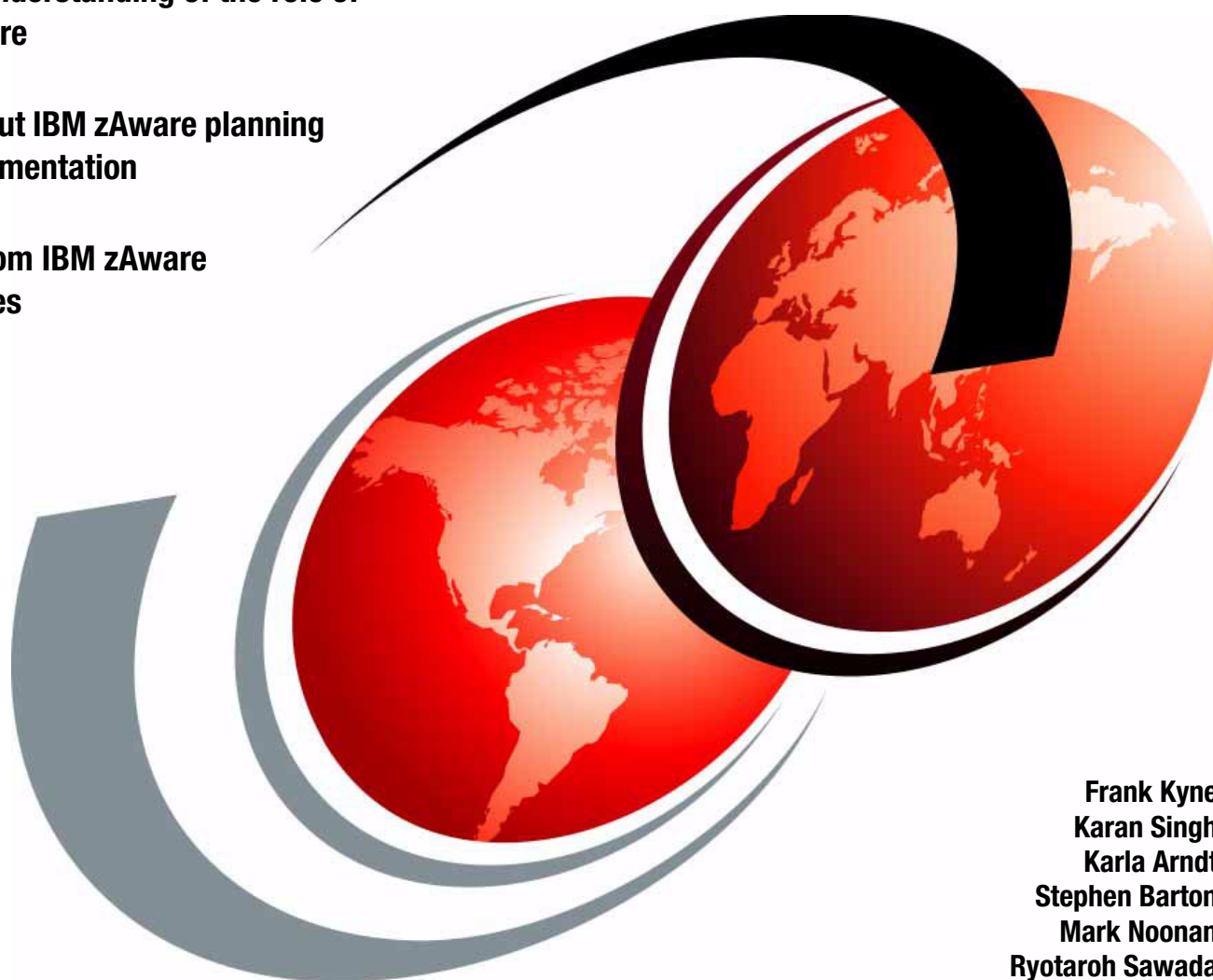


Extending IBM z/OS System Management Functions with IBM zAware

Gain an understanding of the role of
IBM zAware

Learn about IBM zAware planning
and implementation

Benefit from IBM zAware
capabilities



Frank Kyne
Karan Singh
Karla Arndt
Stephen Barton
Mark Noonan
Ryotaro Sawada

Redbooks



International Technical Support Organization

**Extending IBM z/OS System Management Functions
with IBM zAware**

March 2013

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (March 2013)

This edition applies to IBM zEnterprise EC12.

© Copyright International Business Machines Corporation 2013. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	vii
Trademarks	viii
Preface	ix
The team who wrote this book	ix
Now you can become a published author, too!	xi
Comments welcome	xi
Stay connected to IBM Redbooks	xi
Summary of changes	xiii
March 2013, First Edition	xiii
Chapter 1. Introduction to IBM zAware	1
1.1 The role of zAware in maximizing availability	2
1.2 Challenges faced by IT departments	3
1.2.1 Keeping up with changing technologies and skills	5
1.2.2 Keeping up with product changes	6
1.2.3 Ensuring products are configured appropriately	6
1.2.4 Analytics can help address these issues	6
1.3 Overview of IBM zAware	7
1.3.1 Architecture of IBM zAware	11
1.3.2 Transporting messages to IBM zAware application	13
1.3.3 Priming IBM zAware with message data	14
1.3.4 Creating the behavioral model	15
1.3.5 Viewing the results of IBM zAware analysis	17
1.4 Layout of this book	19
Chapter 2. IBM z/OS system management functions	21
2.1 Overview of System z system management functions	23
2.1.1 z/OS component functions	27
2.2 IBM z/OS Management Facility	28
2.2.1 z/OSMF tasks	28
2.2.2 z/OSMF Incident Log	28
2.3 IBM Health Checker for z/OS	30
2.3.1 Getting the most out of IBM Health Checker for z/OS	32
2.4 Runtime Diagnostics	32
2.4.1 Invoking Runtime Diagnostics	34
2.4.2 Component analysis	35
2.4.3 Global resource contention	36
2.4.4 Address space execution	39
2.5 Predictive Failure Analysis	42
2.5.1 Predictive Failure Analysis overview	42
2.5.2 Types of abnormal behavior detected	43
2.5.3 PFA and IBM Health Checker for z/OS integration	45
2.5.4 PFA processing	46
2.5.5 PFA reports	51
2.5.6 PFA checks	51
2.5.7 How PFA groups address spaces for monitoring	60
2.5.8 PFA and Runtime Diagnostics integration	63

2.5.9 Achieving maximum benefit from PFA	65
2.6 IBM System z Advanced Workload Analysis Reporter	71
2.6.1 Preparing IBM zAware for use	71
2.6.2 Using IBM zAware	75
2.6.3 Achieving maximum benefit from IBM zAware	81
2.7 Message analysis with the various functions and products	82
2.7.1 Runtime Diagnostics critical message analysis	82
2.7.2 PFA message arrival rate check	83
2.7.3 IBM zAware message analysis	84
2.7.4 Comparison summary	85
2.8 Additional tips to achieve high availability	88
2.9 Sample scenarios	89
2.9.1 The system is unresponsive	90
2.9.2 A sysplex problem exists in which all LPARs are affected	90
2.9.3 Software changes have been made to your system	90
2.9.4 IBM zAware detects an anomaly	91
2.9.5 PFA message arrival rate check exception issued for a high rate	91
2.9.6 PFA exception issued for a low rate	92
2.9.7 PFA exception issued for a high SMF arrival or high ENQ request rate	92
2.9.8 Runtime Diagnostics message event detected	93
2.9.9 PFA and Runtime Diagnostics examples	94
Chapter 3. Planning for an IBM zAware implementation	95
3.1 Planning overview	96
3.2 Selecting which systems to monitor with IBM zAware	96
3.3 Hardware resources	98
3.3.1 CPU capacity	98
3.3.2 Network connectivity	102
3.3.3 DASD storage	102
3.3.4 z/OS requirements	105
3.4 Technical resources	107
3.4.1 Hardware Management Console	107
3.4.2 z/OS MVS	107
3.4.3 Network	108
3.4.4 Security	109
3.5 Using IBM zAware	110
3.6 Monitoring IBM zAware availability	111
3.7 Accessing the IBM zAware GUI	111
Chapter 4. IBM zAware installation	113
4.1 Overview of the IBM zAware installation process	114
4.2 IBM zAware LPAR	117
4.3 Network	128
4.4 Security considerations	130
4.4.1 RACF definition on the monitored systems	131
4.4.2 User authentication to use the IBM zAware GUI	131
4.4.3 LDAP setup	134
4.5 Prepare System Logger	140
4.6 Connecting to IBM zAware	142
4.7 Bulk Data Load Utility	146
Chapter 5. Maintaining and managing IBM zAware	153
5.1 Managing IBM zAware components	154
5.2 Starting and stopping the IBM zAware application	155

5.2.1 Starting and stopping the IBM zAware Analytics Engine	156
5.2.2 IBM zAware LPAR automation considerations	157
5.2.3 IBM zAware support for dynamic configuration changes	157
5.3 Managing IBM zAware disks	157
5.3.1 User ID authority requirements	158
5.3.2 Adding disks to the IBM zAware configuration	158
5.3.3 Removing disks from the IBM zAware configuration	160
5.3.4 Backing up the IBM zAware file system	161
5.3.5 Sharing disks between IBM zAware LPARs	164
5.3.6 Backup IBM zAware LPARs	165
5.4 Managing connections from monitored clients	165
5.5 Maintaining IBM zAware data	167
5.5.1 Bulk Data Load Utility	167
5.5.2 Creating and updating the system model	168
5.5.3 Moving systems between sysplexes in IBM zAware	174
5.5.4 Data management granularity	175
5.6 Managing IBM zAware firmware	175
5.7 Disaster recovery considerations	178
5.8 Daily and weekly management tasks	178
5.8.1 Daily management tasks	179
5.8.2 Weekly management tasks	179
5.9 Checklist for adding a monitored client	179
5.10 System Logger commands for IBM zAware support	180
5.10.1 IBM zAware messages	182
5.11 Problem determination for IBM zAware	183
5.11.1 Problems during IBM zAware initial setup	183
5.11.2 Confirming connectivity to IBM zAware	183
5.11.3 Checking for problem notifications on the IBM zAware GUI	184
5.11.4 Sending diagnostic information to IBM	185
Chapter 6. Integrating IBM zAware with other IBM products	191
6.1 Introduction	192
6.2 z/OSMF	192
6.3 IBM Tivoli NetView for z/OS	196
6.3.1 Using the ZAI sample programs	198
6.3.2 Using the information retrieved from the IBM zAware API	204
6.4 IBM zAware Application Programming Interface	205
Appendix A. Syslog Message Analysis Program	207
Message Analysis Program	208
Capacity planning	208
Running the MAP program	208
Determining ability to build a model	211
Appendix B. Activating TCP/IP AT-TLS	213
Impact of AT-TLS on TCP/IP startup	214
AT-TLS policy for IBM zAware certificate	214
Adding the IBM zAware certificate to the RACF SITE keyring	215
Creating the client certificate for the system	215
AT-TLS policy file details	215
Appendix C. Using automation to monitor IBM zAware connections	219
Querying the status of connections to IBM zAware	220

Appendix D. Problem determination sample 223

Diagnosing System Logger connection errors 224

Related publications 229

IBM Redbooks 229

Other publications 229

Online resources 230

Help from IBM 230

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

CICS®	NetView®	S/390®
DB2®	OMEGAMON®	System i®
FlashCopy®	Parallel Sysplex®	System z®
GDPS®	RACF®	Tivoli®
HiperSockets™	Redbooks®	WebSphere®
IBM®	Redpaper™	z/OS®
IMS™	Redbooks (logo)  ®	z/VM®
MVS™	RMF™	zEnterprise®

The following terms are trademarks of other companies:

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

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

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

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

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication explains the capabilities of the IBM System z® Advanced Workload Analysis Reporter (IBM zAware), and shows how you can use it as an integral part of your existing System z management tools.

IBM zAware is an integrated, self-learning, analytics solution for IBM z/OS® that helps identify unusual system behavior in near real time. It is designed to help IT personnel improve problem determination so they can restore service quickly and improve overall availability.

The book gives you a conceptual description of the IBM zAware appliance. It will help you to understand how it fits into the family of IBM mainframe system management tools that include Runtime Diagnostics, Predictive Failure Analysis (PFA), IBM Health Checker for z/OS, and z/OS Management Facility (z/OSMF).

You are provided with the information you need to get IBM zAware up and running so you can start to benefit from its capabilities immediately. You will learn how to manage an IBM zAware environment, and see how other products can use the IBM zAware Application Programming Interface to extract information from IBM zAware for their own use. The target audience includes system programmers, system operators, configuration planners, and system automation analysts.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Frank Kyne is an Executive IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on all areas of IBM Parallel Sysplex® and high availability. Before joining the ITSO 14 years ago, Frank worked in IBM Ireland as an MVS™ system programmer.

Karan Singh is a Project Leader at the ITSO Poughkeepsie Center, responsible for producing a variety of System z and z/OS IBM Redbooks publications. A generalist with more than 15 years of experience working with a variety of clients and IBM internal systems, he has acquired expertise in core z/OS technologies and a broad familiarity with System z. Karan holds a Diploma in Computer Programming, a Bachelors Degree in Liberal Arts, and a Masters Degree in Teaching.

Karla Arndt is a Senior Software Engineer located in Rochester, Minnesota, USA. She has nearly 25 years of experience and has worked on several platforms, including components of IBM System i®, z/OS, and the IBM Tivoli® Directory Server. She holds a degree in Applied Mathematics and Computer Science from the University of North Dakota. Karla is the technical team leader for Predictive Failure Analysis and Runtime Diagnostics. She holds several patents in the areas of autonomic computing, predictive technology, and referential integrity.

Stephen Barton is an I/T Specialist in the United States. He has 37 years of experience in the Information Technology field, and has worked at IBM for 24 years. His areas of expertise include Large Systems software (MVS, IBM OS/390®, and z/OS) and storage (DASD, tape,

and DFSMS architecture and products). This was Steve's first opportunity to participate in an ITSO residency and to collaborate on an IBM Redbooks publication.

Mark Noonan is an IT Specialist in Australia. He has many years of experience in Operations and Systems Management on the mainframe. He has worked at IBM for 18 years. His areas of expertise include IBM Tivoli NetView® for z/OS and IBM Tivoli System Automation for z/OS. Mark also teaches IBM classes about these products. He has certification from The Open Group as a Master IT Specialist.

Ryotaro Sawada is an IT Specialist in Japan. He has five years of experience in System z technical support. His areas of expertise include z/OS, IBM Tivoli NetView for z/OS, IBM Tivoli System Automation for z/OS, and IBM Tivoli Workload Scheduler. Ryotaro has supported several System z clients in the area of the system management.

Thanks to the following people for their contributions to this project:

David Bennin
Bob Haimowitz
International Technical Support Organization, Poughkeepsie Center

Gavin Foster
IBM Australia

Robert Abrams
Riaz Ahmad
Stephen Anania
Chris Brooker
Jim Caffrey
Anuja Deedwaniya
Jon Entwistle
Erin Farr
Angela Fatzinger
Garth Godfrey
Larry Green
Frank Hagerty
Marianne Hammer
Joe Jeffrey
Kevin Kelley
Edward King
James LaBarge
Amy Lander
Larry Law
Randall Melton
Geoff Miller
Danny Nieves
Sandeep Perumbuduri
Daniel Rinck
Neil Shah
Paul Smith
John Troy
Doug Zobre
IBM USA

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- Send your comments in an email to:

redbooks@us.ibm.com

- Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

<http://www.facebook.com/IBMRedbooks>

- Follow us on Twitter:

<http://twitter.com/ibmredbooks>

- Look for us on LinkedIn:

<http://www.linkedin.com/groups?home=&gid=2130806>

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>

- Stay current on recent Redbooks publications with RSS Feeds:

<http://www.redbooks.ibm.com/rss.html>

Summary of changes

This edition contains minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-8070-00
for Extending IBM z/OS System Management Functions with IBM zAware
as created or updated on August 1, 2013.

March 2013, First Edition

This revision reflects the addition, deletion, or modification of new and changed information described below.

Changed information

- Minor corrections.



Introduction to IBM zAware

This chapter introduces the IBM System z Advanced Workload Analysis Reporter (IBM zAware). It discusses the challenges faced by organizations to meet ever-increasing availability objectives. It also introduces the tools that IBM provides to use the z/OS rich messaging capabilities in helping you meet those objectives.

The following topics are discussed:

- ▶ Why we wrote this book about IBM zAware and other IBM System z system management products
- ▶ The IT availability challenges facing all organizations
- ▶ The challenges that IBM zAware helps you to handle
- ▶ The architecture of IBM zAware
- ▶ The layout of the remainder of the book

1.1 The role of zAware in maximizing availability

Most modern businesses rely so heavily on information technology that an unplanned outage can result in financial and reputation consequences for the business. We have all grown accustomed to the frenzied news headlines about the impact of a “computer failure” at some company or other.

In this book, we demonstrate how IBM z/OS system management products and functions can be used to improve the availability and resiliency of your systems. We also introduce the IBM System z Advanced Workload Analysis Reporter (IBM zAware) and show how it fits into this family of functions and products.

The availability challenges of today are fundamentally the same as those faced since the beginning of the IT industry. To set the reference for today, briefly look at the history of computing from an availability perspective.

In 1981, the IBM Journal of Research and Development published an article titled *Reliability, Availability, and Serviceability of IBM Computer Systems: A Quarter Century of Progress*¹ (the diagram in Figure 1-1 on page 3 is derived from that article). It reviewed changes in technology and explained how those changes were leading to better management of the mainframe computing platform and improvements in system availability.

Now, more than thirty years after that article was published, this book explores some of the same issues. Although the technology has advanced significantly during this period, the fundamental challenge of improving Reliability, Availability, and Serviceability (RAS) still exists.

The IBM mainframe solution for large systems general purpose computing provides all the required RAS characteristics. However, IBM continues to strive to improve its capabilities. The mainframe has been improving RAS with each new generation of processors with better fault tolerance in the hardware components, built-in redundancy of components, and improved software recovery. Each of these improvements in component technology contributes to making mainframe applications ever more available.

Mainframe computer systems are complex. They work cooperatively to provide RAS, ranging from the hardware and its fault tolerance and recovery architectures, to the operating system and its ability to manage the hardware and software executing on it, and to the applications we use to provide services. Each component has a role, and to ensure that these components are available we need warning and recovery systems; that is, systems management.

¹ IBM Journal of Research and Development, Volume 25, No. 5, September 1981

The diagram in Figure 1-1 shows the layers of Reliability, Availability and Serviceability. Each layer protects and corrects itself where possible, and informs the layers further out of any problems that it cannot self-correct. Problems are identified and fixed as close to the source of the problem as possible. However if a component is unable to self-correct, notification goes to a higher layer for capture, analysis, and recovery.

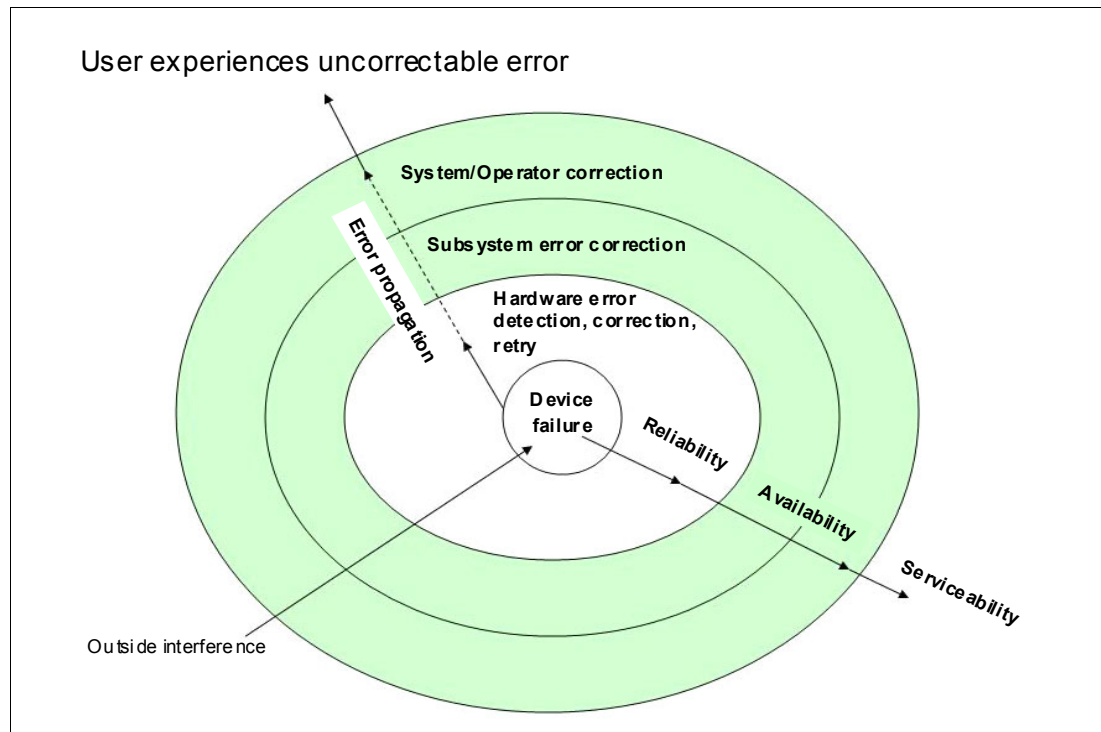


Figure 1-1 Reliability, Availability, and Serviceability

This book focuses on the Availability aspect of RAS and highlights IBM tools that can help you to improve it.

Because computing technology has advanced dramatically since 1981, you now have access to more information. The vastly increased computing capacities give you the ability to collect and analyze large amounts of data, which then turns into information. IBM zAware adds analytical capabilities to turn that information into insights that can help you improve the availability of your systems.

1.2 Challenges faced by IT departments

In your enterprise, you might have one z/OS system or many z/OS systems. They might be financial systems, or be vital to national security. They might serve a retail organization, or perhaps run an airline. Regardless of the role or the functions they support, your z/OS systems are expected to meet increasingly stringent availability objectives.

To optimize the availability of these systems, you might have implemented a high availability architecture in which you have removed as many single points of failure as possible. You might have duplicated your network links, your operating systems, applications, and all the other components that make up your service, to improve the availability of your business applications.

Yet even with high availability architectures, and regardless of the industry, there are challenges such as these to overcome in managing your systems and the new technologies you deploy:

- ▶ Attracting and retaining skilled staff to configure and maintain your technology, and keeping their skills current with new technologies.
- ▶ Managing the products that are used in your organization:
 - Handling the increases in the variety of messages being issued, and the rate at which they are issued
 - Determining which messages are important
 - Managing changes to messages
 - Identifying new messages and determining whether they are important in your environment
- ▶ Determining whether a system is configured appropriately:
 - Checking configuration settings against industry preferred practices values or organization-selected values
- ▶ Determining the operating state of your systems:
 - Identifying when the system might be misbehaving
 - Capturing abnormalities in system behavior or configuration, preferably before they become visible as a service interruption

Often mainframe features are taken for granted. One of those features is the wealth of information that is provided about the functioning and health of the system by console messages and commands. A colleague who moved from the mainframe to another platform was once heard to comment “I used to complain about all the messages I got on MVS. But at least we *got* messages!” This is an indication of the value that messages can provide when you are trying to investigate a system problem.

Apart from its obvious purpose of conveying a specific piece of information, the message traffic on a system can be looked at from a number of perspectives:

- ▶ Specific messages inform you about problems with a particular component or with the system as a whole.
- ▶ Specific messages require a well-known response. These can be added to and handled by your automation product.
- ▶ Messages that you have not seen before, or that are quite rare, might indicate a problem.
- ▶ Receiving many messages from a given address space or with the same prefix might indicate a problem.
- ▶ Receiving similar messages from different address spaces (timeout messages, for example) can indicate a problem.
- ▶ Sudden increases in message rates across a number of members of the sysplex can indicate a potential problem.
- ▶ Receiving many more messages than normal might indicate a problem.
- ▶ Receiving significantly fewer messages than normal might indicate a problem.

Similarly, the *presence* of particular messages, for example \$HASP395 SMFDUMPV ENDED, might be an indication that the system is still processing work successfully.

The point is that there are many different ways that system messages can be used. Each of those different perspectives can potentially convey valuable information about the health or

lack of health of the system. IBM zAware, together with its companion functions and products, makes use of this attribute of z/OS to help you better manage your z/OS environment and deliver higher levels of service.

1.2.1 Keeping up with changing technologies and skills

The rapid changes in technology that IT professionals are experiencing today are both interesting and challenging. You need to constantly adjust your skills to manage the new technology. Similarly, technology needs to take into account the skills that are available in the marketplace.

For technologists with many years of mainframe experience, life becomes a balance between maintaining your expertise on ever-changing processes and products, and developing the next generation of mainframe professionals.

