utilities as the RACF Database and SMF Unload Utilities and numerous trade journal articles. Mark is an award-winning speaker at many conferences and the co-author of the book “Mainframe Security for the Security Expert: An Introduction to RACF”.

Barbara Neumann is an Advisory Software Engineer and a Managing Editor of z/OS Hot Topics. Barbara is an Information Planner and Developer for z/OS MVS in Poughkeepsie, NY. She holds a BA in mathematics from Vassar College and an MBA from the University of Rochester.

Mike Onghena is a Senior Software Engineer in the RACF design and development group at IBM.

Bonnie Ordonez is a software engineer specializing in the z/OS Job Scheduler and SMF components. Bonnie joined IBM in Poughkeepsie, NY in 1998. She has a B.S degree in Computer Science from Florida International University.

John Paveza is a Senior Technical Staff Member in z/OS. His 34 years with IBM includes Storage Subsystem development as well as DFSMS Software.

Rolando Perez is a Software Engineer and UI Designer in the IBM Systems Group. He is currently the lead UI Designer for z/OSMF and IBM zAware and developer of the CFSizer web application. His areas of interest include design, prototyping, mobile and web development. He has multiple patent applications in the area of UI design and mobile applications. He is a graduate of Rensselaer Polytechnic Institute.

Kathy Pfeiffer is the z Systems Hardware and Software Information Solutions Manager.

Daniel Rosa is a Senior Software Engineer based at IBM in Poughkeepsie, New York. He contributes to design and development in the Supervisor component of z/OS. His expertise includes HiperDispatch and Simultaneous Multithreading.

Eric Rosenfeld, CISSP®, joined IBM as a software engineer in 1985 and has been developing security products for 22 years. He has worked in RACF Development and is currently a designer and developer for TSO/E.

David Shackelford has been with IBM 23 years, working in design and architecture of copy services on z/OS and distributed systems. He is a Master Inventor and is currently solution architect for XRC and software based copy services.

John C. Shebey III is a Senior Software Engineer at IBM Poughkeepsie, where he leads the z/OS UNIX Level 2 Support team. John has been with IBM for seventeen years and is a frequent speaker at SHARE. He holds a BS degree in Mathematics from Virginia Tech.

Susan Shumway is a technical information developer at IBM Poughkeepsie. She holds degrees in engineering and technical communication from Rensselaer Polytechnic Institute. She currently writes for MVS and CIM and is a lead planner for the z/OS information development team.

Horst Sinram is a Senior Technical Staff Member in the IBM Germany Research & Development lab in Boeblingen, Germany. His responsibilities include z/OS Workload Management and Capacity Management. Horst is a frequent speaker at major conferences and also works frequently with clients world-wide.

Corey Smeaton is a software engineer and the lead developer for FlashCopy® and PPRC items in DFSMS System Data Mover (SDM). He joined IBM in 2005 and holds a degree in Computer Engineering from The University of Arizona.

Anthony Sofia is a Senior Software Engineer at IBM Poughkeepsie. His expertise includes zEnterprise Data Compression and the z/OS System Management Facilities. He has been with IBM for 10 years and enjoys spending his free time with his family.

Mike Spiegel is a Senior Software Engineer for the z/OS System Integrity Competency Center at IBM Poughkeepsie. Previously, Mike worked in z/OS UNIX Kernel and File System Development. He was one of the original designers of z/OS UNIX and also worked in TSO/E Development.

Warren Stanley is a Senior Technical Staff Member and Master Inventor in the IBM DS8000 Copy Services microcode development organization. He is the microcode architect for Multiple Target PPRC, in addition to being a Copy Services architect and developer for PPRC replication and IBM z/OS Global Mirror (XRC). Warren joined IBM in 1982 and holds a degree in Electrical and Computer Engineering from California State Polytechnic University in Pomona, California.

Jerry Stevens is a Senior Technical Staff Member with IBM Systems Unit Enterprise Networking Solutions architecture, strategy and design team, focusing on communications hardware architecture, protocols and related solutions. He has over 30 years of experience with z Systems, IBM operating systems and network communications.
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

Bruce Talbott is an Advisory Software Engineer performing development for JES2 since 2007. Bruce’s 25 years with IBM have included development and Level3 experience on System i in areas such as work management, local and remote workstations, and data communications.

Yunli Tang is a recent new hire and a Software Engineer on the IBM zAware team. She recently graduated from Binghamton University with an MS degree in Computer Science. Her main interests are in machine learning and data analysis.

Carrie Van Noorden is an Advisory Software Engineer in DFSMS at IBM San Jose, California. With IBM for nearly 30 years, Carrie is the team lead for the Device Services team, focusing on DADSM/CVAF development.

Jessie Yu is an Advisory Software Engineer for the z/OS Parallel Sysplex development team in Poughkeepsie, NY. She has worked on a number of features for both XCF and XES during the 8 years she’s been with the team. Prior to joining Parallel Sysplex, she worked for z/OS Integration Test, concentrating on IMS testing.

Steve Wall is a senior certified IT specialist, based in Montpellier, France, working for the GDPS/Active-Active Development team. He has 30 years of experience of working with z Systems, with CICS TS Development, the z/OS Benchmark Center in Montpellier and with GDPS/Active-Active Development.

Laurie Ward, CISSP®, works in Poughkeepsie, NY as a RACF developer and designer. She has worked on many RACF development projects since 1988, including B1 support, RACF Remote Sharing Facility (RRSF), Dynamic CDT and Custom Fields.

Steve Warren is a Senior Software Designer and Developer in IBM Poughkeepsie and serves as the technical lead for both the z/OS BCPI and the z/OS Client Web Enablement Toolkit teams. Steve has been with IBM for 26 years.

Tom Wasik is the team lead in JES2 design and development at IBM Rochester. He’s spent 30 years working on JES2 at IBM.

Bruce Wells, CISSP®, is a senior software engineer for the RACF design and development group, where he has worked for the past 24 years. He has extensive experience in both the z/OS and z/VM versions of the product, and his recent contributions include enhancements to password security and the RACF Remote Sharing function.

Scott Woolley, CISSP®, has been part of RACF development for many years. He is proud to have been part of the dynamic team that recently delivered the enhancement RACF password support. You’re welcome.

* Certified Information Systems Security Professional
** Certified Secure Software Lifecycle Professional

* Certified Information Systems Security Professional

** Certified Secure Software Lifecycle Professional
The z/OS Hot Topics Newsletter is published yearly as a service to customers, providing articles of interest to the z/OS community.

The information in this Newsletter is presented as is and is based on the best knowledge of the authors. No warranty is provided (neither expressed nor implied). IBM may not offer the products, features or services discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM representative for information on the products or services available in your area.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries. IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at ibm.com/legal/copytrade.shtml.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.
No experience necessary.

This year marks the tenth anniversary of the IBM-sponsored Master the Mainframe Championship for students in the U.S. and Canada.

Read more about these amazing, competitive, and driven students on page 28:
Master the Mainframe turns 10