For many years, enterprises drove up their mainframe processing capacity, transaction rates, data volumes, and complexity. And the famous mainframe system management strengths enabled this exponential growth without increasing staff numbers. It was only with the relatively recent resurgence of the mainframe that many companies started hiring mainframe staff again. As a result, the experience profile of mainframe staff in your company might reflect the curve shown in Figure 1-2.

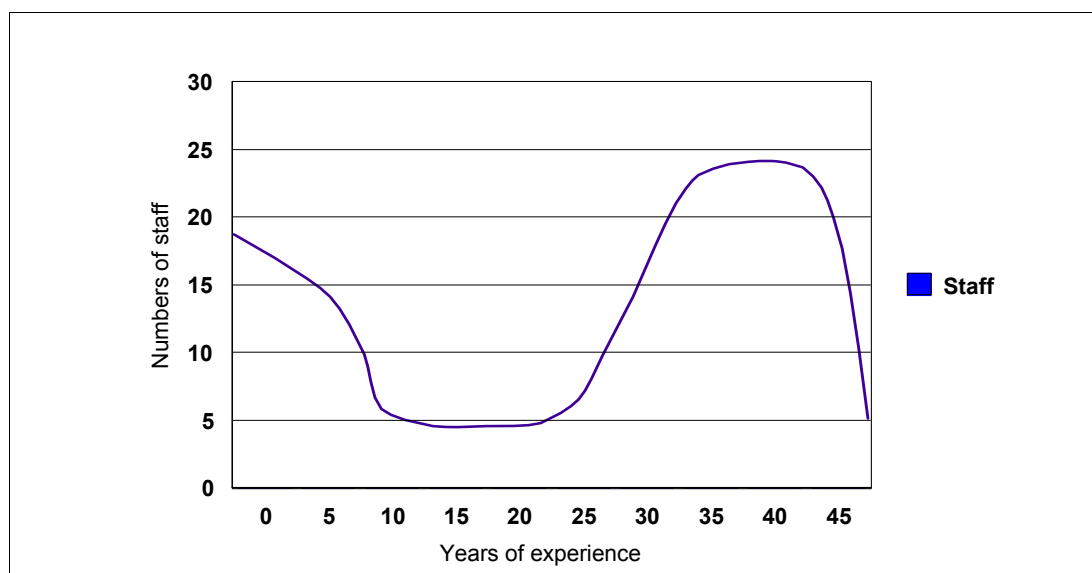


Figure 1-2 Distribution of mainframe experience

The new generation of mainframe staff often belongs to so-called “Generation Y,” referring to those born between the early 1980s and the early 2000s. This generation has many years of computer experience before they join the workforce. However, that experience is likely to be with interfaces that are different than the traditional mainframe “green screen.” To minimize the length of time needed to enable this new mainframe staff to become productive, information must be presented in a format that is familiar to this generation.

IBM zAware combines the traditional richness of z/OS console messages with an intuitive interface that can quickly be mastered by both new and experienced mainframe staff members.

1.2.2 Keeping up with product changes

Mainframe systems are typically a combination of many products. Every time you install a new release of a product it might provide new functions, retire various older functions, and change existing functions.

The product's migration manual documents what you need to consider when migrating to a new release. However, these migration manuals can be quite large and reviewing them to determine how the new release will impact you can take a significant amount of time.

Apart from new releases, maintenance can also add functions and change existing functions. Presumably the text that comes with the fix describes the changes that it introduces, especially if new messages are being added. Even so, do you always have the time to review all this documentation and understand how it might impact your systems?

After you determine how to migrate your existing functions, you then might need to consider what new functions are available and whether to take advantage of them.

Any product update or maintenance you install can have large numbers of console messages that can potentially be issued. How do you manage that? Do you read the message manual and try to determine which ones are important and add them to your monitoring tools? Or do you just add messages *after* your investigation of a problem identifies a message that is related to that problem?

Even with IBM message standards, you cannot always rely on messages ending in A (Action), E (Eventual Action), or W (Warning) being the important messages on *your* systems. Quite often the informational messages are the ones that need to be captured and to have a response automated.

This is another example of an area where IBM zAware can help you. Based on the model it builds of your system's past message behavior, which includes IBM, vendor, and application messages, IBM zAware can help identify messages that might be of interest to you, without you having to explicitly predefine them in any tool.

1.2.3 Ensuring products are configured appropriately

Each new version of a product can add new parameters, and can change the value of existing ones. Although unusual, it is possible that the default value for a parameter can change, thus causing the product to behave in a different way even though you did not explicitly change anything.

Even apart from new versions or fixes, it is possible that a change in your environment or configuration can result in a product behaving differently, or a single point of failure being introduced that you are not aware of.

IBM Health Checker for z/OS checks your configuration against the IBM-, vendor-, or user-provided best practice and informs you about any deviations. Additionally, IBM zAware can help by informing you about anomalous messages that might indicate possible configuration problems.

1.2.4 Analytics can help address these issues

New technology brings new challenges in the form of new skills to be mastered, new products to manage, and more complexity and interrelationships. However, new technologies can also

be harnessed to help you. One such new technology is *analytics*. Analytics has been described as the science of logical analysis.

The challenges we discuss here can benefit from analytic capabilities. By applying modelling and domain-specific knowledge to the information the system is producing, analytics can provide you with more insight into what might be occurring on your systems.

This is where the analytic engine of IBM zAware and its results might help you pinpoint changes in system message behavior. The IBM zAware application looks at the current behavior of your system and compares it to a model of past behavior. Based on the information available to it, IBM zAware can warn you about changes in messages on your system before the reason for the difference escalates into a problem.

1.3 Overview of IBM zAware

No one likes to get the call informing you that a critical business application is unavailable or impaired in its ability to perform its functions. IBM zAware, used together with your other system management tools, provides another view of your systems' behavior and helps you answer questions such as:

- ▶ Are my systems showing abnormal message activity?
- ▶ When did this abnormal message activity start?
- ▶ Is this abnormal message activity repetitive?
- ▶ Are there messages appearing that have never appeared before?
- ▶ Do the times of abnormal message activity coincide with problems in the system?
- ▶ Is the abnormal behavior limited to one system or are multiple systems involved?

IBM zAware creates a model of the normal operating characteristics of a z/OS system using message data captured from OPERLOG. This message data includes any well-formed message captured by OPERLOG (that is, any message that has a message ID), whether it is from an IBM product, a non-IBM product, or one of your own application programs. This model of past system behavior is used as the base against which to compare message patterns that are occurring now. The results of that comparison might help you answer these questions.

IBM zAware determines, using its model of each system, what messages are new or if messages have been issued out of context based on the past normal behavior of the system. The model contains patterns of message ID occurrence over a previous period and does not need to know what job or started task issued the message. It also does not need to use the text of a message.

One of the challenges faced by anyone trying to monitor a z/OS system is that the rate at which console messages are issued on modern z/OS systems is no longer consumable by a human operator. In addition to the sheer volume of messages, there is also the complexity caused by an increasing number of products, and the relationships between those products, to be considered. IBM zAware can consolidate the vast volume of message traffic on your systems and analyze them to pinpoint anomalies that might be causing your systems to have problems. It can also detect relationships between products or events that you might not even have been aware of.

IBM zAware complements your existing system management products by analyzing *all* messages, rather than having to be told in advance which ones to monitor. You might have IBM Tivoli NetView for z/OS or IBM Tivoli System Automation for z/OS installed and use them

to monitor your systems for known exceptions. But their monitoring is based on information that relates specific messages to particular problem situations and indicates what actions to take. If new messages are not defined to automation, the message will more than likely be ignored, resulting in potentially important information going undetected. Because these are messages that were never seen on that system before, they are ideally suited to IBM zAware, which is specifically looking for anomalous messages.

IBM zAware strives to help you resolve complex problems faster. It does this by identifying the following types of anomalous message behavior:

- Rare or never-seen-before messages

IBM zAware assigns a score to a message ID based on how often the message appears in the model for that system. A score of 101 means that there were no occurrences of the message in the model. A score of 0 means that the message appeared in nearly every interval. The higher the score, the less frequently that message appeared in the model.

- Frequent messages

Frequent messages are those that are being seen more frequently than in the model for that system.

- Messages out of context

Messages out of context are those that are found outside of their normal cluster of messages based on the model for that system.

Tip: Some commonly-used terms have special meanings when used in relation to IBM zAware:

Interesting IBM zAware algorithms have a set of message IDs that are identified as indicative of a diagnostically useful event. It refers to those messages as “interesting” messages.

Unique When IBM zAware talks about having a number of “unique” message IDs in an interval, this does not mean that the message IDs were never seen before, or that they do not appear on any other system. Rather, it refers to how many *different* message IDs it observed in the interval.

IBM zAware provides answers to questions at the following critical times:

- After a change has been made on your system

When new software levels such as a new release of the operating system, middleware, or applications are installed, or when system settings or system configurations have been altered, IBM zAware answers the following questions:

- Are new messages being issued during the periods immediately following changes to your system?
- Are more messages issued than expected?

- When looking for the cause of a random, intermittent problem:

- Which system is behaving abnormally based on its having a significant number of messages not seen before, or a high score generated by anomalous messages or anomalous message patterns?
- Is more than one system exhibiting a changed pattern of message activity around the same time?
- When did the system start to behave abnormally?

- For a selected 10-minute interval:
 - Are new messages that were rarely or never seen in the model being issued during periods leading up to the problem or during the periods when the problem is reported?
 - Is the volume of messages higher than expected?
 - Are messages issued out of context, meaning that they are normally seen in a cluster, but did not appear with the other messages that they normally appear with?
 - Was there an unusually high number of any particular message?
 - What messages are unusual?
 - Is a particular component issuing unusual messages?
 - How often did the message occur?
 - When did the message start to occur?

With IBM zAware, each system in your sysplex is analyzed separately. However, IBM zAware provides you with a view that shows all systems in your sysplex so that you can easily find the system that might be the cause of the problem and drill down for more details. This view also lets you easily see if there appears to be a relationship between a number of systems, all displaying anomalous behavior around the same time. You can also use IBM zAware to identify situations you might not even have been aware of, and to gain insight into relationships that you might not know exist.

The information presented on the IBM zAware Analysis view of the user interface (Figure 1-3 on page 10) is designed to help you quickly discover differences in the message profile of your system. The Analysis view shows a 24-four hour view (in 10-minute intervals) of the message behavior of each monitored system. You can then investigate and determine what, if any, actions to take based on the analysis results that are presented for each interval.

IBM zAware API: IBM zAware itself does not take any actions. However, you can use the IBM zAware API to give your automation tools access to the same information that IBM zAware presents on its GUI.

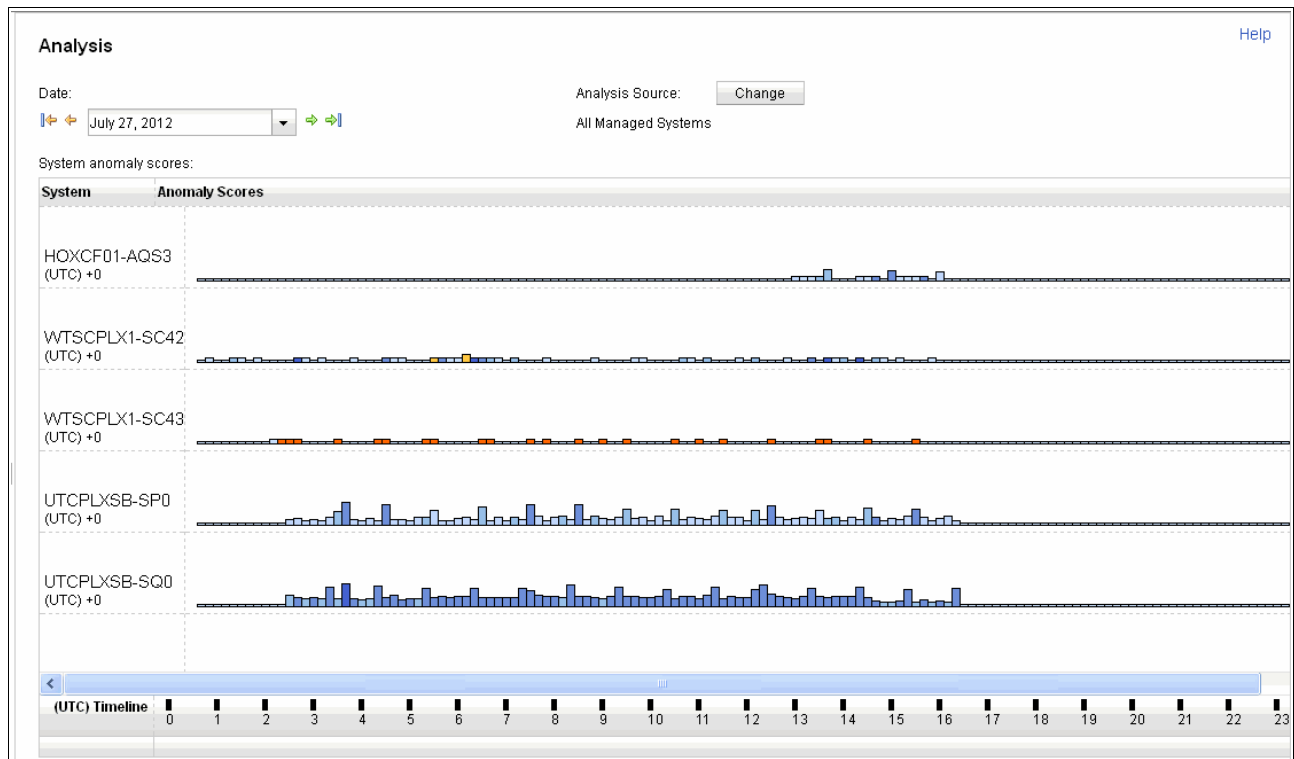


Figure 1-3 IBM zAware Analysis view

Because the IBM zAware application retains a view of past message behavior, you can compare the current period with previous periods to find patterns that might be repetitive. Furthermore, you can compare those periods across systems of the same or other sysplexes, where you might have application components on multiple systems.

The Analysis view shown in Figure 1-3 provides a color-coded and scored view of how close the current message patterns and volumes are to the model of system behavior. The view can also be an indication of the presence of messages that were not seen on that system previously.

The information shown for each interval in the Analysis view is stored in the IBM zAware application file system as XML files. These files can be viewed using the IBM zAware GUI or retrieved using the application programming interface (API) provided with IBM zAware for as many days as you specify the results are to be retained for (up to 10 years).

On the Analysis view in Figure 1-3, you see systems from a number of sysplexes. This view can be changed to show only the systems in a selected sysplex. You can see the results of the analysis of message activity across all the systems at the same point in time, based on the colors and heights of the bars in this view. This can help you see where systems in the same sysplex might be influencing each other. For example a problem on one system, which hosts your databases, might be reflected as changes in messages on your network concentrator system. By displaying only those systems in the same sysplex, you can see whether there are correlations between interval results, at the same time across all the systems.

1.3.1 Architecture of IBM zAware

The IBM zAware tool has two components: the IBM zAware application itself, and the external systems (“monitored clients” in IBM zAware terminology) that send data to IBM zAware for analysis.

At the time of writing, z/OS is the only monitored client of IBM zAware.

The IBM zAware application executes in a special purpose logical partition of an IBM zEnterprise EC12 or later processor. Details about the partition configuration and the IBM zAware application requirements are contained in Chapter 3, “Planning for an IBM zAware implementation” on page 95.

Components of the IBM zAware application

Because IBM zAware is a special purpose partition, the programs making up the application are not available to be viewed by you. However, having an understanding of the concepts making up the different parts of the IBM zAware application can help you better manage the application. Figure 1-4 shows a component view of the IBM zAware application.

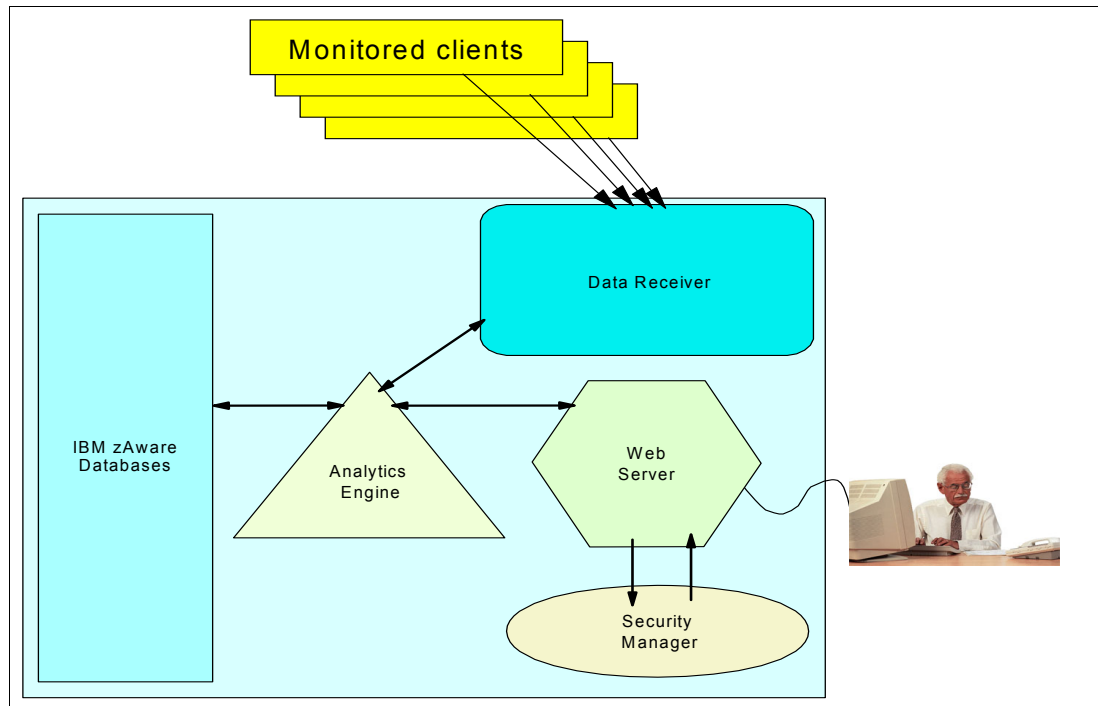


Figure 1-4 IBM zAware component view

The components perform the following functions:

- ▶ The Web server provides the interface between the user and the IBM zAware application. It serves the web pages that are used to view the IBM zAware analysis and to configure the IBM zAware application. It also provides the API to retrieve analysis data from IBM zAware and return it to other system management products.
- ▶ The Security Manager is responsible for controlling access to the IBM zAware application. It manages the various types of security that are supported by IBM zAware:
 - Secure Sockets Layer (SSL) certificates
 - Lightweight Directory Access Protocol (LDAP)
 - IBM WebSphere® Local Repository using the Integrated Solutions Console

- ▶ The Data Receiver manages the TCP/IP connections between monitored clients and IBM zAware. It manages the receipt of the data from all monitored clients.
- ▶ The Analytics Engine performs the analysis of the real-time data. It also performs the training that creates and updates the models of system behavior.
- ▶ The database stores the models, data for the analysis, and the results of the analysis for display or retrieval.

Although the IBM zAware database model is proprietary, you can consider that it contains information such as the message IDs, the interval that each message was issued in, the number of times the message occurred in that interval, the cluster of messages (if any) that the message ID is part of, whether there are any domain-specific rules to be applied to that message ID, sample message text, and other information.

Also, conceptually, IBM zAware has two databases for each monitored client:

- One database contains this type of information for messages that are received in real time or that are sent to IBM zAware using the Bulk Data Load utility. This is called the *instrumentation data database*.

The instrumentation data database can be viewed as a staging area for newly arrived message information. The information in this database is used as input to create or update the model database.

You can control how long this data is retained by IBM zAware.

- One database contains similar types of information that represent the model of this system. We will call this the *model database*.

The model database is updated with the information from the instrumentation data database on a regular basis; this process is known as *training*. When IBM zAware performs its analysis of messages that are arriving in real time, it uses the model database to determine the “normal” behavior for this system.

The retention period for the data in this database is separate from the retention period for the instrumentation data database.

Analysis of real-time messages: When performing its analysis of real-time messages, IBM zAware only compares those messages against the model database. Information in the instrumentation data database that has not yet been used to update the model is *not* included in the message set that IBM zAware uses as the model of that system’s normal behavior.

The topic of determining the appropriate frequency of updates for your model is discussed in “Determining the Training period” on page 172.

IBM zAware communication interfaces

IBM zAware has a web browser-based user interface for configuring IBM zAware and viewing the analysis results. IBM zAware also receives data (console messages), sent using TCP/IP, from monitored clients.

The IBM zAware partition can be defined with any of the following network connections to receive data from a monitored client:

- ▶ Open Systems Adapter (OSA)
- ▶ Hipersockets
- ▶ Intra-ensemble data network (IEDN)

IBM zAware must have at least one OSA. This is used for the web browser interface and, optionally, the monitored client connections.

To establish the socket connection from z/OS to IBM zAware, each monitored client must be configured to connect to the IBM zAware IP address or host name. Figure 1-5 shows how each connection is used.

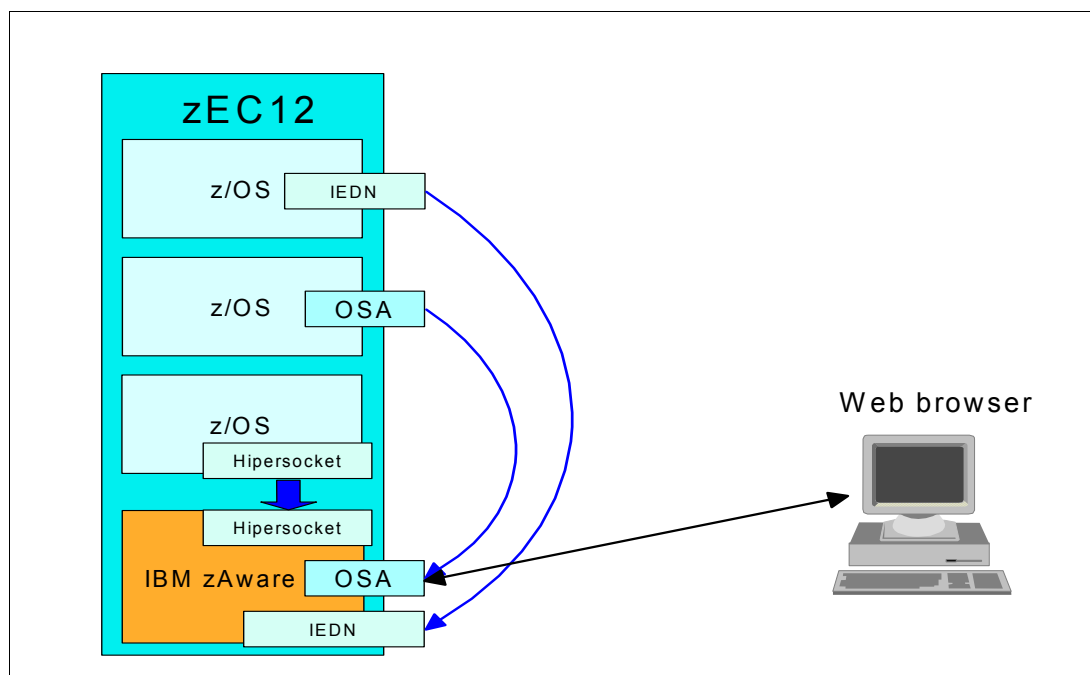


Figure 1-5 IP connections into IBM zAware

To access the web server user interface for IBM zAware, configure an OSA adapter connection to allow access from the supported web browsers. The web browsers can connect using the IP address or the host name of the IBM zAware partition.

1.3.2 Transporting messages to IBM zAware application

To send real-time message data from a monitored client to the IBM zAware application, a connection is needed from the log source on the monitored client to the IBM zAware application.

System Logger sends z/OS messages to IBM zAware from log streams that have been defined with the ZAI parameters. When a message is written to the OPERLOG log stream, the message is copied to a special buffer in System Logger (reserved for use by messages that are destined for IBM zAware). The buffer is then dequeued and sent to the IBM zAware

application across a TCP/IP connection. Figure 1-6 shows the path a message takes from System Logger to the IBM zAware application.

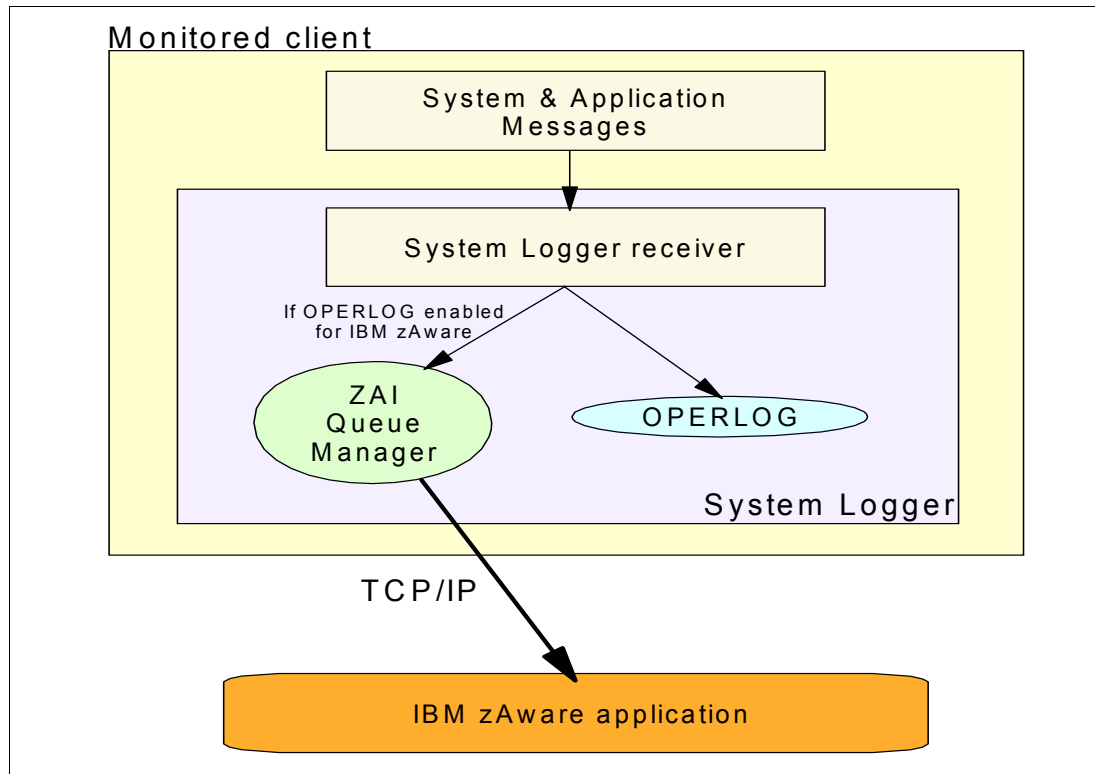


Figure 1-6 System Logger connection to IBM zAware application

1.3.3 Priming IBM zAware with message data

IBM zAware does not perform analysis on incoming messages from a system until it has a model of the normal behavior for a system. If you think about it, this makes perfect sense. After all, how can IBM zAware know whether or not a given message is “normal” for a system unless it has a model of normal behavior to compare that message against?

And that, in turn, raises the question about how many hours’ or days’ worth of message activity is needed to build a model of normal behavior for a system. The message activity on your systems is probably different on a weekend day than on a mid-week day. And the month-end processing might be different than the rest of the month. For many businesses with quarterly business cycles, the quarter-end activity might be different than at other times. Therefore, the value of the analysis that IBM zAware performs is related to the number of days in the model. A model that is created with only a few days of data will probably not be as representative (and as valuable) as one that contains many days of activity.

Therefore, to produce a representative model, many days of OPERLOG or SYSLOG message data are needed. You *could* build up this information from messages passed to IBM zAware in real time from OPERLOG. However, it would be quite some time before you started to get valuable and reliable analysis results from IBM zAware.

Alternatively, you can extract prior message data from your syslog or OPERLOG archives and send that to IBM zAware using the Bulk Data Load utility that is provided by IBM zAware. A useful interval to start with is 90 days’ worth of data. Within reason, the more message data you provide to IBM zAware, the more valuable will be the resulting model of system behavior.

Figure 1-7 on page 15 shows how System Logger is used to load both real-time and archived log data into IBM zAware. The Bulk Data Load utility is discussed in more detail in “The Bulk Data Load Utility” on page 73.

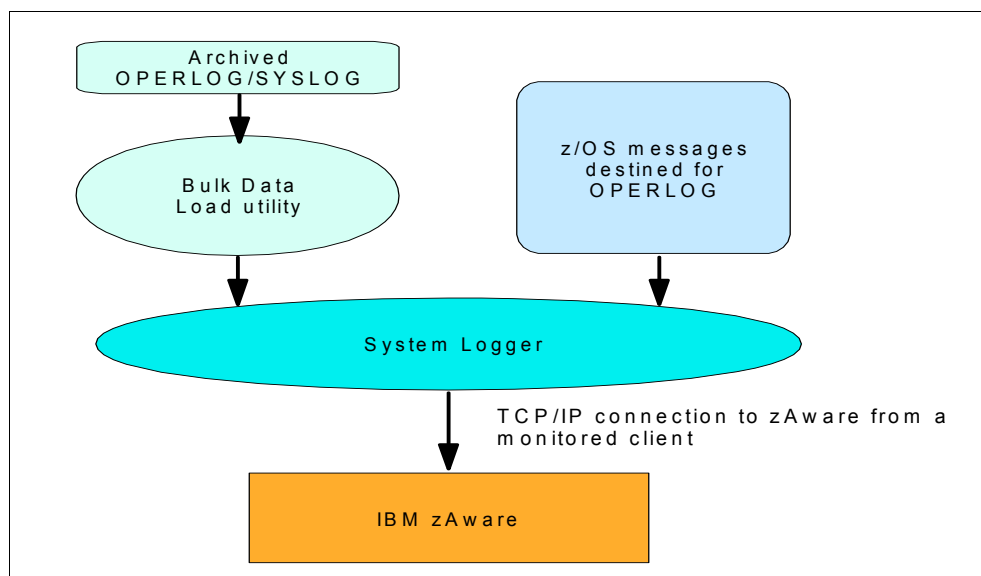


Figure 1-7 How data gets loaded into IBM zAware

Prerequisite events: The Analysis results available through the web browser will not be populated with any information for a given system until both of the following events have taken place:

- ▶ The monitored client has been connected to IBM zAware.
- ▶ A model of the monitored client has been built, using a mechanism known as “training.”

1.3.4 Creating the behavioral model

The IBM zAware application needs a model of the past behavior of a system before it is able to perform any analysis of the messages received in real time from a monitored client. It is the comparison against the model that produces the analysis results that show the behavior of a system.

To create the model of system behavior, IBM zAware provides a function known as “training,” as mentioned. When you train a system, IBM zAware updates its model database using the information from the instrumentation data database. The training process is typically set up to run automatically somewhere between once a week and once a month.

Training considerations:

- ▶ You need to go through the training process for each monitored client.
- ▶ IBM zAware creates a separate model for each system.
- ▶ When analyzing the message traffic for a system, IBM zAware only compares it against the model for that system. Whether a specific message is in the model for another system is not relevant.

Figure 1-8 on page 16 shows that system SC43 has mostly orange bars. Orange means that the anomaly score for that interval is quite high. The reason that SC43 only has orange bars is because its model was created using a small amount of data. As a result, nearly every message that arrives for SC43 is treated as being rare or unique.

Conversely, three months of message data were used to build the models for each of the other systems. As a result, the determination of anomalous activity on those systems is far more accurate and there are few yellow or orange bars for those systems.

This is a useful example to illustrate how the results might not be useful when the model is built using messages from a small number of days. To create a valuable model, use messages from a larger number of days.

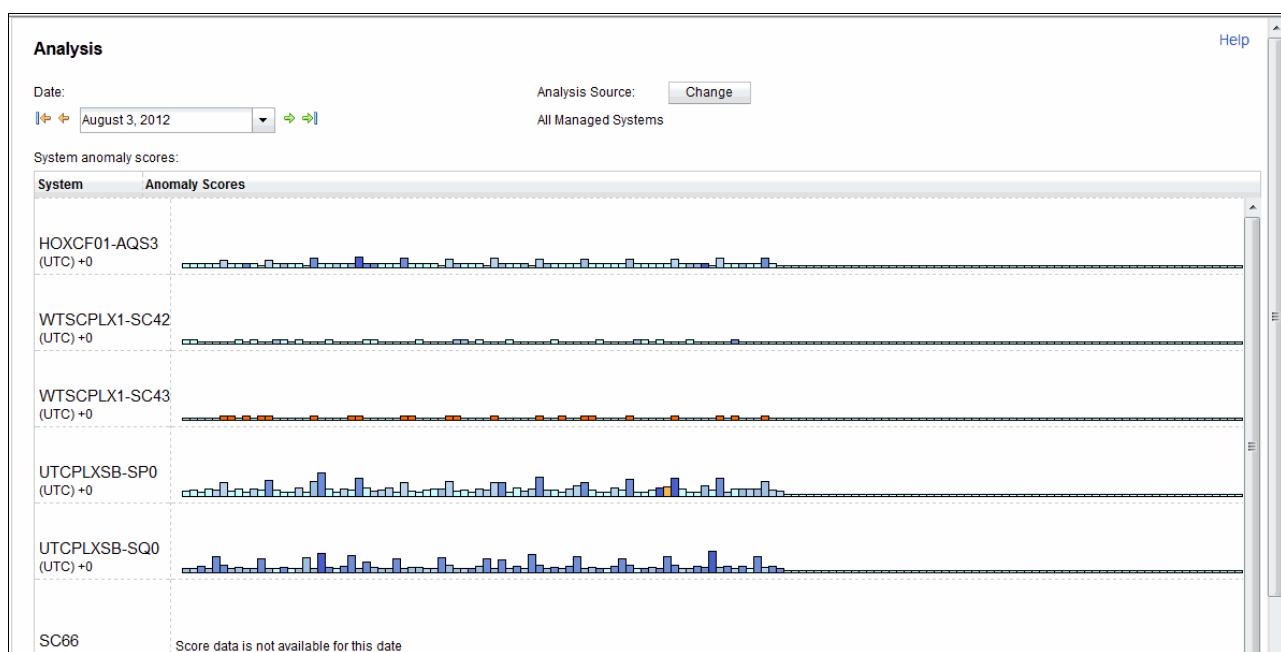


Figure 1-8 Analysis view - results of model using a small training period

The bars for the systems other than SC43 are mostly blue. Light shades indicate results are close to the model of normal behavior. Darker shades indicate the message activity is outside the normal patterns for that system. The height of the bar indicates how many *different* message identifiers have appeared in that period; it does not represent a count of the total messages issued on the system.

Remember that IBM zAware only creates analysis views for data that is received in real time. It does not create those views for a time period that is loaded using the Bulk Data Load utility. For example, in Figure 1-9 on page 17, there are no colored bars for most of the day for systems AQS3 and SP0. That is because the message data for most of the day for those two

systems was loaded using the Bulk Data Load utility. Even though the model includes data for those times, this information will not show up in the analysis view.



Figure 1-9 Analysis view for period with only bulk load data

Updating consideration: Updating the model for a system does *not* result in IBM zAware going back and re-analyzing the data that it received in real time. As a result, the results shown in the Analysis view will not be updated after the model has been updated manually or automatically.

1.3.5 Viewing the results of IBM zAware analysis

After you have prepared the IBM zAware application, you are ready to analyze real-time message data coming from your z/OS systems.

The Analysis page shows the results of the message data comparison between the model and the real-time messages. In 2.7, “Message analysis with the various functions and products” on page 82, sample scenarios are provided that illustrate how IBM zAware analysis can be used with other IBM products and functions to help you monitor your systems.

The Analysis page gives you a view of each monitored client and which sysplex that client is a member of. It also provides a graphical display (for each 10-minute period) of a system's behavior over the current UTC 24-hour period. You can drill down into any 10-minute period to see the message details for that 10-minute interval.

Using the interval details: The interval details can be retrieved using the IBM zAware API and used in your system management tools, for example by IBM Tivoli NetView for z/OS. This is discussed in more detail in 6.3, “IBM Tivoli NetView for z/OS” on page 196.

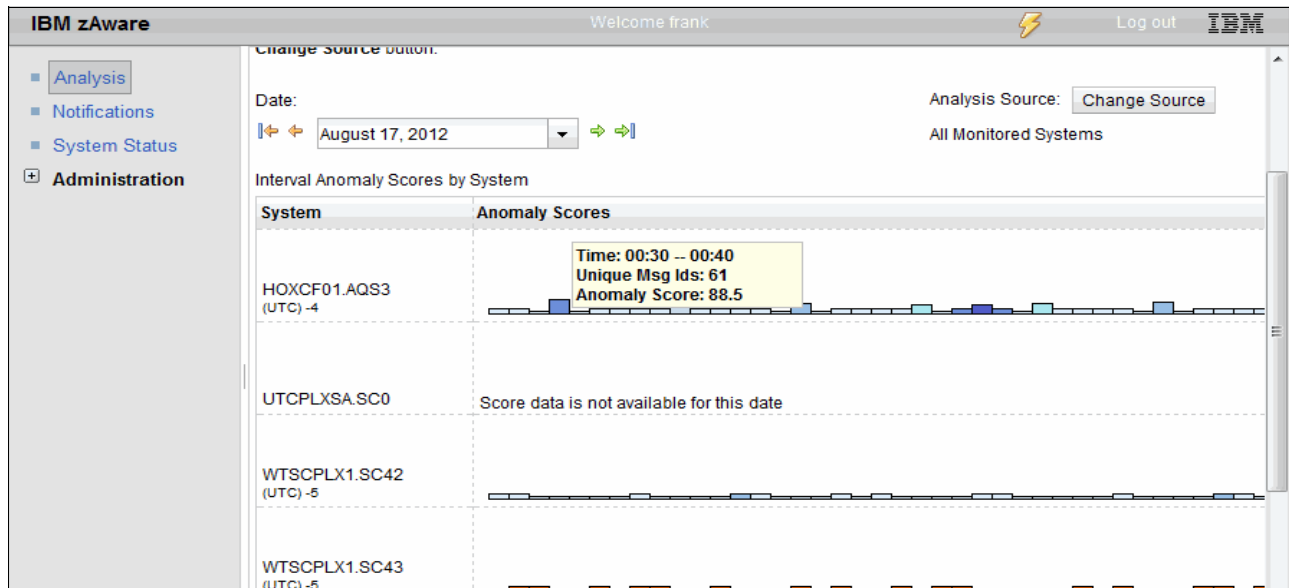


Figure 1-10 Analysis page

Figure 1-10 shows an Analysis page, with each bar representing a 10-minute monitoring period. You can change the zoom level of the display to make the bars larger, by making the time viewed smaller. You can then scroll the Analysis page results to look at the enlarged bars for the time period that you are interested in.

The timeline for the Analysis page is in UTC time. When you position your mouse over each bar, the time in the detail box is also UTC time, as shown in Figure 1-10. By normalizing the reporting for all systems to UTC, you can see what is happening across all systems at the same time irrespective of their local time offset. This is important because activity on one system is often related to events on related systems. You will need to know your UTC offset to determine the local time for each system.

Detailed view of individual Analysis bar

Each bar on the Analysis page can be expanded to show information about the messages that were issued during the interval represented by that bar. It also shows how they contribute

to the color and the interval anomaly score. When you click a bar you are brought to the interval view, as shown in Figure 1-11.

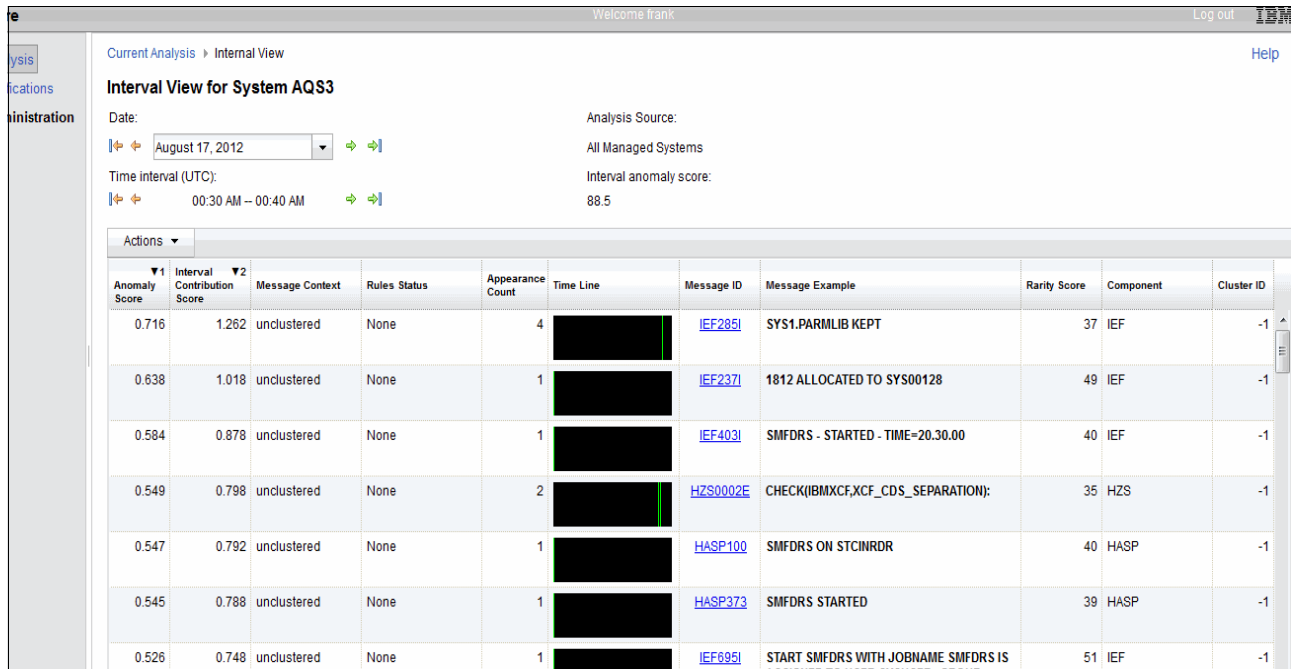


Figure 1-11 Interval View window

The interval view provides information about every message ID that was issued during the interval. The default order of message ID presentation places the largest contributors to the interval anomaly score at the top of the list.

The details of the information provided in this view are provided in “The Bulk Data Load Utility” on page 73. However, at this point it is sufficient to understand that IBM zAware provides you with summary information in these ways:

- ▶ Across all your systems in the Analysis view
- ▶ With detailed information for each system for each interval in the Interval view

This ability allows you to quickly scan all your systems for anomalous messages, and then drill down for the detail of intervals that interest you.

1.4 Layout of this book

This chapter introduces the IBM System z Advanced Workload Analysis Reporter (IBM zAware) and explains why you might need it. It discusses why the book was written, and contrasts the availability challenges faced today with those faced in 1981. It shows how the quest to improve system and application availability is an ongoing journey.

Chapter 2, “IBM z/OS system management functions” on page 21 explains the availability-related z/OS products and functions of IBM zAware, namely Predictive Failure Analysis, Runtime Diagnostics, IBM Health Checker for z/OS, and zOS System Management Facility (zOSMF). That chapter also delves into the details of how these products and functions work.

Also discussed is how to use IBM zAware along with the other tools to identify possible problems in your system. Scenarios are presented that illustrate where IBM zAware can help you, and how to use it with the other tools.

Chapter 3, “Planning for an IBM zAware implementation” on page 95 provides you with the considerations related to planning the implementation of IBM zAware. This includes hardware requirements, software setup, and the skills that are required.

Chapter 4, “IBM zAware installation” on page 113 discusses the process of installing IBM zAware, and illustrates the steps we used to install and customize IBM zAware in our environment. Detailed reference information is provided in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623. Our example illustrates how we completed the installation based on the information in that book.

Chapter 5, “Maintaining and managing IBM zAware” on page 153 answers many questions about the following topics:

- ▶ How to manage and maintain the IBM zAware installation
- ▶ Management of the network connections and how to start, stop, and display them
- ▶ How z/OS system IPLs or system failures affect IBM zAware
- ▶ General day-to-day management of IBM zAware

Chapter 6, “Integrating IBM zAware with other IBM products” on page 191 shows several of the ways that other system management products can integrate the information that IBM zAware provides.



IBM z/OS system management functions

IBM System z has always had a reputation for delivering the highest levels of system availability. The operating system components have integrated failure detection to allow recovery actions to occur without operator intervention. But even with that capability integrated into the operating system, failures can still occur.

As part of the effort to deliver even better availability, System z has created a set of system management functions that not only detect failures, but *avoid* failures. When failures do occur, these new functions help you more easily diagnose the cause of the problem and advise solutions to help you either to avoid an outage or reduce your mean time to recovery. Failures will still occur, but the advancements in failure avoidance and early failure detection mean that more failures can be mitigated before being visible to users, and before they impact your business.

As explained in 1.2.4, “Analytics can help address these issues” on page 6, you cannot achieve high availability by looking at only one view of your system or by using only one tool. Therefore, System z has multiple functions and products that perform their processing at different layers of the software stack, using different metrics for their early detection and different methodologies to detect abnormal behavior.

In this chapter, we highlight the various component functions and products available on System z that are designed to help you achieve your availability goals. We also provide sample scenarios in which each of them can be used.

The following topics are discussed:

- ▶ Overview of System z system management functions
- ▶ IBM z/OS Management Facility
- ▶ IBM Health Checker for z/OS
- ▶ Runtime Diagnostics
- ▶ Predictive Failure Analysis
- ▶ IBM zAware

- ▶ Message analysis with the various functions and products
- ▶ Additional tips
- ▶ Sample scenarios

2.1 Overview of System z system management functions

There are three general categories of software-detected system failures: masked failures, hard failures, and soft failures, as explained here:

Masked failure	A masked failure is a failure that is detected and corrected by the software.
Hard failure	A hard failure occurs when the software fails completely, quickly, and cleanly. For example, a hard failure occurs when the operating system abnormally terminates an address space.
Soft failure	<p>Sometimes a software component behaves in an unexpected or unusual manner. This abnormal behavior can be combined with events that usually do not generate failures, but produce secondary effects that might eventually result in a system failure. These types of failures are known as soft failures. You might have also heard these types of failures referred to as “sick, but not dead” incidents, meaning that the system appears to be running, but is not totally available.</p> <p>Soft failures are often manifested as stalled or hung processes, resource contention, storage growth, or repetitive errors.</p>

Clients have told us that soft failures account for a small percentage of the problems when compared to masked failures and hard failures, but they cause the largest business impact.

Soft failures are difficult to diagnose because the failure often does not occur in the address space causing the problem. In fact, the failure can often occur in an address space that might not even be associated with the issue.

During a soft failure, the problem often escalates from a minor problem in one address space or component to the point that the program eventually stops working and might even cause other components to fail. This sympathy sickness has been observed when either hard failures or abnormal behavior generate a system failure that could not be easily isolated to a failing component or subcomponent.

Because soft failures are difficult to detect, are quite unique, can be triggered anywhere in either software or hardware, and occur infrequently, their isolation is quite difficult. Therefore, to achieve high availability and reduce the rate of system failures, system management functions have been enhanced over the years and new functions have been developed. Additional tools have been developed to help you quickly discover the source of problems when they do arise.

Some of the system management functions are embedded directly in z/OS. These are sometimes referred to as *in-band* processing. Some of these functions help you avoid failures. Others help you detect, diagnose, and recover from the failures.

With the introduction of IBM zAware, we now have a tool that is independent of the operating system. Being independent of the operating system is sometimes referred to as *out-of-band* processing. Having an independent systems management function allows diagnosis of failures to occur even when the operating system is nonfunctional. It also allows compute-intensive analysis to be offloaded from the operating system's LPAR.

System message logs have always been a critical resource for diagnosing operating system, middleware, and application problems. However, the volume of messages sent to the z/OS logs makes it difficult for operators and system programmers to quickly identify the messages required to pinpoint the problem.

Systems management functions including Runtime Diagnostics, Predictive Failure Analysis (PFA), and IBM zAware all provide functions that use messages and message logs to aid in detecting and diagnosing a problem. The differences in how these functions and products use the messages is described in 2.7, “Message analysis with the various functions and products” on page 82.

Figure 2-1 illustrates the system management functions included in z/OS. Each of the functions is described later in this book.

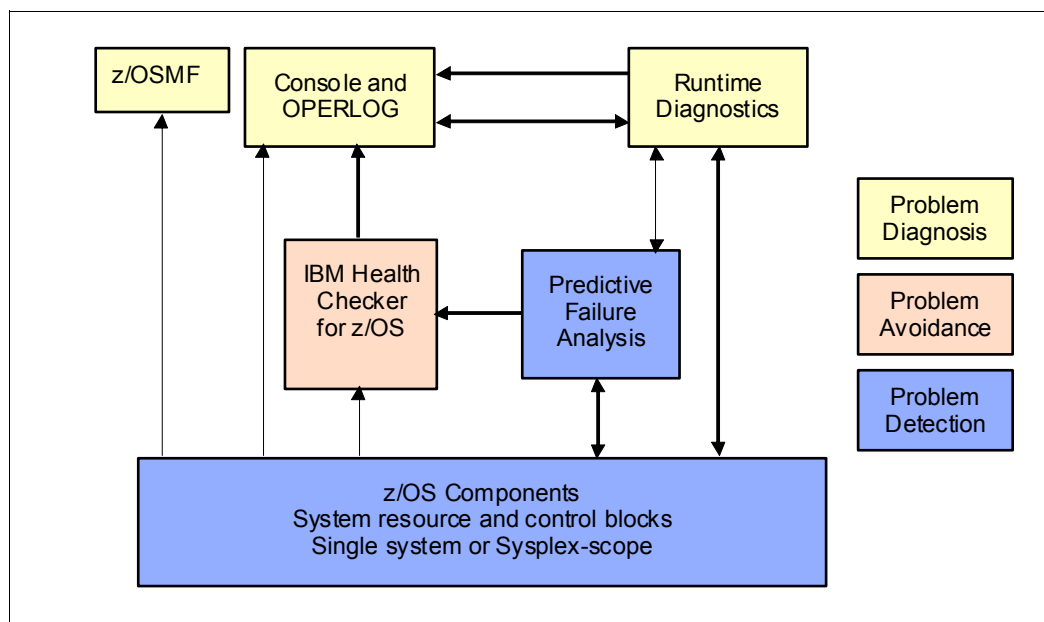


Figure 2-1 z/OS system management component relationship

Notice that each of the system management functions has its own specialty and some of them could be put into more than one specialty category:

- Problem detection
 - z/OS operating system components have always had problem detection embedded in their logic.

As they detect abnormal situations such as software abends, their recovery code is invoked to handle the failure without operator intervention. Some z/OS components also provide soft failure detection, which usually results in an alert of some kind. See 2.1.1, “z/OS component functions” on page 27 for more information about the system management portion of z/OS operating system components.
 - Predictive Failure Analysis (PFA) detects potential soft failures and alerts the operator before the failure escalates into a system problem.

PFA’s output can also be used to help diagnose the failure. See 2.5, “Predictive Failure Analysis” on page 42 for more information about PFA.
- Problem avoidance
 - IBM Health Checker for z/OS is designed to identify configuration errors before they result in soft failures.

Health checks can be written by IBM component owners, ISVs, and by you. Health checks can set thresholds for system resources and detect migration actions that have not been completed so that your systems avoid these types of problems. See 2.3, “IBM

Health Checker for z/OS” on page 30 for more information about IBM Health Checker for z/OS.

► Problem diagnosis

- z/OS OPERLOG and the z/OS console contain messages issued by z/OS components, middleware, and applications.

These messages are useful for diagnosing problems and are a data source for IBM and ISV problem determination and system automation.

- Runtime Diagnostics is a useful tool that can provide quick point-in-time diagnosis of problems.

This component is designed to be used for quick analysis of specific z/OS system issues to give the system operator an indication of where problem investigation should start. You can point Runtime Diagnostics to a system other than the system on which the command is being issued to detect problems on another system in your sysplex. See 2.4, “Runtime Diagnostics” on page 32 for more information about Runtime Diagnostics.

- z/OS Management Facility (z/OSMF) runs on the z/OS system and manages z/OS from z/OS itself.

z/OSMF is a Web 2.0-based application on z/OS, with direct access to z/OS data and information. It serves multiple roles in delivering simplified tasks to improve system programmer and operator productivity.

This book focuses on the role of z/OSMF as it relates to problem determination due to soft failures which is simplified with the z/OS incident log task. See 2.2, “IBM z/OS Management Facility” on page 28 for more information.

So, where does IBM zAware fit in the picture in Figure 2-1 on page 24? It is independent and resides outside of z/OS. It is designed to complement the functions in that figure by using powerful analytics algorithms to *detect* and help *diagnose* message anomalies without the need to access z/OS. This design provides the following benefits:

- It uses fewer system resources. This can sometimes be important when a system problem is occurring.
- It is operational even if all systems in your sysplex are having a problem.
- It allows you to easily see which system or systems are experiencing anomalous message behavior.

Figure 2-2 provides a simplified view of the various z/OS system management functions and products, identifies their locations in relation to z/OS, and gives a rudimentary view of their interfaces.

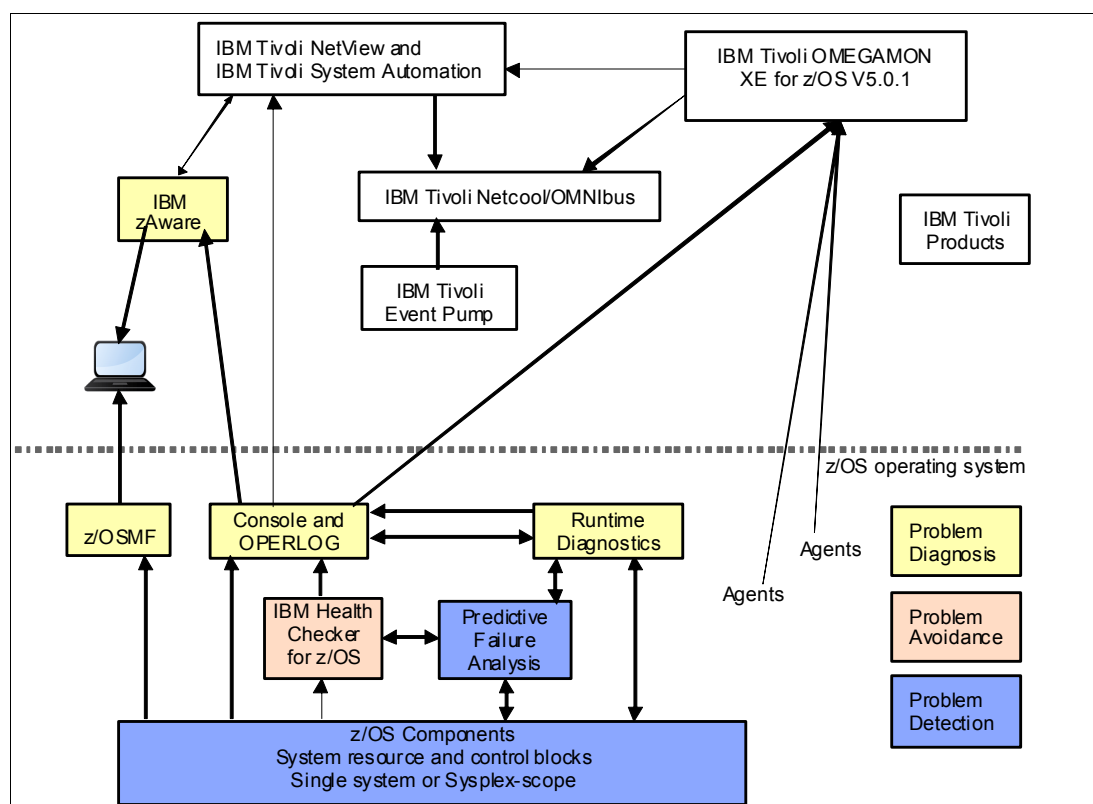


Figure 2-2 Systems management functions and products

Any component or product that uses OPERLOG or other system logs can participate in the systems management realm. For example, both IBM Health Checker for z/OS (including PFA output) and Runtime Diagnostics send their results to the z/OS system log.

Other examples include:

- ▶ The IBM Tivoli NetView CANZLOG function consolidates messages from the z/OS syslog and the NetView netlog. NetView can automate messages from either of these sources. Additionally, as shown in Figure 2-2, NetView is able to use the IBM zAware API to retrieve information from the IBM zAware LPAR, allowing you to consolidate information from IBM zAware into the NetView operator interface.
- ▶ IBM Tivoli System Automation can be used to automate messages, and policies can be established to control corrective actions.
- ▶ IBM OMEGAMON® XE for z/OS 5.1.0 not only uses the messages, but also shows all health checks produced by IBM Health Checker for z/OS (including those owned by PFA).
- ▶ IBM zAware uses analytics to detect anomalous behavior using the OPERLOG.
- ▶ Performance issues and system problems can be detected by OMEGAMON.
- ▶ Netcool/OMNIBus provides centralized monitoring of health check alerts, performance, network activity, and so on.

- ▶ Tivoli Event Pump parses the messages, interprets the resource information within the messages, and converts the message to an event that can be read by other products.
- ▶ You can get alerts if PFA detects an abnormal condition and forward those events to other Tivoli event management products.

2.1.1 z/OS component functions

As shown in Figure 1-1 on page 3, detecting soft failures as close to the source as possible uses the minimum amount of system resources and ensures that the failure is responded to in the minimal time. Whenever possible, z/OS components try to detect and address soft failures themselves.

The following list provides a few examples of how z/OS components detect problems on a *single* system:

- ▶ The GRS component added enhanced contention analysis to identify enqueue blockers and waiters a number of releases ago. More recently, GRS implemented similar support for latches. Both enqueue contention detection and latch contention detection use the **D GRS,ANALYZE** command. The latch identity string is exploited by z/OS UNIX System Services, the Logger component, and IBM RACF® for their latches. In addition, Runtime Diagnostics takes advantage of the functions provided by GRS in its analysis of enqueue contention and GRS latch contention.
- ▶ The Missing Interrupt Handler (MIH) component intercepts incomplete I/O operations to prevent an application or system outage due to a device, control unit, or hardware (cabling) error. After the scope of the problem is understood, hardware and software recovery mechanisms are invoked and diagnostic data is captured.
- ▶ The system invokes the I/O Timing Facility to identify slow I/O response times by monitoring I/O requests that exceed I/O timing limits for devices.
- ▶ IBM has made improvements in channel recovery such that for frequently occurring path errors, the path is taken offline rather than having the hardware or operating system repeatedly try to recover the path.
- ▶ z/OS can detect contention in the Catalog Address Space, which identifies catalog tasks that appear to be stuck while waiting on an event. When this occurs, a symptom record is created and the task is terminated if the system deems it is safe to do so.
- ▶ The JES2 Monitor assists in determining why JES2 is not responding to requests. It monitors conditions that can seriously impact JES2 performance.

The following list provides a few examples of how z/OS components detect problems in a *sysplex*:

- ▶ XCF stalled member support detects whether a system that appears to be healthy is actually performing useful work. There might be critical functions that are non-operational that are making the system unusable and induce sympathy sickness elsewhere in the sysplex. When a problem is detected, action should be taken to restore the system to normal operation or remove it from the sysplex to avoid sympathy sickness.
- ▶ Sysplex Failure Management (SFM) detects and can automatically address soft failures that can cause sympathy sickness conditions when a system or sysplex application is unresponsive and might be holding resources needed by other systems in the sysplex.
- ▶ With *critical member* support, XCF terminates a critical member if it is “impaired” long enough. A critical member is a member of an XCF group that identifies itself as “critical” when joining the group. For example, GRS declares itself as a critical member when it joins its XCF group. If GRS cannot perform work for as long as the failure detection

interval, it is marked as impaired. GRS indicates that when this occurs, the system on which it is running should be removed from the sysplex to avoid sympathy sickness.

- ▶ System Status Detection (SSD) uses z/OS BCPii to let XCF query the state of other systems through authorized interfaces through the Support Element and HMC network. If XCF detects that the system in question is in a state where it can no longer operate or recover, the system can be immediately partitioned out of the sysplex without having to wait for the failure detection interval to expire.

2.2 IBM z/OS Management Facility

IBM z/OS Management Facility (z/OSMF) simplifies, optimizes, and modernizes the z/OS system programmer and operator experience. z/OSMF delivers solutions using a task-oriented, web browser-based user interface to improve productivity by making the tasks easier to understand and use. z/OSMF makes the day-to-day operations and administration of your system easier for both new and experienced system programmers and operators.

2.2.1 z/OSMF tasks

The primary objective of z/OSMF is to make new staff productive as quickly as possible. This is accomplished by automating tasks and reducing the learning curve through a modern, simplified, and intuitive task-based, browser-based interface.

z/OSMF provides assistance in a number of areas:

- ▶ A configuration category with the Configuration Assistant
- ▶ The Links category that lets you add web links to the z/OSMF GUI
- ▶ A Performance category containing sysplex status, monitoring desktops, and workload management tasks
- ▶ The Problem Determination category that provides the incident log function
- ▶ The z/OSMF Administration category to let you administer z/OSMF itself

Because this book is focused on tools to help you avoid, detect, and diagnose soft failures, we only discuss the incident log task here.

Details about z/OSMF and its tasks can be found in other publications such as *z/OSMF Configuration Guide*, SA38-0652 and the IBM Redbooks document *z/OS Management Facility*, SG24-7851.

2.2.2 z/OSMF Incident Log

With its focus on simplification, z/OSMF is a natural addition to the set of z/OS functions focused on high availability and problem determination. Even with all the system management functions designed to avoid and detect failures, what if a problem still occurs? You need to diagnose and resolve the problem as quickly as possible to reduce your mean time to recovery and hopefully ensure that the problem does not occur again. The z/OSMF Incident Log task is designed to help you achieve higher availability by making problem determination diagnostic data management easier and quicker.

The reality is that when a problem occurs, it can be complicated and time-consuming to collect the right data and documentation to perform your problem analysis. And after you

have the data for your problem analysis, it can be difficult to manage, particularly for less experienced staff.

The z/OSMF Incident Log task was created to alleviate these common troubleshooting challenges. It improves the first failure data capture for human-detected and system-detected problems that result in SVC dumps and creates diagnostic data snapshots for log data. It also provides a simple interface that allows you to perform these tasks:

- ▶ FTP this data to IBM or other software providers
- ▶ Display summary and detail information
- ▶ Drive incident management actions
- ▶ Manage the data related to the incident

For example, an incident might contain snapshots with 30 minutes of OPERLOG data (or SYSLOG data, if OPERLOG is not activated), one hour of LOGREC detail, and 24 hours of LOGREC summary. These materials are commonly required for z/OS-related problems reported to IBM service.

You can review all the incidents for your sysplex and drill down to see the diagnostic data associated with each incident. These details include the key system problem symptoms and the system symptom string.

If needed, you can then easily FTP the documents to IBM or an ISV or elsewhere for further debugging without having to remember where the data was archived. It also simplifies the steps involved to take another dump for a previously-recognized problem.

The z/OSMF Incident Log also integrates the IBM z/OS Problem Documentation Upload Utility (PDUU), which encrypts all of the files associated with an incident and transmits them to IBM using parallel sessions to send the material (such as a large system dump) more quickly.

z/OSMF Incident Log is accessed directly from the IBM z/OS Management Facility main panel and is located under “Problem Determination” as shown in Figure 2-3.

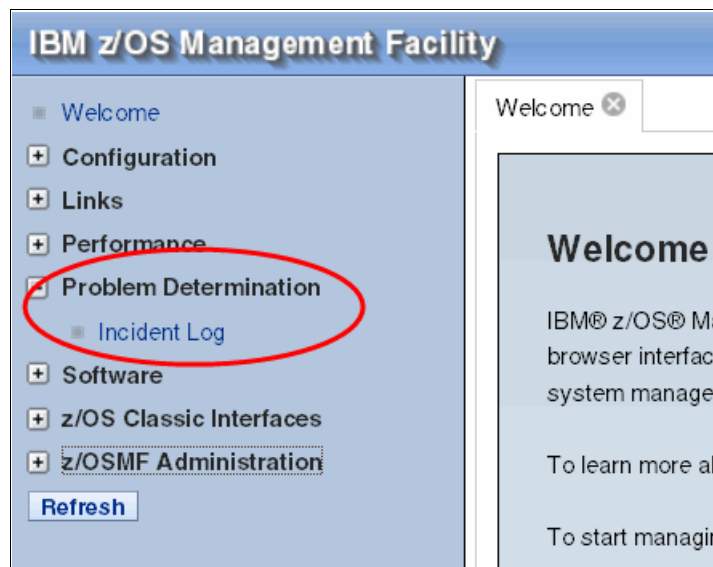


Figure 2-3 z/OSMF Incident Log

2.3 IBM Health Checker for z/OS

The IBM Health Checker for z/OS helps identify potential configuration problems before they impact availability or cause system outages. It checks the active z/OS and sysplex settings and definitions and compares them to IBM recommended best practice configuration values or customer-defined overrides. It generates output with detailed messages and reports to inform you of any potential problems and suggested actions to resolve them.

IBM Health Checker for z/OS is a preventative application that should be set up to run continuously on all systems to detect potential problems and to alert you if such a situation exists. It is also useful when new software releases or maintenance is installed on your system that might result in configuration issues.

It is not intended as a monitoring tool itself, but the output from health checks can be used by your monitoring tools. For example, NetView could raise an alert if a Health Checker exception above a certain level is issued. Also, some Health Check exception messages are contained in the “domain-specific” rules in IBM zAware.

IBM Health Checker for z/OS consists of the following parts:

- ▶ Framework

The framework is an interface that manages services like check registration, messaging, scheduling, command processing, logging, and reporting. It is an open architecture that supports check development by IBM products, independent software vendors (ISVs), and customers.

The framework is used by Predictive Failure Analysis as described in 2.5.3, “PFA and IBM Health Checker for z/OS integration” on page 45.

- ▶ Checks

Checks are programs or routines that evaluate component, element, or product-specific setting or definition, and look for potential problems on a running system. Checks are independent of the framework. The IBM checks are owned, delivered, and supported by the corresponding element of z/OS.

The following is an example of the types of things that the health checks provided by IBM look for:

- ▶ Check that specific system settings conform to best practices and recommendations.
- ▶ Check the configuration for single points of failure.

It is especially important that these checks run on a regular basis in case an unnoticed hardware change or failure has inadvertently introduced a single point of failure.

- ▶ Check the usage of finite system resources, such as common storage.
- ▶ Check that all the system’s availability-enhancing capabilities are being exploited.
- ▶ Check that sensitive system resources are protected.

You can, of course, also use vendor-provided or user-written checks to look for things such as:

- ▶ Ensuring compliance with installation-defined system and conventions. For example, you can ensure that user IDs are set up consistently based on the company’s requirements.
- ▶ Implement and confirm security compliance and audit. For example, ensure that installation-specific resources are protected properly.

- ▶ Check for configuration settings that changed since the last IPL to determine whether the change will be lost upon the next IPL. For example, compare the static content in Parmlib members used at IPL to the settings that are currently in effect.
- ▶ Detect changes, either those implemented dynamically or introduced by an IPL, that cause the system to diverge from the recognized best practice or your override. For example, you might have determined that the optimal setting for the WIDGET parameter in this system is 13. If a colleague changes that parameter to some other value without informing you or going through the normal change management process, a health check could inform you of the change.

Starting with z/OS V1R10, a new type of health check was introduced to exploit the IBM Health Checker for z/OS framework. *Migration health checks* help you determine the appropriateness of various migration actions:

- ▶ *Before* you migrate to the new z/OS release, you can use these new checks to assist with your migration planning.
- ▶ *After* you migrate, you can rerun these checks to verify that the migration actions were successfully performed.

These migration checks only report on the applicability of the specific migration actions on your current active system; they do not make any changes to the system.

IBM provides a rich set of checks. But in addition, other vendors, customers, and consultants can write and add their own check routines to IBM Health Checker for z/OS as shown in Figure 2-4. By utilizing the ability of these various components and products to detect potential problems, IBM Health Checker for z/OS allows you to avoid future problems by alerting you of their existence.

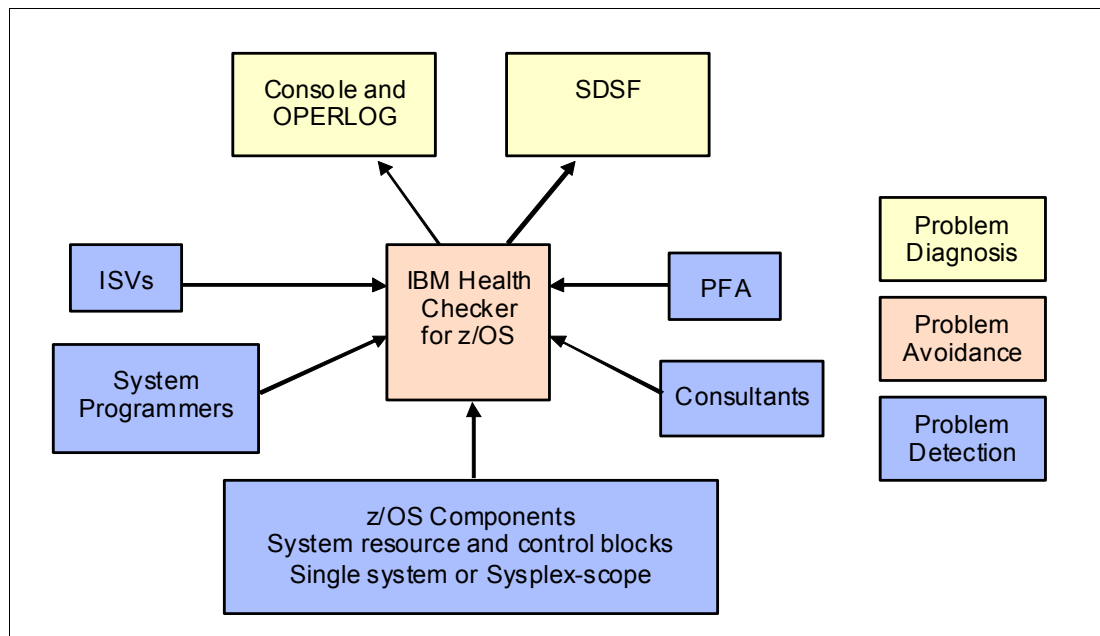


Figure 2-4 IBM Health Checker for z/OS

Details of IBM Health Checker for z/OS are well documented and not repeated here. For more information, see the IBM Redbooks document *Exploiting the IBM Health Checker for z/OS Infrastructure*, REDP-4590 and *IBM Health Checker for z/OS*, SA22-7994.

2.3.1 Getting the most out of IBM Health Checker for z/OS

The following tips explain how to use IBM Health Checker for z/OS to optimize your availability:

- ▶ Always start IBM Health Checker for z/OS automatically at IPL.
- ▶ Do not ignore the results of the migration checks.
- ▶ Investigate exceptions and take appropriate action. For example, do not simply change a configuration value without knowing why it should be changed.
 - That said, the first time you start the Health Checker, you might get a large number of exceptions. The goal should be to eventually remove all exceptions by fixing the condition *or* by tuning the check so that it more accurately reflects the normal behavior of *your* system.

If the check cannot be changed to avoid the exception and you are sure that skipping this check will not expose system problems, you can deactivate it.

In either case, your objective should be that no exceptions are detected during normal operation.

- ▶ After your systems run with no exceptions, you are in an ideal position to know that a health check exception indicates something is abnormal.
- ▶ Automate exceptions issued by IBM Health Checker for z/OS:
 - The messages produced by each check are documented together with the check in *IBM Health Checker for z/OS, SA22-7994*. Because a given check might vary in significance from one enterprise to another (or even one system to another), you can use the WTOTYPE and SEVERITY parameters for each health check to control how exceptions are presented.
- ▶ In an IBM GDPS® environment, the recommendations provided by GDPS might not be the same as those provided by z/OS. If there is a clash, use the value provided by GDPS value.

2.4 Runtime Diagnostics

What if a problem occurs on your system? Do you fear getting that phone call saying that something is wrong and you have no idea where to start looking? Is there a “bridge call” in your future and do you need an initial check of the system before attending it?

Starting with z/OS 1.12, the Runtime Diagnostics function is available to reduce the time needed to analyze and diagnose a problem and to reduce the experience level needed by the system operator. It provides timely, comprehensive diagnostics at a critical time period and suggests next steps for you to take to resolve the problem quickly.

Runtime Diagnostics was created as a joint design effort between IBM development and service organizations.

- ▶ IBM service personnel provided the expertise in knowing what types of symptoms they most often saw in diagnostic data, and the types of data that are most helpful when investigating system failures.
- ▶ IBM development personnel provided the expertise in knowing what types of symptom data component experts need when diagnosing problems, and the knowledge for creating this type of service tool within z/OS.

The goal of Runtime Diagnostics is to diagnose a failure in a timely manner and mimic the types of investigation that an experienced system programmer would do to help isolate the problem as quickly as possible.

- ▶ It looks for specific evidence of soft failures at the current point in time, with a goal of doing so in 60 seconds or less.
- ▶ It also provides recommendations for the next steps that the system programmer should take in the analysis, such as what to investigate further. However, it does not take any corrective action itself, because many of the recommended actions can be disruptive. For example, Runtime Diagnostics might recommend that jobs need to be cancelled.

As shown in Figure 2-5, Runtime Diagnostics obtains its data from OPERLOG, system control blocks, and system APIs. It performs no processing until explicitly invoked, and has minimal dependencies on system services when performing its diagnosis.

The Runtime Diagnostics output is called an *event*. A Runtime Diagnostics event is simply a multiline WTO that describes the event and the recommended actions to take. If you want the output to be directed to a sequential data set, you can do so by editing the HZR procedure that is shipped in SYS1.PROCLIB and modifying the HZROUT DD statement to point at a data set instead of DD DUMMY.

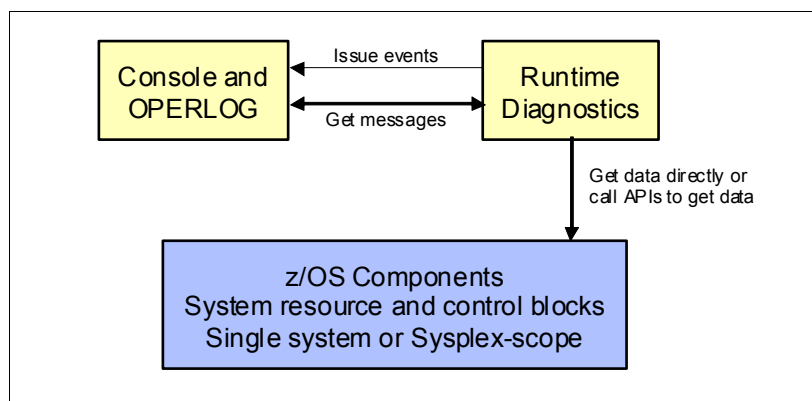


Figure 2-5 Runtime Diagnostics input and output

There are three categories of Runtime Diagnostics symptoms, all of which are shown in Figure 2-6. Runtime Diagnostics identifies problems in the following areas:

- ▶ Specific components through OPERLOG messages that are related to the component
- ▶ Global resource contention issues with enqueues, GRS-managed latches, and z/OS UNIX file system latches
- ▶ Address space execution issues by identifying high CPU usage, local lock suspension issues, and TCB loops.

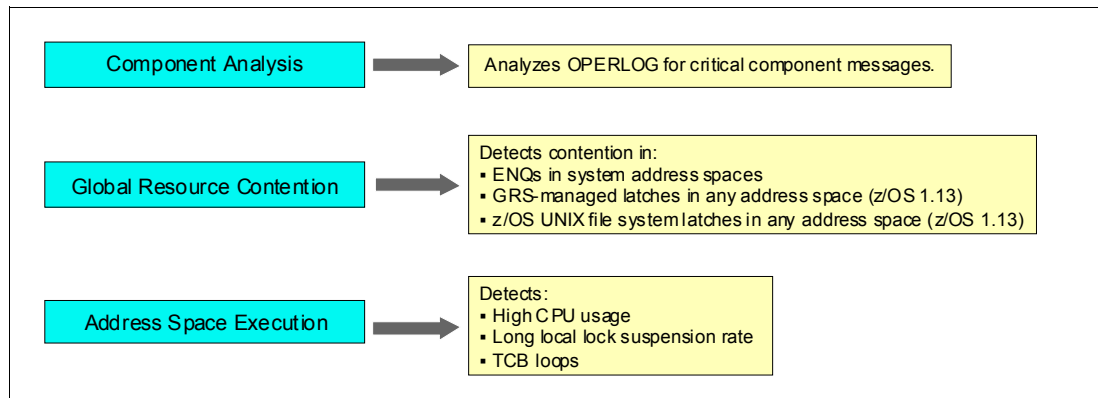


Figure 2-6 Runtime Diagnostics detection types

2.4.1 Invoking Runtime Diagnostics

Depending on your release, there are differences in how you invoke Runtime Diagnostics:

- ▶ In z/OS 1.12, Runtime Diagnostics was a started task that you started when needed using the **start hzr,sub=mstr** command. When it completed its analysis, the task would end.
- ▶ In z/OS 1.13, Runtime Diagnostics became a long-running started task that is started with the same command used in z/OS 1.12. However, now it does not end until you issue a **p hzr** command. In z/OS 1.13, the Runtime Diagnostics started task will remain started, but will not do anything until you request it to run its diagnostics using the **modify** command: **f hzr,analyze**.

The reason that it is a long-running started task in z/OS 1.13 is because it needs to be available for PFA to invoke if PFA thinks a metric might be abnormally low, and to avoid startup and shutdown resource thrashing. The integration between Runtime Diagnostics and PFA is described in 2.5.8, “PFA and Runtime Diagnostics integration” on page 63.

If you want to target a system other than the home system for critical message detection and enqueue contention checking, you can use the **SYSNAME** parameter. In z/OS 1.12, this parameter is specified on the **start** command. In z/OS 1.13, this parameter is specified on the **modify** command.

Even though Runtime Diagnostics was first made available in z/OS 1.12, the targeted system can be at z/OS 1.11 or later and the diagnostics are still performed.

There is also a **DEBUG** parameter that takes dumps when different types of events are found, or when they are not found and you believe they should be. This parameter is specified on the **start** command in z/OS 1.12 and on the **modify** command in z/OS 1.13. It should only be used when the IBM service organization requests this information.

Runtime Diagnostics is not an automated process and is not a system monitor. Its intent is to be a quick, on-demand, diagnostics tool that you use when you believe a soft failure is occurring.

If you choose to automate calls to Runtime Diagnostics, such as invoking it every hour, it is likely that events will be issued even though you are not experiencing soft failures. Some of the events also occur when the system is behaving normally, but they are not a real issue unless you have noticed that the system is having problems. These events are probably not indicative of a system problem at that time and you might waste time investigating these

events when no system problem is occurring. This point is especially true for the address space execution and global resource contention categories.

For more information about starting and controlling Runtime Diagnostics, refer to *z/OS Problem Management*, G325-2564.

2.4.2 Component analysis

Runtime Diagnostics searches OPERLOG for any occurrences of an IBM-defined list of messages that are deemed to be critical to system availability. These messages were identified by z/OS component experts from both IBM service and development as the types that when found, truly indicate a potential system problem. However, due to the volume of messages in the OPERLOG, they are difficult to detect because their occurrence is rare and requires an experienced system programmer to know which ones are critical. If any of these messages are found, an event is created.

Runtime Diagnostics looks back for one hour in the OPERLOG (if that much data is available). For some critical component messages detected by Runtime Diagnostics, additional diagnostics are done. For example, some related messages are grouped into a single event. Runtime Diagnostics also recognizes when a critical situation has been relieved. In some cases, it only shows the last message if a critical message for the same resource name is repeated with some frequency.

Do not expect to always find the full text of the message in the event. Rather, a summarized version that includes the message identifier, up to five lines of message text, and the recommended actions are created in the event. A list of the messages that Runtime Diagnostics detects can be found in *z/OS Problem Management*, G325-2564.

Figure 2-7 shows an example of the Runtime Diagnostics output for a critical message event.

```
EVENT 02: HIGH - CF          - SYSTEM: SY1      2011/02/15 - 14:47:03
IXC585E STRUCTURE LIST01 IN COUPLING FACILITY TESTCFN,
PHYSICAL STRUCTURE VERSION C7565A8D E48F6410,
IS AT OR ABOVE STRUCTURE FULL MONITORING THRESHOLD OF 80%.
ENTRIES:  IN-USE:           491 TOTAL:           583, 84% FULL
ELEMENTS: IN-USE:           508 TOTAL:          1167, 43% FULL
      ERROR: INDICATED STRUCTURE IS APPROACHING FULL MONITORING THRESHOLD.
      ACTION: D XCF,STR,STRNAME=strname TO GET STRUCTURE INFORMATION.
      ACTION: INCREASE STRUCTURE SIZE OR TAKE ACTION AGAINST APPLICATION.
```

Figure 2-7 Runtime Diagnostics critical message event

Critical message detection is performed for the system for which Runtime Diagnostics was invoked, even when SYSNAME is not the system on which the command was run. That is, you can target any system in your sysplex that is at z/OS 1.11 or above and the critical message detection is performed for it if OPERLOG is available.

To minimize the CPU cost of scanning potentially millions of messages in the OPERLOG, Runtime Diagnostics only searches for messages that were issued on the system for which it was invoked. That is, you cannot use Runtime Diagnostics to search the OPERLOG for all systems in the sysplex with one command. To have critical message detection performed for all systems in the sysplex, the command would have to be issued for each system separately.

Tips for critical message detection:

- ▶ OPERLOG must be active.
- ▶ The list of messages for which OPERLOG is searched is a hardcoded list defined by IBM and cannot be modified.
- ▶ OPERLOG is searched back one hour from the current time (if available).
- ▶ The Runtime Diagnostics event contains a summary of the message and the recommended actions.
- ▶ Some messages are grouped into one event that describes a single problem symptom.
- ▶ The detection is performed on the system specified in the SYSNAME parameter. If SYSNAME was not specified, the detection will be performed on the system that the command was issued on.

For a list of the messages for which Runtime Diagnostics creates events, refer to *z/OS Problem Management*, G325-2564.

2.4.3 Global resource contention

Runtime Diagnostics detects three types of global resource contention, as explained here.

Enqueue contention for system address spaces

This type of event is created when a system address space is waiting for an enqueue for more than five seconds.

Runtime Diagnostics has a hardcoded list of system address spaces for which enqueue contention could mean that a system failure is occurring. You cannot modify this list of address spaces.

The checking performed by Runtime Diagnostics to detect enqueue contention is equivalent to invoking the **D GRS,AN,WAITER** command. The event is created when a *waiter* is found that has been waiting for more than five seconds. No event is created for blockers that exist on this system because the blocker is listed in the event along with the waiter's information.

Figure 2-8 shows a sample enqueue contention event. Notice that the event shows both the waiter and the blocker ASID, job name, and the system name, along with other diagnostic information.

```

f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
REQ: 004 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 13:51:32
INTERVAL: 60 MINUTES
EVENTS:
  FOUND: 04 - PRIORITIES: HIGH:04 MED:00 LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01 ENQ:01 LOCK:01
-----
EVENT 01: HIGH - ENQ              - SYSTEM: SY1      2010/12/21 - 13:51:32
ENQ WAITER - ASID:0001 - JOBNAME:MASTER - SYSTEM:SY1
ENQ BLOCKER - ASID:0001 - JOBNAME:MASTER - SYSTEM:SY2
QNAME: TESTENQ
RNAME: TESTOFAVERYVERYVERYVERYLOOOOOOOOOOOOOOOOOOOOONGRNAMEI234567...
  ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
  ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
  ACTION: ASIDS.

```

Figure 2-8 Runtime Diagnostics enqueue contention event

Enqueue contention checking is performed for the system for which Runtime Diagnostics was invoked. That is, you can target any system in your sysplex that is at z/OS 1.11 or above.

Tips for enqueue contention detection:

- ▶ The address space must be in the list of system address spaces as defined by Runtime Diagnostics. This list cannot be modified.
- ▶ The address space must be waiting for an enqueue for at least five seconds.
- ▶ The detection is performed on the system specified in the SYSNAME parameter, or on the home system if SYSNAME was not specified.

GRS-managed latch contention

Contention checking for GRS-managed latches is available starting with z/OS 1.13. A GRS latch contention event is created when any address space has been waiting for a latch for more than five minutes.

Latch contention information is obtained directly from GRS by the use of GRS APIs. z/OS UNIX System Services file system latches are omitted from this category because they have their own category (this is described in “z/OS UNIX System Services file system latch contention” on page 38).

Figure 2-9 shows an example of a GRS latch contention event. The “top waiter” is the address space that has been waiting the longest for the latch set. The recommended action is circled.

```

F HZR,ANALYZE
HZR0200I RUNTIME DIAGNOSTICS RESULT 692
SUMMARY: SUCCESS
REQ: 002 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 14:32:01
INTERVAL: 60 MINUTES
EVENTS:
  FOUND: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
  TYPES: LATCH:02
-----
EVENT 01: HIGH - LATCH - SYSTEM: SY1      2010/12/21 - 14:32:01
LATCH SET NAME: SYSTEST.LATCH TESTSET
LATCH NUMBER:3      CASID:0039 CJOBNAME:TSTLATCH
TOP WAITER - ASID:0039 - JOBNAME:TSTLATCH - TCB/WEB:004E2A70
TOP BLOCKER- ASID:0039 - JOBNAME:TSTLATCH - TCB/WEB:004FF028
ERROR: ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
ACTION: D GRS,AN,LATCH,DEP,CASID=0039,LAT=(SYSTEST.L*,3),DET
ACTION: TO ANALYZE THE LATCH DEPENDENCIES. USE YOUR SOFTWARE
ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.

```

Figure 2-9 Runtime Diagnostics GRS latch contention event

Tips for GRS latch contention detection:

- ▶ This function is available starting with z/OS 1.13.
- ▶ The event is created for any address space waiting for a latch for more than five minutes.
- ▶ Contention in z/OS UNIX System Services file system latches are handled separately.
- ▶ The top waiter is the address space that has been waiting for the latch set the longest.

z/OS UNIX System Services file system latch contention

Also new in z/OS 1.13 is z/OS UNIX System Services file system latch contention checking. An OMVS event is created when threads have been waiting for more than five minutes in z/OS UNIX System Services.

These types of latches are separated from the GRS-managed latch contention events because additional processing is performed to determine the specific source of the contention.

The normal action to take when this type of event occurs is to issue a **D OMVS,W,A** command to determine the ASID and the job names of the waiters to investigate (this is circled in the example event shown in Figure 2-10).

```
F HZR,ANALYZE
HZR0200I RUNTIME DIAGNOSTICS RESULT 692
SUMMARY: SUCCESS
REQ: 009 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 14:24:29
INTERVAL: 60 MINUTES
EVENTS:
FOUND: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
TYPES: OMVS:01
TYPES: LOCK:01
-----
EVENT 01: HIGH - OMVS - SYSTEM: SY1      2010/12/21 - 14:24:29
ASID:000E - JOBNAME:OMVS
MOUNT LATCH WAITERS: 1
FILE SYSTEM LATCH WAITERS: 0
XSYS AND OTHER THREADS WAITING FOR z/OS UNIX: 1
ERROR: z/OS UNIX MIGHT HAVE FILE SYSTEM LATCH CONTENTION.
ACTION: D OMVS,W,A TO INVESTIGATE z/OS UNIX FILE SYSTEM LATCH
ACTION: CONTENTION, ACTIVITY AND WAITING THREADS. USE YOUR SOFTWARE
ACTION: MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
```

Figure 2-10 Runtime Diagnostics z/OS UNIX Latch contention event

Tips for z/OS UNIX System Services file system latch contention detection:

- ▶ This function is available starting with z/OS 1.13.
- ▶ The event is created when a thread in any address space is waiting for a UNIX System Services latch for more than five minutes.
- ▶ The normal action to take is to issue the **D OMVS,W,A** command.

2.4.4 Address space execution

Runtime Diagnostics provides three types of checking to determine whether execution within any address space is behaving abnormally such that it might cause a soft failure.

CPU analysis

The Runtime Diagnostics CPU analysis is a point-in-time inspection to determine whether any task is using more than 95 percent of the capacity of a single CPU. This level of CPU consumption might indicate that the address space is in a loop.

The analysis is performed by taking two snapshots one second apart to calculate the percentage of CPU used based on the capacity of a single CPU within the LPAR. It is possible for the usage reported to be greater than 100 percent if the address space has multiple TCBs and several of the TCBs are individually using a high percentage of the capacity of a single CPU.

When a high CPU event and a loop event occur for the same job, there is high probability that the task in the job is in a loop in which case the normal corrective action is to cancel the job.

Figure 2-11 shows an example of a high CPU event created by Runtime Diagnostics.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
REQ: 004 TARGET SYSTEM: SY1      HOME:      2010/12/21 13:51:32
INTERVAL: 60 MINUTES
EVENTS:
  FOUND: 04- PRIORITIES: HIGH:04  MED:00  LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01 ENQ:01 LOCK:01
-----
EVENT 02: HIGH- HIGHCPU      - SYSTEM: SY1      2010/12/21- 13:51:33
ASID CPU RATE:99%  ASID:002E  JOBNAME:IBMUSERX
STEPNAME:STEP1     PROCSTEP:      JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 11:22:51
  ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
  ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 2-11 Runtime Diagnostics High CPU event

Tips for CPU analysis:

- ▶ Two snapshots are taken one second apart.
- ▶ If any task is using more than 95 percent of a single CPU, a high CPU event is created.
- ▶ The event might show usage greater than 100 percent.
- ▶ A high CPU event combined with a loop event for the same job usually indicates the job is truly in a loop.

Local lock suspension

Local lock suspension analysis is a point-in-time check of local lock suspension for any address space.

Runtime Diagnostics calculates the amount of time an address space is suspended waiting for the local lock. If an address space is suspended more than 50 percent of the time waiting for a local lock, Runtime Diagnostics creates a lock event for that address space.

Figure 2-12 shows an example of a local lock suspension event.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
REQ: 004 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 13:51:32
INTERVAL: 60 MINUTES
EVENTS:
  FOUND: 04 - PRIORITIES: HIGH:04 MED:00 LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01 ENQ:01 LOCK:01
-----
EVENT 04: HIGH - LOCK          - SYSTEM: SY1      2010/12/21 - 13:51:33
HIGH LOCAL LOCK SUSPENSION RATE - ASID:000A JOBNAME:WLM
STEPNAME:WLM      PROCSTEP:IEFPROC  JOBID:+++++++ USERID:+++++++
JOBSTART:2010/12/21 - 11:15:08
ERROR: ADDRESS SPACE HAS HIGH LOCAL LOCK SUSPENSION RATE.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
-----
```

Figure 2-12 Runtime Diagnostics local lock suspension event

Tip for local lock suspension analysis:

- If an address space is suspended for more than 50 percent of the time, a high local lock suspension rate event is created.

TCB loop detection

Runtime Diagnostics looks through all tasks in all address spaces to determine whether a task appears to be looping.

Runtime Diagnostics does this by examining various system information for indicators of consistent repetitive activity that typically appears when a task is in a loop.

When a high CPU event and a loop event occur for the same job, there is a high probability that the task in the job is in a loop, in which case the normal corrective action is to cancel the job.

Figure 2-13 shows an example of a high CPU event and a loop event for the same job.

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
REQ: 004 TARGET SYSTEM: SY1      HOME:SY1      2010/12/21 - 13:51:32
INTERVAL: 60 MINUTES
EVENTS:
  FOUND: 04 - PRIORITIES: HIGH:04 MED:00 LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01 ENQ:01 LOCK:01
-----
EVENT 02: HIGH - HIGHCPU - SYSTEM: SY1      2010/12/21 - 13:51:33
ASID CPU RATE:99% ASID:002E JOBNAME:IBMUSERX
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
  ERROR: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MIGHT BE LOOPING.
  ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
-----
EVENT 03: HIGH - LOOP - SYSTEM: SY1      2010/12/21 - 13:51:14
ASID:002E JOBNAME:IBMUSERX TCB:004FF1C0
STEPNAME:STEP1 PROCSTEP: JOBID:JOB00045 USERID:IBMUSER
JOBSTART:2010/12/21 - 11:22:51
  ERROR: ADDRESS SPACE MIGHT BE IN A LOOP.
  ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
```

Figure 2-13 Runtime Diagnostics High CPU event and loop event

Tips for TCB loop detection:

- ▶ The analysis is done for all tasks in all address spaces.
- ▶ A high CPU event combined with a loop event for the same job usually indicates that the job is truly in a loop.

2.5 Predictive Failure Analysis

Most experienced IT professionals have probably encountered many examples of soft failures. Consider the following example: a common transaction that rarely fails hits a glitch that causes it to fail more frequently. The recovery code in the underlying component is normally successful and applications continue to work even though the problem is occurring. The system programmer has not noticed or has not been notified that something is amiss.

By the time the system programmer is notified, there have been multiple failing transactions and the system has become heavily loaded. Recovery continues to occur, but it slows down the transaction manager to the point that completely unrelated transactions start experiencing time-outs.

Although this example is hypothetical, it shows that if a soft failure like the one described occurs, it needs to have some external symptoms before the system programmer can start to take corrective action.

2.5.1 Predictive Failure Analysis overview

The goal of predictive analysis and early detection is to notify the system programmer as soon as the system becomes aware that a problem is occurring. Predictive Failure Analysis

(PFA) detects problems before they are visible externally by checking resources and metrics at different layers of the software stack that can indicate that resource exhaustion, damage to address spaces, or damage to the system could be occurring by comparing existing metric values to trends for the LPAR.

PFA is not intended to find problems that will bring the system down the instant they start. Rather, it can detect potential problems on a human-time scale.

Various of the metrics that PFA uses in its analysis are closer to the hardware, and others are closer to the application. It is easier for PFA to detect an error that is close to the hardware. The closer the metric is to the application, the more difficult it is to determine whether this is really a soft failure or simply an anomaly due to a workload change.

Figure 2-14 on page 43 shows how PFA fits into z/OS:

- PFA obtains its data from z/OS control blocks and published programming interfaces.
- It collaborates with Runtime Diagnostics when it thinks that some metric has an abnormally low value to determine whether this really represents a problem. This is a new capability that was delivered in z/OS 1.13. For more information about this capability, refer to “PFA and Runtime Diagnostics integration” on page 63.
- Finally, it sends the results to IBM Health Checker for z/OS, which issues the message as a WTO if necessary and makes the output viewable in SDSF.

Details about PFA processing and integration are provided in the following sections. All documentation related to PFA (including installation steps, management, and usage) is contained in *z/OS Problem Management*, G325-2564.

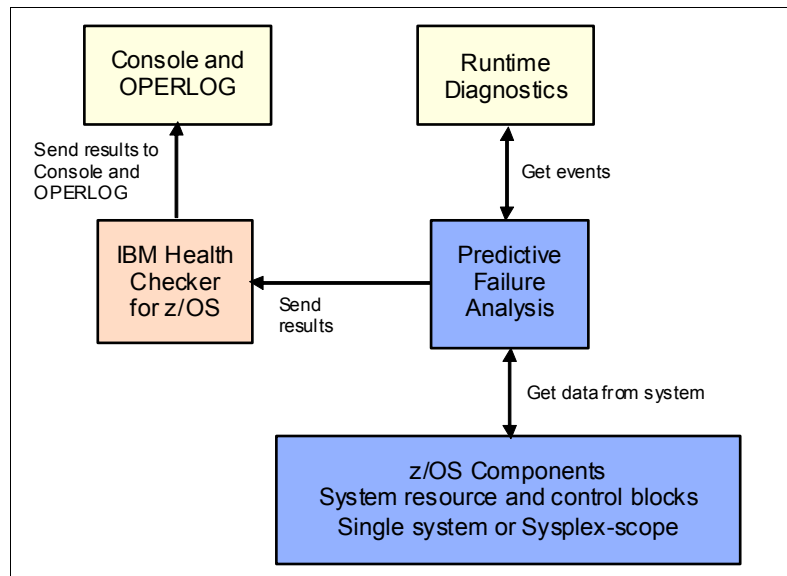


Figure 2-14 PFA input and output

2.5.2 Types of abnormal behavior detected

After the review of many soft failures on clients' systems, it was determined that there are four common reasons why a system has stopped functioning:

- Damaged address space or system

The indication of a damaged system is typically when there are recurring or recursive errors somewhere in the software stack.

- **Serialization problems**

Serialization problems are most often caused by priority inversion, classic deadlocks, and owner-gone scenarios.

- **Physical or software resource exhaustion**

Exhaustion of a shared and finite system resource.

- **Indeterminate or unexpected states**

PFA focuses on the damaged address space or system and the resource exhaustion categories.

PFA uses historical data together with machine learning and mathematical modeling to *detect* abnormal behavior and the potential causes of this abnormal behavior. It also *predicts* whether exhaustion of certain finite resources will occur if the current trend continues.

PFA's objective is to convert soft failures to correctable incidents by alerting you as soon as it detects that a problem might be occurring or predicts that one will occur in the future.

PFA does not use hardcoded thresholds to determine abnormal behavior. Rather, it collects the data from the system and models it to determine what is normal for that system and what kinds of resource usage trends are occurring. It then uses advanced algorithms in conjunction with configurable values for sensitivity to determine whether an abnormal condition exists or whether the current trend indicates future exhaustion.

There are three types of abnormal behavior detection that PFA's algorithms incorporate:

Future prediction	This processing performs trend analysis and models the behavior into the <i>future</i> to predict whether the current trend will exhaust a common resource.
Expected value	This processing performs trend analysis and models the behavior to determine what value should be expected at the <i>current time</i> to determine whether an abnormal condition is occurring.
Expected rate	This processing performs trend analysis and models the behavior to determine whether the current rate, when <i>compared to rates for multiple time periods</i> , indicates that an abnormal condition is occurring. The rate is often calculated by normalizing the value of the metric being analyzed by the CPU being used. By using a normalized rate and comparing against multiple time period predictions, normal workload changes do not appear as abnormal conditions.

For both the expected value and the expected rate types of predictions, PFA clusters the historical data so that trends within the data can be identified. It then determines which trend is currently active and uses the prediction for that trend in its comparisons.

Tip: PFA does not take the time of day or the day of the week into account when comparing current values to projections of what the current values should be. However, by using CPU consumption to normalize its rate metrics, it effectively takes into account whether the system is busy or quiet. In practice, this has been found to be effective and negates the need to take time or day into account when creating its model of system behavior.

2.5.3 PFA and IBM Health Checker for z/OS integration

It is important to understand the relationship between IBM Health Checker for z/OS and PFA so that you can fully benefit from IBM Health Checker for z/OS and PFA integration.

PFA exploits the remote check feature of the IBM Health Checker for z/OS framework to control the scheduling of the PFA checks and to present the results back to the operator. Therefore, the commands, SDSF health check interface, and reporting mechanism available through IBM Health Checker for z/OS are fully usable by the PFA checks.

Figure 2-15 on page 45 shows the flow of PFA processing to and from IBM Health Checker for z/OS. Each step is described separately in 2.5.4, “PFA processing” on page 46.

Information about the IBM Health Checker for z/OS can be found in the IBM Redpaper™ *Exploiting the IBM Health Checker for z/OS Infrastructure*, REDP-4590, and in *IBM Health Checker for z/OS*, SA22-7994.

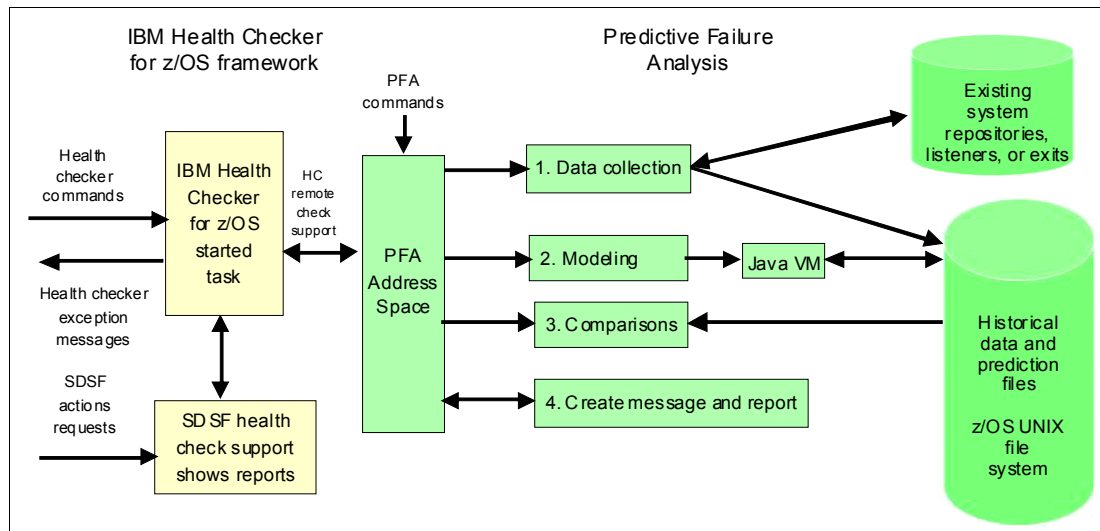


Figure 2-15 PFA and IBM Health Checker for z/OS integration

If IBM Health Checker for z/OS is not active prior to PFA starting, PFA collects data and creates predictions as usual, but waits for IBM Health Checker for z/OS to start before performing comparisons or issuing results. This is because those functions are dependent on IBM Health Checker for z/OS.

If IBM Health Checker for z/OS is not available when it is started, PFA uses the default configuration values such as COLLECTINT and MODELINT. These values can be overridden by parameters you specify in IBM Health Checker for z/OS, but that depends on IBM Health Checker for z/OS being available. An example of the HZSPRMxx statements that you use to change the COLLECTINT are shown in Example 2-1.

Example 2-1 Sample HZSPRMxx statements to modify a PFA check

```
/* */
/*-----*/
/*      Sample PFA override      */
/*-----*/
/* */
ADD POLICY UPDATE CHECK(IBM PFA,PFA_COMMON_STORAGE_USAGE)
ACTIVE
```

```
PARMS=('COLLECTINT(10)')  
DATE(20120827)  
REASON('Update collection interval')
```

You can use the health check commands both to display and modify the PFA checks. Unlike other health checks, the PFA checks allow you to modify their parameters individually and accumulate the changes rather than requiring *all* parameters to be specified when modifying a check. This feature allows you to change one parameter without needing to remember the settings for all the others.

For example, to change only the STDDEV parameter of the JES spool usage check and set it to 6, issue the following command:

```
f hzsproc,update,check(ibmpfa,pfa_j*),parm('stddev(6)')
```

Display all parameters and obtain status of PFA checks: IBM Health Checker for z/OS is unaware of the cumulative nature of the PFA checks' parameters. Therefore, if you display a PFA check using the IBM Health Checker for z/OS modify command to display details, or you look at the parameters in the reports, only the parameters that were last modified are displayed.

To properly display all the parameters and to obtain the status of the PFA checks, use the PFA **MODIFY** command with the display details option such as shown:

```
f pfa,display,check(pfa_*),detail
```

Variations of the **MODIFY PFA** command and a description of the parameters can be found in *z/OS Problem Management*, G325-2564.

The results of PFA's comparisons are sent to IBM Health Checker for z/OS, which writes the report. If an exception occurs, a WTO is issued by default. You can configure the type of message issued when an exception occurs by changing the check's **SEVERITY** or **WTOTYPE** in IBM Health Checker for z/OS.

2.5.4 PFA processing

PFA contains check-specific code that collects the data for an individual check; creates a prediction; and compares the current values to an internal value created by PFA using the current predictions, domain knowledge, and customer-configurable parameters that adjust the sensitivity of the comparisons.

Depending on the result of the comparison, PFA uses the remote check support with IBM Health Checker for z/OS to issue an exception or informational message and to create a report in SDSF. Some of these functions are written in Java and therefore are eligible to run on a zAAP if the system is configured with one (see 2.5.9, "Achieving maximum benefit from PFA" on page 65 for more information).

Data collection

Data is collected for each check by interrogating system control blocks or using externally available interfaces such as APIs, ENF listeners, and dynamic exits. Data collection (shown in Figure 2-16 on page 47) occurs as follows:

1. Data collection occurs asynchronously on an interval that you can configure (although it is not anticipated that changing it will be necessary). For example, the default value for the data collection function for a check might be to collect data every fifteen minutes. This parameter is called **COLLECTINT**.

2. The data is stored in files in the z/OS UNIX file system. Data that is deemed by PFA to be too old for use in making new predictions is automatically deleted by PFA.

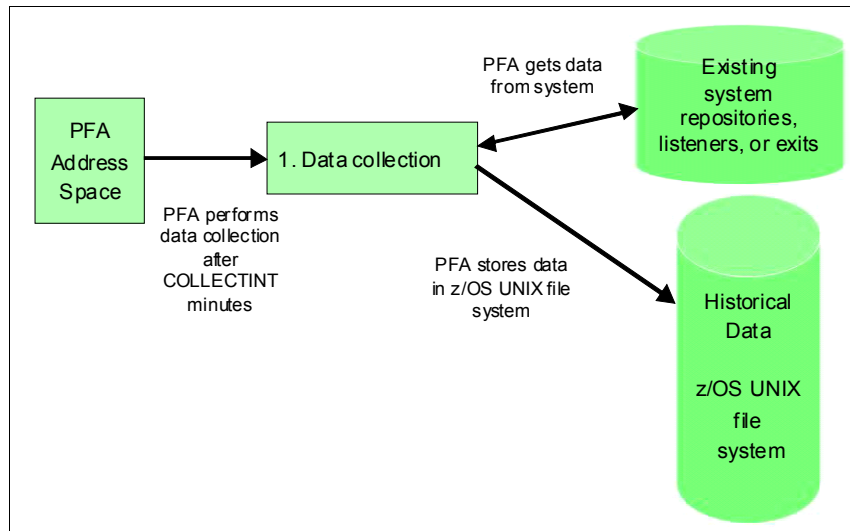


Figure 2-16 PFA data collection processing

Modeling

The data collected in the first step is modeled asynchronously to create predictions for the current trends.

Depending on the type of check, the result of the modeling might be a value that is predicted to occur in the future based on the current trends, or it might be a value that is expected at this point in time based on the current trends.

When the prediction is an expected value, PFA clusters the data into appropriate parts based on the historical values to determine workload differences within the data and creates predictions for each cluster. It then chooses which cluster's prediction to use in the comparisons.

For expected rate comparisons over time, PFA creates predictions for multiple time ranges and then determines which time range predictions should be applied during the comparison. By performing this clustering and creating multiple predictions, PFA can better detect workload changes which otherwise will lead to unwarranted exceptions.

Modeling (shown in Figure 2-17 on page 48) occurs as follows:

- Modeling occurs asynchronously on an interval that you can control (although it is not anticipated that changing it will be necessary). For example, the default value for the modeling interval for all checks (starting in z/OS 1.12) is to run every twelve hours. This is controlled by the MODELINT parameter.
- When PFA starts, the first prediction is created after the check has determined that there is enough data to make a prediction, rather than necessarily waiting for the MODELINT number of minutes.
 - For some checks, the first prediction is created seven hours after PFA starts as follows:
If PFA started at IPL, the first hour's worth of data (when everything is in startup mode) is discarded so that it does not skew the prediction. If PFA started more than an hour after the IPL, data collection can begin immediately. This reduces the wait time for the first prediction from seven hours to six hours.

PFA then performs data collection for six hours (or the MODELINT number of minutes if it is less than six hours) and the first prediction is created. Subsequent predictions in a stable environment occur every twelve hours thereafter.

- For other checks, the first prediction is created thirteen hours after PFA starts.

The checks that first create a prediction thirteen hours after PFA starts are those that need a warm-up period to detect which jobs to track individually.

If PFA started at IPL, the first hour's worth of data is discarded so that it does not skew the results of the warm-up. If PFA started more than an hour after IPL, data collection for the warm-up can begin immediately, thereby reducing the wait time for the first prediction from thirteen hours to twelve hours.

PFA then collects data for six hours for the warm-up, after which PFA determines the address spaces with the highest rates to track individually.

After the address spaces are properly categorized, PFA collects data for six hours (or the MODELINT number of minutes if it is less than six hours) and the first prediction is created. Subsequent predictions in a stable environment are made every twelve hours thereafter.

- ▶ Modeling runs when it is determined that it is time to update the model with new predictions based on the configured value. It can also run dynamically when PFA determines that the model might be too old to make accurate comparisons because the trend is changing.
- ▶ PFA requires at least four successful collections before modeling is attempted.
- ▶ The results of the modeling are stored in files in the z/OS UNIX file system. PFA automatically deletes files when they are no longer required.

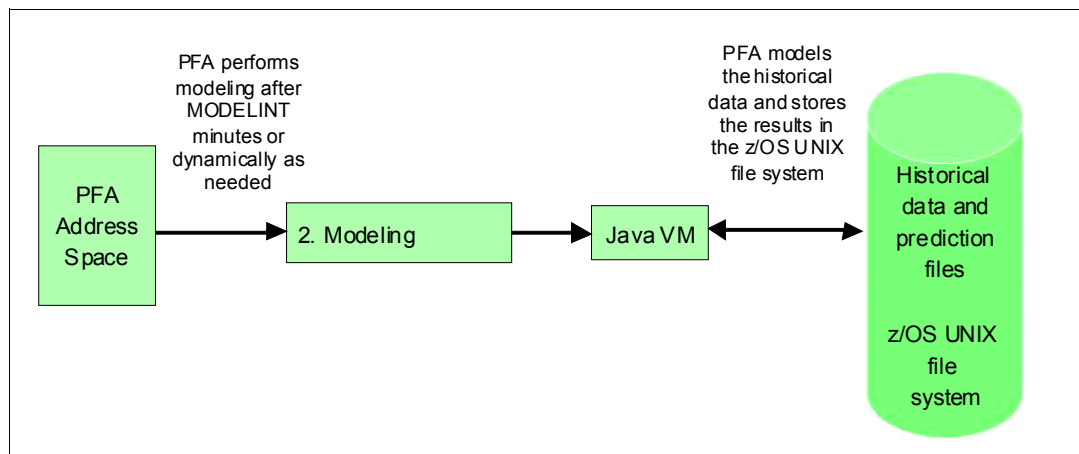


Figure 2-17 PFA modeling processing

Important points about PFA modeling:

- ▶ Modeling is done in Java and is zAAP eligible. It is strongly advised to offload PFA's Java processing to a zAAP.
- ▶ Even though the MODELINT value is configurable, you cannot force PFA to model at a specific time of day due to the fact that PFA models more frequently when your system is unstable to ensure that the model is current. When the system stabilizes, PFA resumes using the specified MODELINT value.

Comparisons

PFA performs the comparisons needed to determine whether an exception is to be issued. It compares what is currently occurring on the system to what was predicted by applying mathematical algorithms based on statistics and *z/OS domain knowledge* (information known about z/OS from experience) and by incorporating the user-defined parameters allowed for the check¹. These algorithms determine whether the current values are abnormal, becoming abnormal, or everything is working fine.

If PFA determines the trend is abnormal or becoming abnormal, the comparison step might cause the check to create a new prediction before the next scheduled modeling interval.

Comparison processing is shown in Figure 2-18.

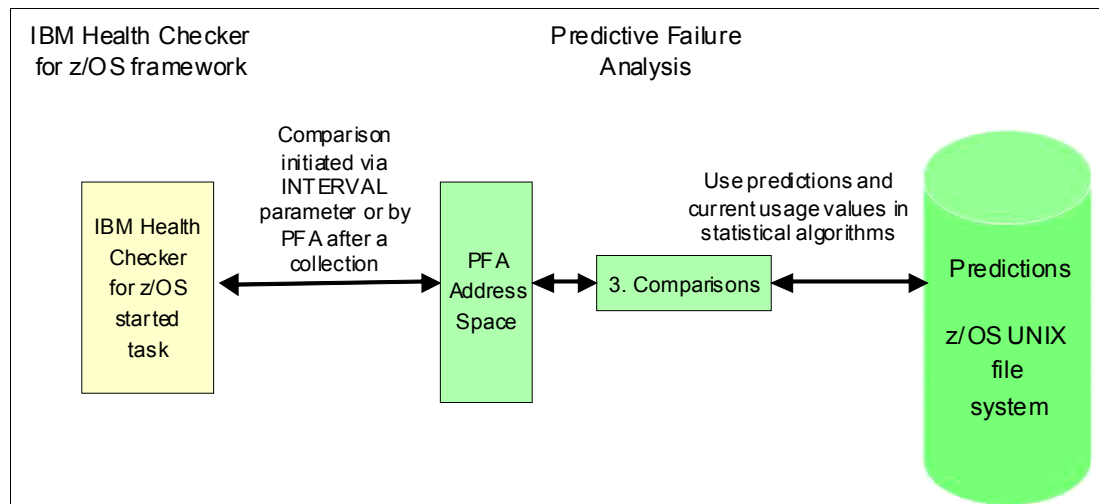


Figure 2-18 PFA comparison processing

For some checks, comparisons are initiated by IBM Health Checker for z/OS when the number of minutes specified in the INTERVAL parameter for the check is reached. For other checks, the comparisons are performed at the end of every collection rather than using the INTERVAL parameter in IBM Health Checker for z/OS.

Important points about PFA comparisons:

- ▶ If the value of the INTERVAL parameter in IBM Health Checker for z/OS is set to ONETIME for a PFA check, it is a check that runs after every collection. Do *not* change this setting. If you do, the comparison uses the data collected at the last collection rather than retrieving new current usage data.
- ▶ Even though IBM Health Checker for z/OS allows you to run a health check on demand by using IBM Health Checker for z/OS interfaces, it is best to allow PFA to run the checks when PFA determines it is time for the comparisons to be performed.

When a check is run manually that has the INTERVAL parameter set to ONETIME, message HZS0400I is issued with message AIR023I embedded. You can display the current INTERVAL value by using the **SDSF CK** command and scrolling to the right to the column titled "INTERVAL."

¹ See 2.5.9, "Achieving maximum benefit from PFA" on page 65 for information about the user overrides that are available.

For each check, configuration parameters are available that affect the results of the check by changing the sensitivity of the comparisons. For further details about how to tune PFA comparison algorithms, see 2.5.9, “Achieving maximum benefit from PFA” on page 65.

Send results and alert the operator

For each comparison (regardless of whether the comparison resulted in an exception, or everything is working normally), PFA creates a report with the current data and predicted data. The report is only accessible by using the SDSF CK function and selecting the PFA check you are interested in.

When PFA issues an exception the following actions occur, as shown in Figure 2-19.

1. Details specific to the exception are provided in the report to aid in your investigation. For example, if PFA issued the exception for comparisons done for specific address spaces, only those address spaces are listed on the report.
2. IBM Health Checker for z/OS issues message HZS000na with the PFA message (AIRHnnnE) embedded in it based on the SEVERITY and WTOTYPE settings of the check. This message goes to the system log and therefore is visible to your automation software.
3. Future WTOs for the same exception are suppressed until new data is available.
4. A directory is created with the data needed for IBM service to investigate the exception in cases where you believe the exception was unwarranted. Some of this information might be useful in your investigation as well.

The data is written to a subdirectory of the PFA check’s directory and is named EXC_timestamp, where the time stamp contains the date and time the exception occurred. A maximum of 30 directories per check are kept. When more are required, the oldest directory is deleted.

5. Modeling might occur more frequently until the data stabilizes.

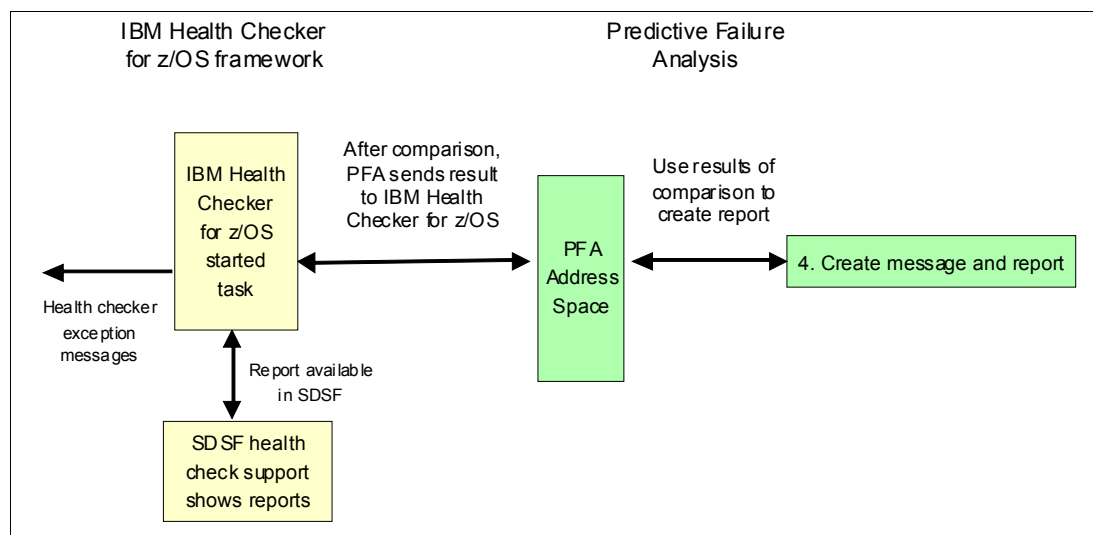


Figure 2-19 PFA sends results to IBM Health Checker for z/OS

2.5.5 PFA reports

Figure 2-20 shows an example of a PFA report for the Message Arrival Rate check. When applicable, PFA reports have the following three parts:

- Heading information

This information includes the status and configuration of the collections and the models.

- System-level information

This information includes the current values and predictions for the total system.

- Address space information

This information includes details for address spaces that PFA is tracking, address spaces for which the exception was issued, or address spaces that might be the cause of the problem.

The report is accompanied by IBM Health Checker for z/OS information and the message that was issued.

Message Arrival Rate Prediction Report						
Last successful model time : 04/05/2012 07:08:01						
Next model time : 04/05/2012 19:08:04						
Model interval : 720						
Last successful collection time : 04/05/2012 08:22:15						
Next collection time : 04/05/2012 08:37:16						
Collection interval : 15						
Message arrival rate						
at last collection interval : 83.52						
Prediction based on 1 hour of data : 98.27						
Prediction based on 24 hours of data: 85.98						
Prediction based on 7 days of data : 100.22						
Top persistent users:						
		Message		Predicted Message		
		Arrival		Arrival Rate		
Job	ASID	Rate	1 Hour	24 Hour	7 Day	
Name						
TRACKED1	001D	58.00	23.88	22.82	15.82	
TRACKED2	0028	11.00	0.34	11.11	12.11	
TRACKED3	0029	11.00	12.43	2.36	8.36	
...						

Figure 2-20 PFA Message Arrival Rate Prediction Report

2.5.6 PFA checks

This section summarizes the PFA health checks and provides details about their execution. Other information about PFA checks such as example reports, file descriptions, best practices, and so on can be found in *z/OS Problem Management*, G325-2564.

With z/OS 1.13, there are six PFA health checks. One check detects resource exhaustion. The other five checks detect a damaged address space or system. Three of those checks can also detect a hung address space or system. A summary of the checks, what each detects, the type of analysis each uses, and the release each is available is shown in Table 2-1 on page 52.

Table 2-1 PFA checks

Resource or metric	Causes detected	Type of analysis	Release
Common storage usage	Detects <i>resource exhaustion</i> in common storage by looking for spikes, leaks, and creeps.	Future prediction	1.10 SPE
LOGREC record arrival rate	Detects a <i>damaged address space or system</i> .	Expected rate	1.10 SPE
Console message arrival rate	Detects a <i>damaged or hung address space or system</i> based on an abnormal rate in the WTOs and WTORs issued, normalized by the amount of CPU being used.	Expected rate	1.11
SMF record arrival rate	Detects a <i>damaged or hung address space or system</i> based on an abnormal rate of SMF record arrivals, normalized by the amount of CPU being used.	Expected rate	1.12
JES spool usage	Detects a <i>damaged persistent address space</i> based on an abnormal increase in the number of track groups used.	Expected value	1.13
Enqueue request rate	Detects a <i>damaged or hung address space or system</i> based on an abnormal rate of enqueue requests, normalized by the amount of CPU being used.	Expected rate	1.13

Apply the PTF for APAR OA40065: If you are familiar with PFA, you might have noticed that the check for frames and slots usage (PFA_FRAMES_AND_SLOTS_USAGE) was omitted from Table 2-1. This check has been permanently removed from PFA with APAR OA40065 because it caused unwarranted exceptions that could not be avoided with the available mechanisms.

It is advisable to apply the PTF for APAR OA40065 as soon as possible. To remove the check from PFA prior to applying the PTF, use the following IBM Health Checker for z/OS command:

```
f hzsproc,delete,check(ibmpfa,pfa_f*)
```

PFA is not an address space monitor. Rather, PFA attempts to detect when abnormal activity on the system could lead to a system outage and alert the operator before the system crashes or hangs.

Notice in Table 2-1 that the only check looking for any type of *resource exhaustion* and using a future prediction is the one that checks for common storage exhaustion. Sometimes, clients are puzzled by the fact that the JES Spool Usage check does not detect exhaustion of track groups.

So, it is important to understand that the JES Spool Usage check is not looking at spool usage from the perspective of warning you when the spool starts to fill (JES2 already has messages that provide that information). Instead, it is looking for damaged persistent address spaces by detecting that an address space is using an abnormal number of track groups compared to what it normally uses. An address space that is increasing the number of track groups it uses at a rate significantly higher than it normally uses might be creating a large job log or might be in a loop, even if it is not exhausting the number of track groups available.

Common storage usage check

The PFA_COMMON_STORAGE_USAGE check is designed to detect future common storage exhaustion.

This check performs trend analysis to determine whether the current usage of common storage will exhaust common storage in the future. It does *not* detect when the common storage usage reaches a certain threshold, such as the percentage used compared to its available capacity.

For example, you might receive an exception for storage exhaustion even though only 53 percent of the available storage is currently being used. This is because PFA has determined that you will exhaust the storage in the *future*, based on your current trend. This early detection allows you to take action before your system starts behaving abnormally.

The example in Figure 2-21 shows an instance of how the current usage trend of SQA was fairly stable and was running at a fairly high percentage of used SQA compared to the capacity. However, the usage then started to increase such that if the new trend continued, SQA exhaustion would have occurred. The system overflows SQA into CSA eventually, but it was not yet doing so and the trend could have clearly indicated a problem on the system.

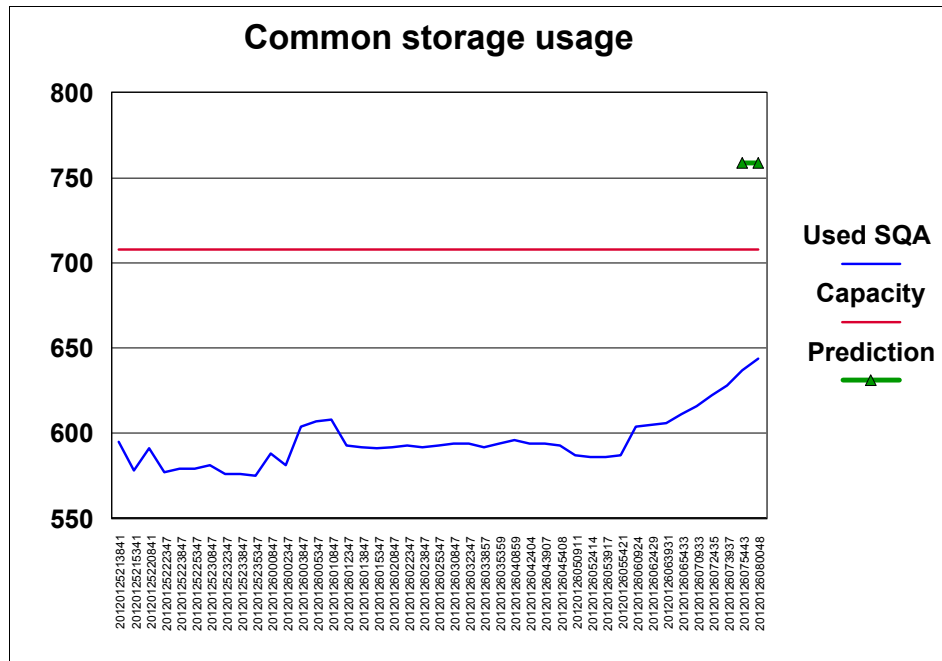


Figure 2-21 PFA Common Storage Usage Check example

In this example, the following sequence occurred:

1. The SQA capacity was 708 KB.
2. For about eight hours, the SQA usage hovered around 600 KB, which is about 85 percent of capacity.
3. Then, for about two hours, the trend started to gradually increase.
4. A new prediction was created after the last collection that detected, based on the current trend, that SQA usage would increase to 759 KB in the next six hours, which was well beyond the capacity.
5. Five minutes later, the current usage was continuing its upward trend and was then at 644 KB. At that point, the exception was issued.

Therefore, when usage was stable, even though it was rather high, the predictive trending indicated no problem would occur. However, when the usage started to steadily increase, the predictive trending indicated it would exceed the available capacity. The continued upward trend caused an alert to be issued to the operator.

PFA also applies z/OS domain knowledge in its comparison algorithms. That is, it knows at what usage level most systems tend to have soft failures for each storage location, and knows to issue exceptions for some locations earlier than others.

The common storage usage check also understands storage expansion from one location to another. If storage expands from SQA into CSA, it is included in the CSA usage, and predictions and comparisons for SQA are not performed. Similarly, when storage expands from ESQA into ECSA, it is included in the ECSA usage and predictions. Comparisons for ESQA are never performed (starting with APAR OA38279) because expansion into ECSA always occurs and such expansion does not cause system problems. This category is still tracked and modeled separately for your reference.

This check cannot detect exhaustion caused by fragmentation or rapid growth such as exhaustion that occurs within a collection interval or on a machine-time scale.

Also, this check does not monitor individual address spaces so it does not detect a storage leak in an individual address space unless that leak can result in storage exhaustion for the entire system.

When an exception occurs, it is not typically the address space that is using the most common storage that is the cause of the problem. Rather, it is the address space whose usage has increased the most recently.

Therefore, the exception report lists the top address spaces whose usage has increased the most in the last *hour* along with their current usage, and a prediction based on their trends (although comparisons are not performed using these values). If *SYSTEM* is listed, no attempt is made to determine the current usage due to performance considerations and UNAVAILABLE is displayed.

PFA_COMMON_STORAGE_USAGE check:

- ▶ This check requires that the following DIAGxx parmlib member options are set:
VSM TRACK CSA(ON) SQA(ON)
- ▶ When an exception occurs, refer to the best practices defined for this check in *z/OS Problem Management*, G325-2564.

LOGREC arrival rate

The PFA_LOGREC_ARRIVAL_RATE check is designed to detect a potentially damaged address space or system based on the LOGREC arrival rate being too high. The rate produced by this check is the number of LOGREC record arrivals within the collection interval grouped by key. The key groupings are LOGREC records for key 0 by itself; keys 1 through 7 as a group; and keys 8 through 15 as a group.

This check models the expected number of LOGREC records by key for four time ranges: 1 hour, 24 hours, 7 days, and 30 days. This check does not use the CPU consumption to normalize the arrival rate. The reason for this is that it assumes that a completely healthy system will not create any LOGREC records, no matter how busy it is. So, any number of LOGREC records greater than zero is considered to be “bad”.

The check models the number of LOGREC records that are normally created by this system over each hour, day, week, and 30 days. This information is then used to detect changes in

the record creation rate because this check assumes that an abnormal number of LOGREC records being created is highly likely to indicate a damaged address space.

Any one of the key groupings within any one of the time ranges can produce an exception. When a time range does not have enough data available for the entire time period, it is not included on the PFA report.

PFA is made aware every time a new LOGREC record is created, and it keeps a running count of the number of records created over the last x minutes. When a collection occurs, the current count is saved in the PFA files.

This check is somewhat unique when compared to the other “expected rate” checks in that it does *not* run after every collection. Instead, it runs whenever it is scheduled by the IBM Health Checker for z/OS, based on the INTERVAL value for the check. When the check is scheduled, it uses the *current* count of LOGREC records created over the last x minutes, where x is the collection interval value, not the value as it existed at the end of the last collection interval. This ensures that the check is always working with the latest information. For example:

- ▶ The time of the last collection was 8:00 a.m. (this data is collected for the time period 7:00 a.m. to 8:00 a.m. because COLLECTINT is 60).
- ▶ The time of the next comparison is 8:45 a.m. (every 15 minutes, because the INTERVAL is set to 15).
- ▶ The LOGREC record arrival rate used in the comparison is the arrival rate from 7:45 a.m. to 8:45 a.m.

This check does not look for individual LOGREC records, patterns of LOGREC records, or bursts of failures unless that burst is within a key grouping.

This check supports the ability to exclude address spaces that issue LOGREC records erratically by allowing you to define them in the PFA_LOGREC_ARRIVAL_RATE/config/EXCLUDED_JOBS file. This feature is especially useful when address spaces are not production-ready and create many LOGREC records on an irregular basis, or when address spaces that issue LOGREC records in a regular fashion are stopped and started frequently.

PFA_LOGREC_ARRIVAL_RATE check:

- ▶ The record must be a software LOGREC record with a usable SDWA to be counted as an arrival.
- ▶ z/OS LOGREC provides two options for the LOGREC recording medium: a System Logger log stream, or the LOGREC data set.

When a LOGREC record is produced, PFA is notified through an ENF listener. If you are writing a LOGREC record to the LOGREC data set and that data set fills up, PFA stops getting notified when a LOGREC is produced.

Therefore, for the best reliability, use the log stream method.

- ▶ If you change the COLLECTINT, perform the following steps to ensure that the values being modelled in the check remain consistent:
 - First, stop PFA.
 - Next, delete the files in the PFA_LOGREC_ARRIVAL_RATE/data directory.
 - Finally, restart PFA.
- ▶ If an exception occurs, refer to the best practices in *z/OS Problem Management*, G325-2564.

Frames and slots usage

As noted in the label box **Apply the PTF for APAR)A40065** on page 52, the PFA_FRAMES_AND_SLOTS_USAGE check has been permanently removed from PFA.

Message arrival rate

The PFA_MESSAGE_ARRIVAL_RATE check detects a potentially damaged address space or system based on the message arrival rate being higher or lower than expected. Messages included in the count of arrivals are single line and multi-line WTO and WTOR messages. Multi-line WTO messages are counted as one message. Branch Entry WTO messages are not counted. Messages are counted prior to being excluded or modified by other functions such as message flood automation.

The message arrival rate is a simple ratio calculated by dividing the number of arrivals in the collection interval by the CPU time used in that same collection interval.

This check does not track the message by individual message identifier, and therefore is not designed to detect abnormal message patterns or single critical messages.

The message arrival rate check performs its comparisons after every collection rather than being scheduled by the IBM Health Checker for z/OS INTERVAL parameter. By performing the check automatically upon successful completion of a collection, the check is able to compare the most recent arrivals with the predictions modeled at the last modeling interval. This design increases the validity of the check itself and the responsiveness of the check to the current activity of the system. Even though it can be run manually using IBM Health Checker for z/OS interfaces, there is no benefit because doing so only repeats the last comparison.

The message arrival rate check creates models and performs comparisons for time ranges based on 1 hour of data, 24 hours of data, and 7 days of data. If the amount of data is not available, the total system rate line for that time range is not printed on the report and UNAVAILABLE is printed for the individual address spaces for that time range.

PFA_MESSAGE_ARRIVAL_RATE check: So that address spaces that produce erratic numbers of messages do not skew the predictions, the following exclusions are done:

- ▶ The CONSOLE address space is excluded from this check's processing. This behavior cannot be changed.
- ▶ Any address space that starts with "JES" is excluded from this check's processing by default through the PFA_MESSAGE_ARRIVAL_RATE/config/EXCLUDED_JOBS file. This behavior can be changed, but is not suggested.

For guidance about what to do when an exception occurs for a rate that is too high, refer to 2.9.5, "PFA message arrival rate check exception issued for a high rate" on page 91.

If an unexpectedly low message arrival rate is detected, examine the report in SDSF for details about why the exception was issued. Use the Runtime Diagnostics output in the report to assist you in diagnosing and fixing the problem. Also, refer to 2.9.6, "PFA exception issued for a low rate" on page 92.

SMF arrival rate

The PFA_SMF_ARRIVAL_RATE check detects a potentially damaged address space or system based on the SMF record arrival rate being higher or lower than expected. The check's metric is the rate derived from the count of the SMF arrivals per the amount of CPU used in the interval.

If SMF is not active, this check is not processed and a WTO message is issued.

The SMF arrival rate check is not designed to detect performance problems caused by insufficient resources, a faulty WLM policy, or spikes in work. However, it might help to determine whether a performance problem detected by a performance monitor is caused by a damaged system.

This check does not detect abnormal SMF patterns or individual types of SMF records.

If SMF is stopped and restarted across a collection interval, PFA detects this change and automatically deletes the previously collected data and re-enters the warm-up phase to detect which jobs to track.

PFA_SMF_ARRIVAL_RATE check: If a change to your SMF definitions would result in a significant change in the number of SMF records being produced, it is advisable to do the following:

1. Stop PFA.
2. Delete the files in the PFA_SMF_ARRIVAL_RATE/data directory.
3. Restart PFA.

This ensures that PFA does not use the previously collected data in the models so that the models accurately reflect the new configuration.

If an unexpectedly high SMF record arrival rate is detected, review the SMF records sent by the address spaces identified on the report and examine the system log to determine what caused the increase in SMF activity. Also, refer to 2.9.7, "PFA exception issued for a high SMF arrival or high ENQ request rate" on page 92.

If an unexpectedly low SMF record arrival rate is detected, examine the report in SDSF for details about why the exception was issued. Use the Runtime Diagnostics output in the report

to assist you in diagnosing and fixing the problem. Also, refer to 2.9.6, “PFA exception issued for a low rate” on page 92.

The SMF arrival rate performs its comparisons after every collection rather than being scheduled through the IBM Health Checker for z/OS INTERVAL parameter. By performing the check automatically upon successful completion of a collection, the check is able to compare the most recent arrivals with the predictions created at the last modeling interval. This enhances both the validity of the comparisons and the responsiveness of the check to the current activity of the system. Even though it can be run manually using IBM Health Checker for z/OS interfaces, there is no benefit because doing so only repeats the last comparison.

The SMF arrival rate check creates models and performs comparisons for time ranges based on 1 hour of data, 24 hours of data, and 7 days of data. If the amount of data is not available, the total system rate line for that time range is not printed on the report and UNAVAILABLE is printed for the individual address spaces for that time range.

This check supports excluding address spaces that issue SMF records erratically by allowing you to identify them in the PFA_SMF_ARRIVAL_RATE/config/EXCLUDED_JOBS file.

JES2 spool usage

The PFA_JES_SPOOL_USAGE check is designed to detect abnormalities in persistent address spaces based on an abnormal increase in the number of spool track groups being used from one collection to the next. An abnormal increase in track groups usage can indicate a damaged job.

This check does not look for exhaustion of track groups. An abnormal increase might occur even if the address space is using a seemingly small number of track groups.

If the increase in the number of track groups is small and an exception is issued that you want to avoid in the future, you can increase the EXCEPTIONMIN parameter to be the minimum increase that must exist for PFA to perform the comparison.

The total number of track groups used per address space is listed on the reports. However, this value is irrelevant to PFA processing and is only provided for your reference.

Figure 2-22 shows a sample JES Spool Usage report for an exception caused by JOB1 and JOB55. The reason for the exception is that the change in the number of track groups used

from one collection to the next is much higher than the expected change. The current number of track groups used is not used by the comparisons and is only provided for your reference.

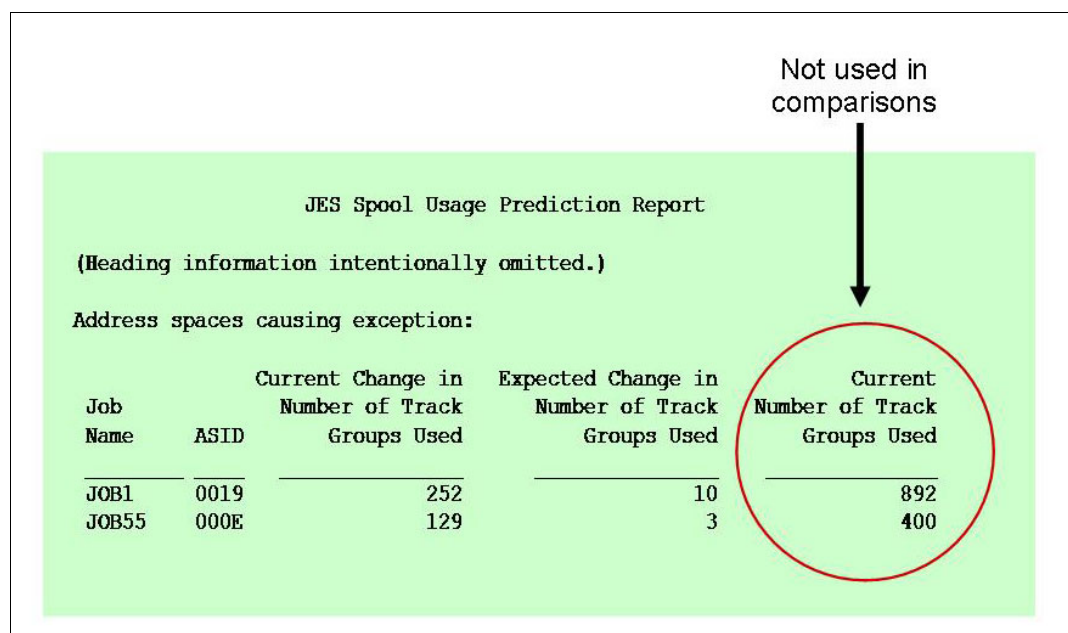


Figure 2-22 PFA JES Spool Usage Prediction Report

Not all persistent address spaces are modeled, but the data is collected for all persistent address spaces. At modeling time, PFA determines which address spaces had the highest *increase* in their usage and makes predictions for those because they are the ones that are showing the most difference in their behavior.

Modeling a subset of the address spaces reduces the CPU cost of the modeling and eliminates data that is inconsequential to causing a soft failure. During the comparisons, if PFA detects that address spaces that were not modeled have increased their usage significantly, it remodels immediately to include those address spaces in the model.

Not all persistent address spaces are listed on the report. When this check determines that no exception exists, the address spaces with the highest increase in their usage are printed on the report rather than those that are using the highest number of track groups. When an exception occurs, the persistent address spaces that caused the exception, that is, those deemed to be using an abnormal number, are printed on the report.

This check supports excluding address spaces that use erratic numbers of track groups by allowing you to identify them in the PFA_JES_SPOOL_USAGE/config/EXCLUDED_JOBS file.

PFA_JES_SPOOL_USAGE check: This check is only active if JES2 is the primary JES. A WTO message is issued when PFA starts if JES2 is not the primary JES, and the check will not be initialized.

Enqueue request rate

The PFA_ENQUEUE_REQUEST_RATE detects a potentially damaged address space or system based on the enqueue request rate being higher or lower than expected. The check's metric is the count of ENQ requests per amount of CPU used in the interval.

The report for this check is nearly identical to that of the message arrival rate and SMF arrival rate checks. However, this check only tracks the top individual persistent address spaces and

the total system as described in 2.5.7, “How PFA groups address spaces for monitoring” on page 60.

This check supports the ability to exclude address spaces from its processing using the PFA_ENQUEUE_REQUEST_RATE/config/EXCLUDED_JOBS file. The PFA install script creates an EXCLUDED_JOBS file that excludes *MASTER* and NETVIEW* when PFA is installed. Other address spaces can be excluded by adding them to this file as needed.

The enqueue request rate check performs its comparisons after every collection rather than being scheduled through the IBM Health Checker for z/OS INTERVAL parameter. By performing the check automatically upon successful completion of a collection, the check is able to compare the most recent system behavior with the predictions created at the last modeling interval. This enhances the validity of the comparisons and the responsiveness of the check to the current activity of the system. Even though it can be run manually using IBM Health Checker for z/OS interfaces, there is no benefit because doing so only repeats the last comparison.

The enqueue request rate check creates models and performs comparisons for time ranges based on 1 hour of data, 24 hours of data, and 7 days of data. If the amount of data is not available, the total system rate line for that time range is not printed on the report and UNAVAILABLE is printed for the individual address spaces for that time range.

This check requires two time ranges to be available before comparisons are made.

PFA_ENQUEUE_REQUEST_RATE check: If you change the maximum number of concurrent ENQ, ISGENQ, RESERVE, GQSCAN and ISGQUERY requests, or change system-wide defaults using the SETGRS command or through the GRSCNFxx Parmlib member, or the system is radically different after an IPL (such as a change from a test system to a production system), perform the following steps:

1. Stop PFA.
2. Delete the files in the PFA_ENQUEUE_REQUEST_RATE/data directory.
3. Restart PFA.

This ensures that PFA is collecting relevant information and that your models accurately reflect the behavior of your system.

If an unexpectedly high enqueue request rate is detected, examine the report in SDSF for details about why the exception was issued. Also, refer to 2.9.7, “PFA exception issued for a high SMF arrival or high ENQ request rate” on page 92.

If an unexpectedly low enqueue request rate is detected, examine the report in SDSF for details about why the exception was issued. Use the Runtime Diagnostics output in the report to assist you in diagnosing and fixing the problem. Also, refer to 2.9.6, “PFA exception issued for a low rate” on page 92.

2.5.7 How PFA groups address spaces for monitoring

Each individual metric that PFA uses has its own characteristics of what is important to track and what data to use. For example:

- PFA excludes the data collected within the first hour after IPL and the last hour of data collected before the last shutdown because that data can skew the results of the prediction.
- PFA also uses the data collected prior to the IPL only if it can be applied to the trends after the IPL. For example, the data collected for common storage exhaustion cannot be

applied after an IPL because the users of storage are not necessarily the same after the IPL. Similarly, the data obtained for the JES spool usage check cannot be used after the IPL.

You cannot manually exclude specific time ranges of data from the model. However, you can exclude data from specific address spaces from being modeled for most checks. More information about excluding specific address spaces can be found in the list item “Eliminate address spaces causing exceptions” on page 66.

Table 2-2 on page 61 describes which address spaces are tracked for each check. For some metrics, the results are the most accurate when using several different groupings of address spaces. Multiple categories allows the check to detect both damage to individual address spaces and to the entire system.

When a check supports several categories, the comparisons are performed in the following order for each category supported. If an exception is found in a category, the exception is issued for that category and the subsequent categories are not compared.

1. Tracked, persistent address spaces
Those with the highest individual rates in a warm-up period.
2. Other persistent address spaces
All the persistent address spaces not being tracked individually are grouped.
3. Other non-persistent address spaces
All the non-persistent address spaces as a group.
4. Total system
All address spaces as a group.

Table 2-2 The address spaces that are tracked per check

The address spaces tracked	Which checks track this data	Description
Total system as a group	Common Storage Usage LOGREC Arrival Rate Message Arrival Rate SMF Arrival Rate ENQ Request Rate	For some metrics, the data is required to be tracked for the entire system to determine whether exhaustion is occurring or if the cumulative failures on the system indicate a damaged system. For example, for exhaustion of common storage, it tracks the common storage usage for the entire system. Similarly, detecting a damaged system based on LOGREC arrival rates by key groupings requires accumulating the arrivals for the entire system.

The address spaces tracked	Which checks track this data	Description
Every persistent address space individually	JES Spool Usage	<p>A <i>persistent address space</i> in PFA is defined to be any address space that was started within the first hour after IPL. This definition was created so that PFA could detect critical system address spaces without using a hardcoded list.</p> <p>Address spaces with duplicate names are not tracked. If a persistent address space restarts while PFA is running, it continues to be considered persistent even if it is assigned a different ASID.</p> <p>This category is used to detect a damaged persistent address space.</p>
Tracked, persistent address spaces	Message Arrival Rate SMF Arrival Rate ENQ Request Rate	<p>Tracking address spaces that have the highest rates individually allows PFA to detect damaged or hung address spaces among the address spaces that tend to have the highest rates for this metric.</p> <p>PFA checks determine which persistent address spaces to track by entering a six-hour warm-up phase for the check to determine which address spaces have the highest rates in that period. The warm-up phase waits for one hour after IPL and then collects data for six hours. At the end of the warm-up phase, the rates are calculated for each persistent address space and those with the highest rates are chosen to be tracked.</p> <p>If PFA is restarted or an IPL occurs, PFA still tracks the same address spaces without entering the warm-up period if those address spaces still exist and the maximum PFA chooses to track were being tracked prior to the IPL or PFA restart. When PFA can track the same persistent address spaces, it can use the data collected prior to the IPL in its modeling.</p> <p>Address spaces with duplicate names are not tracked individually, but if an address space that is tracked ends and restarts, it continues to be tracked even if it was assigned a different ASID. In this case, the data collected from the previous address space is used when making predictions for this address space.</p>
All other persistent address spaces as a group	Message Arrival Rate SMF Arrival Rate	Some metrics also categorize all persistent address spaces that are not being tracked individually as a group and compare the actual usage of those persistent address spaces in this category to the group average.
Non-persistent address spaces as a group	Message Arrival Rate SMF Arrival Rate	Some metrics track the address spaces that started more than an hour after IPL as the non-persistent address spaces as a group.

2.5.8 PFA and Runtime Diagnostics integration

Consider the case where the system is issuing an unusually low number of messages. This could mean that there is simply little happening at that time (it could be a public holiday, for example). Or it might mean that there is a significant problem on the system and all the work has come to a halt. You would not want PFA issuing an exception in the first case. But you would want it to issue an exception in the latter case. To help PFA determine whether the low message rate indicates a problem or not, it exploits the diagnostic capabilities of Runtime Diagnostics.

Starting with z/OS 1.13, PFA and Runtime Diagnostics were integrated so that PFA can detect a potentially hung persistent address space or system by determining that a metric is *too low* compared to its expected value. Prior to this enhancement, PFA was only able to detect if the metric was abnormally high.

As shown in Figure 2-23, PFA performs its comparisons, then the following activities occur:

- ▶ If PFA believes that a metric is abnormally low, it verifies its findings with Runtime Diagnostics.
- ▶ If Runtime Diagnostics determines that the situation really is a problem, a PFA exception is issued and the Runtime Diagnostics output is included in the PFA report.
 - If PFA detects the problem to be in specific address spaces, it asks Runtime Diagnostics to perform checking only for those address spaces.
 - If PFA detects the problem to be for the entire system, it directs Runtime Diagnostics to perform its comprehensive checking.
 - In either case, when PFA invokes Runtime Diagnostics, critical message analysis is not performed.

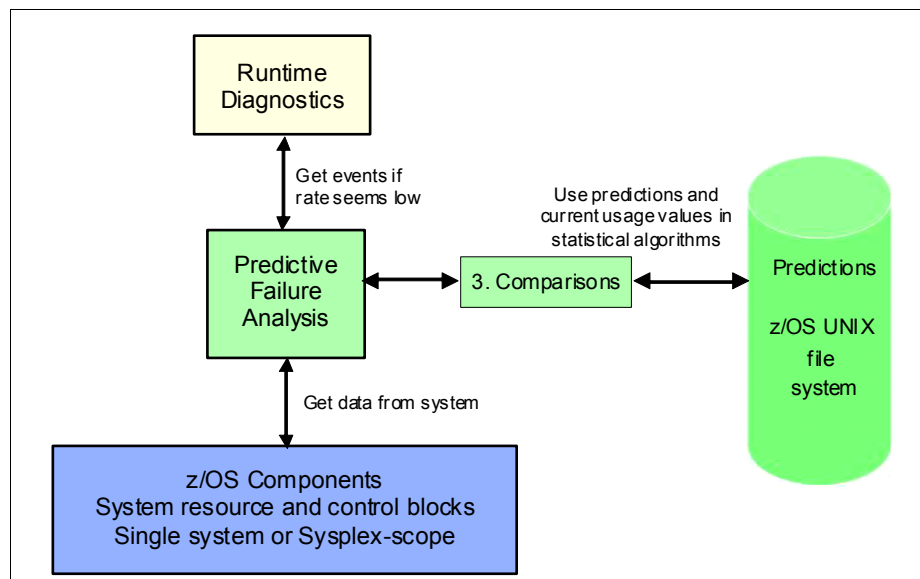


Figure 2-23 PFA and Runtime Diagnostics integration

This new feature is supported by three checks: message arrival rate, SMF arrival rate, and the enqueue request rate check. The message arrival rate and SMF arrival rate checks support detection of an abnormally low condition for the tracked, persistent address spaces; the other persistent address spaces as a group; and the total system rates. The enqueue request rate supports this checking for the tracked, persistent address spaces and the total system rate.

The Runtime Diagnostics address space must be active for PFA to interact with it. If Runtime Diagnostics is not active when PFA starts, PFA issues a WTO message and does not perform the comparisons for detecting a “too low” condition.

If Runtime Diagnostics is started after PFA starts, PFA detects that Runtime Diagnostics is now active and performs the checking for “too low” when the next comparison is performed.

If PFA has detected that the rate was too high in any of the categories it compares, it does not perform checking for a rate being too low.

The supported categories are checked in the following order:

1. Tracked, persistent address spaces
2. Other persistent address spaces
3. Total system rate

If a category detects a too low condition and Runtime Diagnostics corroborates the comparison, exceptions for the subsequent categories are not issued.

Figure 2-24 shows an example of the Message Arrival Rate prediction report for a “too low” exception.

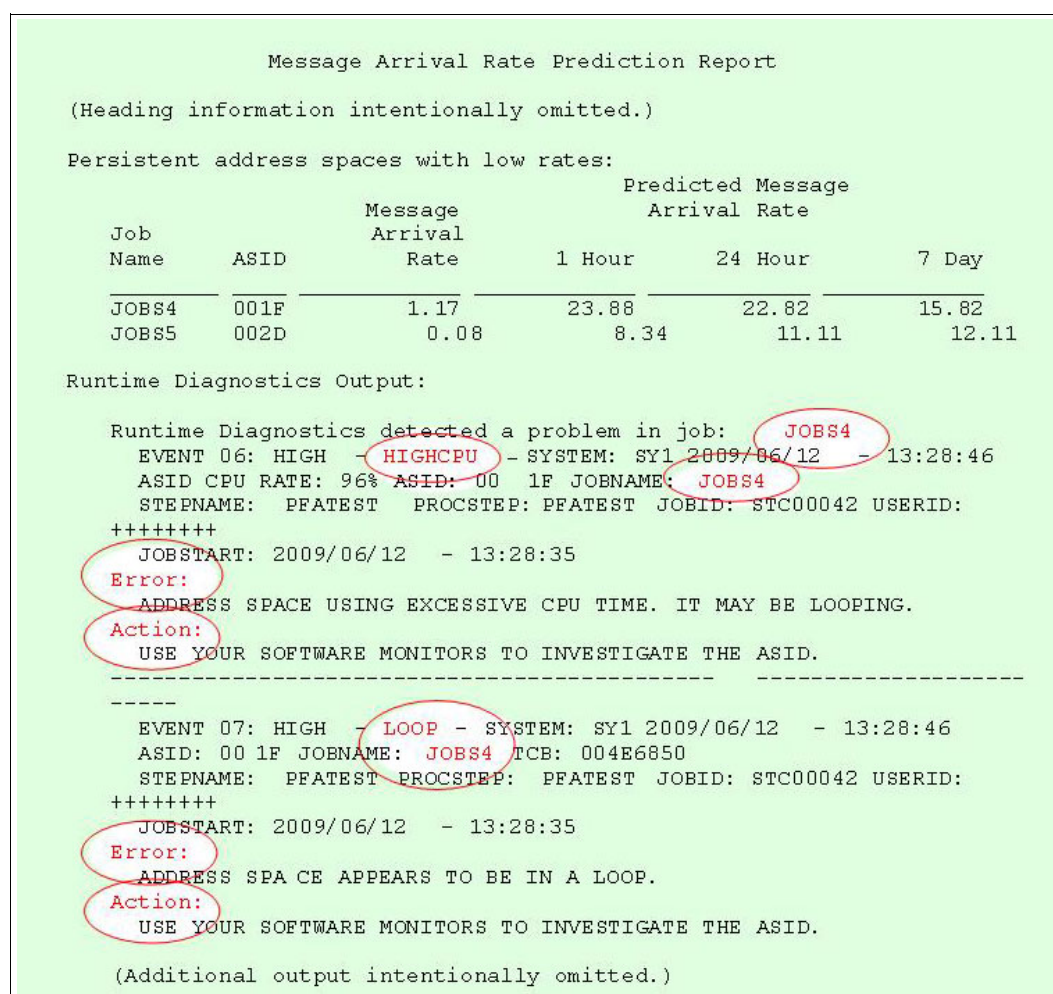


Figure 2-24 PFA and Runtime Diagnostics exception report

When an exception for an abnormally low condition is found, a health check exception is issued describing the problem. The PFA report includes the current rates and predicted rates for the categories that were failing. It also includes the Runtime Diagnostics output received when PFA called Runtime Diagnostics to verify the problem.

Note that in this example, PFA detected that jobs JOBS4 and JOBS5 had a Message Arrival Rate that was likely too low when compared to their expected rates. Runtime Diagnostics verified that there could be a problem by detecting both a HIGHCPU and a LOOP event for JOBS4. Therefore, the abnormally low message arrival rate coupled with the results of Runtime Diagnostics show that JOBS4 is likely to be looping. The Runtime Diagnostics output for JOBS5 was similar, but was purposely omitted from this example for simplicity.

As with the other PFA prediction reports, the PFA prediction reports for abnormally low conditions are available in SDSF.

PFA uses Runtime Diagnostics only for “too low” problems: PFA only uses Runtime Diagnostics for “too-low” problems because PFA is unable to determine whether or not the lack of activity is really a problem. For “too-high” problems, PFA is able to determine the problem on its own and is usually able to present helpful diagnostic information based on the information that is available to it.

For this reason, and also to eliminate the CPU cost of running Runtime Diagnostics when it is not really needed, PFA does not invoke Runtime Diagnostics for “too-high” problems.

2.5.9 Achieving maximum benefit from PFA

The following tips can help you obtain maximum benefit from using PFA:

- ▶ Start PFA and its companion address spaces (Runtime Diagnostics and IBM Health Checker for z/OS) at IPL.
- ▶ Use a zAAP to offload PFA modeling processing.

The complex algorithms used by modeling would be compute-intensive in any language. Because this processing in PFA is done in Java, you can offload this work to a zAAP (or zAAP on zIIP).

- ▶ Follow the installation steps in *z/OS Problem Management*, G325-2564. The following steps are the most common omissions:
 - Run the install script for each release. The most common indication that the install script was not run is the following message:

```
AIRO10I PFA CHECK check_name WAS UNABLE TO OPEN LOG FILE, ERRNO=00000081  
ERRNOJR=0594003D
```
 - Update the `ini` file to specify the paths to PFA's Java code and the Java code for your system if it is not in the paths specified by default. If PFA cannot find the Java code, the problem is typically reported as:

```
AIRO22I REQUEST TO INVOKE MODELING FAILED FOR CHECK NAME=check_name. UNIX  
SIGNAL RECEIVED=00000000 EXIT VALUE=00000006 (or EXIT VALUE=00000002).
```
 - Allocate the appropriate amount of DASD for your release as documented in *z/OS Problem Management*, G325-2564.
 - Update your PFA JCL to specify the path in which you installed the executable code if it is not the default path `/usr/lpp/bcp`. The most common symptom of this omission is

that even after modeling should have occurred, the successful model count for the checks is 0 even though there are no AIR022I messages.

Also, the *systemName*MODEL.LOG file contains the following line:

```
AIRHLJVM failed BPX1SPN errno=00000081 errnoj=053B006C
```

The default PFA procedure is shown in Example 2-2. To resolve the previous error, update the path statement to point at the correct directory:

Example 2-2 PFA procedure

```
//PFA      EXEC PGM=AIRAMBGN,REGION=OK,TIME=NOLIMIT,  
//        PARM='path=(/usr/lpp/bcp)'
```

- ▶ Develop automation routines for the PFA Health Check exceptions.

There are several ways to accomplish this task. For more information, refer to *IBM Health Checker for z/OS*, SA22-7994.

- ▶ Stay current by getting the most recent PFA PTFs.

- PFA strives for continuous improvements in its algorithms so that soft failures can be detected more accurately. These improvements are released in the service stream when they are available. To obtain a list of all available PFA service, search on the PFA component ID: 5752SCPFA.
- If you believe you have received an invalid exception and tuning is not helping, contact IBM service. Customer feedback on potential problems helps IBM to improve the PFA algorithms.

- ▶ Use the PFA modify command to display parameters.

Use the PFA modify command rather than the IBM Health Checker for z/OS modify command to display all the parameters and status details for a PFA check, as described in the label box **Display all parameters and obtain status of PFA checks** on page 46.

- ▶ Eliminate address spaces causing exceptions.

Exclude from PFA processing those address spaces that are normally unstable and unpredictable.

For example, when you suspect certain jobs or address spaces are inconsistent and have the potential of being restarted often, you can increase the accuracy of PFA reporting by excluding the job or address spaces from analysis.

Excluding jobs is also useful in a test environment where certain test programs are expected to have erratic behavior and can cause unwanted exceptions, such as issuing many LOGRECs or many WTOs and causing exceptions for those checks.

To exclude jobs from processing, follow these steps:

- First, add the jobs to the EXCLUDED_JOBS file in the check's /config directory.
- Then use the update command **f pfa,update,check(pfa_check_name)** to have PFA read the contents of the file and start excluding the jobs you added.

- ▶ Change the type of WTO of a check.

The type of WTO issued when a PFA check issues an exception is controlled by the SEVERITY and WTOTYPE parameters of the Health Check. The default value for all PFA checks is SEVERITY(MED), which issues an eventual action WTO for a PFA check exception.

If this value is not appropriate for a PFA check in your installation, you can modify the WTOTYPE setting using the IBM Health Checker for z/OS modify command. For

example, to change the JES spool usage check so that it issues informational messages to the console, use the following command:

```
f hzsproc,update,check(ibmpfa,pfa_j*),wtotype=info
```

- Configure checks to reduce or eliminate unwarranted exceptions.

Exceptions tend to be ignored if they occur too frequently when a real problem does not exist. Therefore, reducing the number of unwarranted exceptions leads not only to better accuracy, but a greater sense of urgency when a PFA exception occurs.

Even with the complex comparison algorithms in place, predictive technology requires part art and part science and unwarranted exceptions can be reported, especially in an environment that is unpredictable. To allow you to influence PFA's results in these environments, mechanisms are provided to allow you to tune the behavior of the individual checks.

All PFA checks were implemented with at least one configuration parameter to help you tune the sensitivity of the check. Default values for all parameters were carefully chosen to minimize the amount of customization required for most systems. However, your individual environment might require different values than the defaults.

Do not be intimidated by these parameters. They are available for your use so that PFA can detect problems in your environment accurately and so that you are satisfied with the results.

- If you are experiencing unwarranted exceptions and the problem is not for a specific address space that is unstable, modify the configuration value for the check to cause the comparisons to be less sensitive. For example, if the standard deviation value (STDDEV) is currently set to 5, incrementally increase the value until the unwarranted exceptions are reduced or eliminated.
- If the problem is for specific address spaces and you have investigated the issue and are sure that they are not going to cause soft failures, consider excluding them from the processing as previously described so that you can determine whether the check is issuing appropriate results for the rest of your address spaces. Alternatively, you can increase the configuration value for the check, but you then might miss real exceptions for other address spaces.

Table 2-3 describes how the values of the configuration parameters affect the sensitivity of the PFA comparison algorithms. For more information about the definitions of each parameter and the parameters supported for each check, see *z/OS Problem Management*, G325-2564.

To modify a value, change it using the IBM Health Checker for z/OS modify command as described in the label box **Display all parameters and obtain status of PFA checks** on page 46.

Table 2-3 PFA configuration parameters to adjust sensitivity of comparisons

Parameter	Supported checks	Description
THRESHOLD	Common storage usage	<p>This parameter is often confused and thought to be a hardcoded threshold (such as 80 percent) that the location must be using in order for the exception to be issued.</p> <p>Instead, the THRESHOLD is a percentage of the capacity that the <i>prediction</i> must be before the comparisons are performed after domain knowledge is applied. That is, it sets the sensitivity level of the comparison to know when <i>exhaustion of the available capacity</i> is being predicted.</p> <p>For example, if PFA's domain knowledge says that the location should allow 100 percent of the available capacity and THRESHOLD(2) is used, the prediction must be at 102 percent of the available capacity before comparisons for exhaustion are performed.</p> <p>This design allows PFA to detect exhaustion even though current usage might not be nearing exhaustion yet.</p> <p>To make this check less sensitive so it issues exceptions less frequently, set THRESHOLD to a higher value so that a larger prediction is required before exceptions can be issued.</p>
STDDEV	LOGREC Arrival Rate JES Spool Usage Message Arrival Rate SMF Arrival Rate Enqueue Request Rate	<p>This value is used as a multiplier to calculate the comparison value for comparisons detecting if the current usage or rate is too high compared to the expected value.</p> <p>A higher STDDEV value requires the difference between the current value and the prediction to be greater in order for an exception to be issued. Therefore, exceptions are issued less frequently with a higher STDDEV.</p> <p>To reduce exceptions indicating the metric is too high for checks that have the STDDEV parameter, incrementally increase the value until you are satisfied with the results.</p> <p>If an exception is issued, a <i>systemName</i>RUN.LOG file is created in the <i>EXC_timestamp</i> directory for this exception. For some checks, the STDDEV value needed to avoid the exception might be printed to help you tune the check if the exception was unwarranted.</p>

Parameter	Supported checks	Description
STDDEVLOW	Message Arrival Rate SMF Arrival Rate Enqueue Request Rate	<p>This parameter is similar to STDDEV except that it is used in the algorithms detecting “too low” conditions.</p> <p>This value is used to set the variance between the current value and the comparison value when detecting rates that are too low.</p> <p>A larger value requires the current rate to be significantly lower than the expected rate in order for an exception to be issued.</p> <p>To reduce exceptions indicating the metric is too low for checks that have the STDDEVLOW parameter, incrementally increase the value until you are satisfied with the results.</p> <p>If an exception is issued, a <i>systemName</i>RUN.LOG file is created in the <i>EXC_timestamp</i> directory for this exception. For some checks, the STDDEVLOW value needed to avoid the exception might be printed to help you tune the check if the exception was unwarranted.</p>
EXCEPTIONMIN	LOGREC Arrival Rate JES Spool Usage Message Arrival Rate SMF Arrival Rate Enqueue Request Rate	<p>This parameter is used when detecting abnormally high values and rates.</p> <p>A higher EXCEPTIONMIN value requires the current value (and sometimes the prediction) to be higher than this value before an exception can be issued.</p> <p>This parameter is used to eliminate exceptions for “noise” on the system. That is, if your system tends to have a low rate for a given metric, you can set this parameter to the minimum value that should be compared so that the normal, low rates are not even compared.</p> <p>To reduce indicating the metric is too high when your system is stable and normally has a relatively low rate or value, set EXCEPTIONMIN to the minimum value you want to be compared.</p> <p>For example, if your system typically has a message arrival rate of 20 and you do not want comparisons for high rates to be performed for rates that are less than 20, set EXCEPTIONMIN(20).</p>

Parameter	Supported checks	Description
LIMITLOW	Message Arrival Rate SMF Arrival Rate Enqueue Request Rate	<p>This parameter is available on all checks that detect hung address spaces or systems based on a determination that the rates were too low.</p> <p>Its purpose is to avoid “too low” conditions for rates that are higher than this value that indicate normal system activity is occurring.</p> <p>That is, comparisons for “too low” conditions are only performed for rates that are less than this value.</p> <p>To reduce exceptions indicating the metric is too low for rates that are relatively high, set LIMITLOW to the maximum value you want compared for “too low” conditions.</p> <p>For example, if your system typically has a message arrival rate of 3, set LIMITLOW to a value less than 3 so that low values are not even compared.</p>
CHECKLOW	Message Arrival Rate SMF Arrival Rate Enqueue Request Rate	<p>This parameter is available on all checks that detect hung address spaces or systems based on a determination that the rates were too low.</p> <p>Use this parameter to bypass comparisons for low rates and values if you do not want PFA to perform this function.</p>
TRACKEDMIN	Message Arrival Rate SMF Arrival Rate Enqueue Request Rate	<p>This parameter is used by the checks that find individual persistent address spaces to track.</p> <p>It defines the minimum rate that an address space must have at the end of the warm-up period in order for it to be tracked.</p> <p>To allow an address space with any rate to be considered for tracking, set this value to 0.</p> <p>If you notice that you have many tracked jobs with normally low rates and you receive unwarranted exceptions for this normal, low behavior, consider setting TRACKEDMIN to a higher value.</p> <p>To force PFA to track a different set of address spaces after changing this value, you must follow these steps:</p> <ol style="list-style-type: none"> 1. Stop PFA. 2. Delete the /data subdirectory for the check for which you are changing the TRACKEDMIN parameter. 3. Start PFA.

- Make configuration changes persistent.

Create a policy in the HZSPRMxx PARMLIB member to make configuration changes to PFA checks persistent.

- Quiesce checks that issue unwarranted exceptions.

Rather than stopping PFA or deleting a PFA check (which requires PFA to be restarted to re-add the check), if unwarranted exceptions are occurring and you need to postpone their investigation, you can quiesce PFA checks so that they do not issue exceptions, but continue their other processing, if desired. This method allows all other checks to continue processing and does not require PFA to be restarted to re-add a deleted check.

To quiesce a check:

- (Optional). if you want to stop the check from collecting, modeling, and issuing exceptions, set the COLLECTINACTIVE parameter to 0. For example, to modify the JES spool usage check to stop collecting and modeling when the check is not active, execute the following command:

```
f hzsproc,update,check(ibmpfa,pfa_j*),parm('collectinactive(0)')
```

Unless the collections and models are causing system issues, we suggest leaving COLLECTINACTIVE set to 1, which is the default on all PFA checks.

- Deactivate the check using IBM Health Checker for z/OS. For example, you can deactivate the JES spool usage check using the following command:

```
f hzsproc,deactivate,check(ibmpfa,pfa_j*)
```

To reactivate the check, use the following command:

```
f hzsproc,activate,check(ibmpfa,pfa_j*)
```

2.6 IBM System z Advanced Workload Analysis Reporter

The latest addition to the family of z/OS-related system management tools is IBM System z Advanced Workload Analysis Reporter (zAware). IBM zAware extends the set of tools designed to help you achieve higher availability. It does this by performing machine learning, pattern recognition, and statistical analysis of console messages to look for unexpected patterns and anomalies to provide fast, near realtime, detection of changed message behavior on your systems.

Chapter 1, “Introduction to IBM zAware” on page 1, introduces IBM zAware and described its architecture. This section summarizes the steps you need to follow to get IBM zAware running, and provides information to help you use it effectively.

2.6.1 Preparing IBM zAware for use

IBM zAware does not require authoring of rules or any coding to query the right fields, for example, to detect anomalies in your systems’ message activity. You will notice, for example, that this section of this chapter is much shorter than the PFA section. Compared to PFA, IBM zAware is a “black box” solution, meaning that little, if any, customization is required or possible.

After you connect a system to IBM zAware, there is little that you can do to influence its analysis of incoming messages. Most of our focus in this book is on planning for, using, and managing IBM zAware.

Figure 2-25 shows the sequence of steps required to prepare IBM zAware for use.

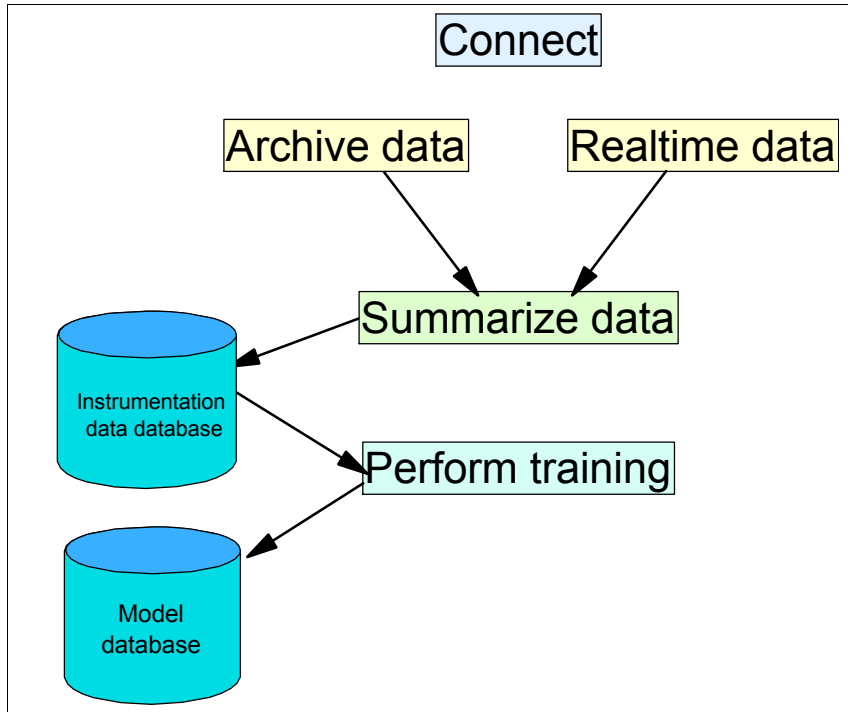


Figure 2-25 Preparing IBM zAware for use

The required steps are summarized here:

1. Connect

Each monitored client needs to connect to IBM zAware. The connection is required for two reasons, that is, to enable the transmission of archived log data, and so that messages can be sent to IBM zAware in real time.

2. Load the instrumentation data database

The instrumentation data database can be primed using archived message data that was sent to IBM zAware using the bulk load utility, or by allowing IBM zAware to collect data in real time until enough data exists for training.

When message data is received by IBM zAware, it is parsed, summarized, and stored in the instrumentation data database, regardless of whether it is real time message data or historical archived data.

3. Training

When the instrumentation data database contains sufficient message data, you can use it to create a model for that system, using a process known as “training.” The training can be run after historical data has been bulk-loaded, or you can wait for the normal realtime message loading to pass over sufficient data to build a representative model of that system’s message behavior.

The following section explains these steps in greater detail.

Connecting monitored clients

Before you can start using IBM zAware to analyze your realtime message traffic compared to the normal behavior for that system, you need to send message data so that IBM zAware can build its model database. To enable the transmission of that data, you need to establish a TCP/IP connection between each monitored client and the IBM zAware LPAR. The network

connection that you establish between a monitored client and the IBM zAware LPAR can be used to transmit both realtime and historical message data.

Note that every monitored client will require its own connection to the IBM zAware LPAR. Even if multiple monitored clients are in the same sysplex, each will still require its own connection and will be responsible for sending its own realtime data.

Chapter 3, “Planning for an IBM zAware implementation” on page 95, contains information to help you plan for the required network connections. Chapter 4, “IBM zAware installation” on page 113, demonstrates our experiences with setting up IBM zAware, based on the information provided in the *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

Assuming that all the prerequisite setup work has been completed, System Logger will start sending realtime message data to IBM zAware as soon as the connection is started (using the System Logger **SETLOGR FORCE, ZAICONNECT** command).

Loading the instrumentation data database

As each message is received by IBM zAware, it is reduced and summarized by message ID, counts of occurrences, and the first occurrence’s text. Clustering information is kept for pattern-matching. All of this information is then used to update the instrumentation data database.

It is possible to wait for the volume of messages received in realtime to build up to the point that you can successfully build a model of the system. However, we expect that most customers will extract historical message data from syslog or OPERLOG archive data sets and use that to prime the instrumentation data database.

The Bulk Data Load Utility allows you to transmit historical message data from archive data sets. You typically run the bulk load function once. However, if you run the utility more than once, IBM zAware simply overlays any data that it had already received.

Remember that no analysis will be performed, and no analysis data presented on the IBM zAware GUI, until a model for the system has been successfully created.

The Bulk Data Load Utility

The Bulk Data Load Utility consists of a batch job that runs a REXX exec to perform the transformation and load of the message data into a temporary log stream for transmission to the IBM zAware application.

The Bulk Data Load Utility processes sequential data sets that contain data from syslog (from JES2) or OPERLOG (from both JES2 and JES3) in HCL (two-digit year) or HCR (four-digit year) format. The input files can contain message data from one or multiple systems.

After the message data has been successfully transmitted to IBM zAware, *and* each monitored client has connected to IBM zAware, then you must use the IBM zAware GUI Assign function to associate each system with the sysplex that it is a member of, as shown in Figure 2-26².

² System Logger passes the sysplex name for each realtime message that is sent to IBM zAware. However, that information is not available for message data that is sourced from an archive data set. That is why you must perform this association using the Assign function.

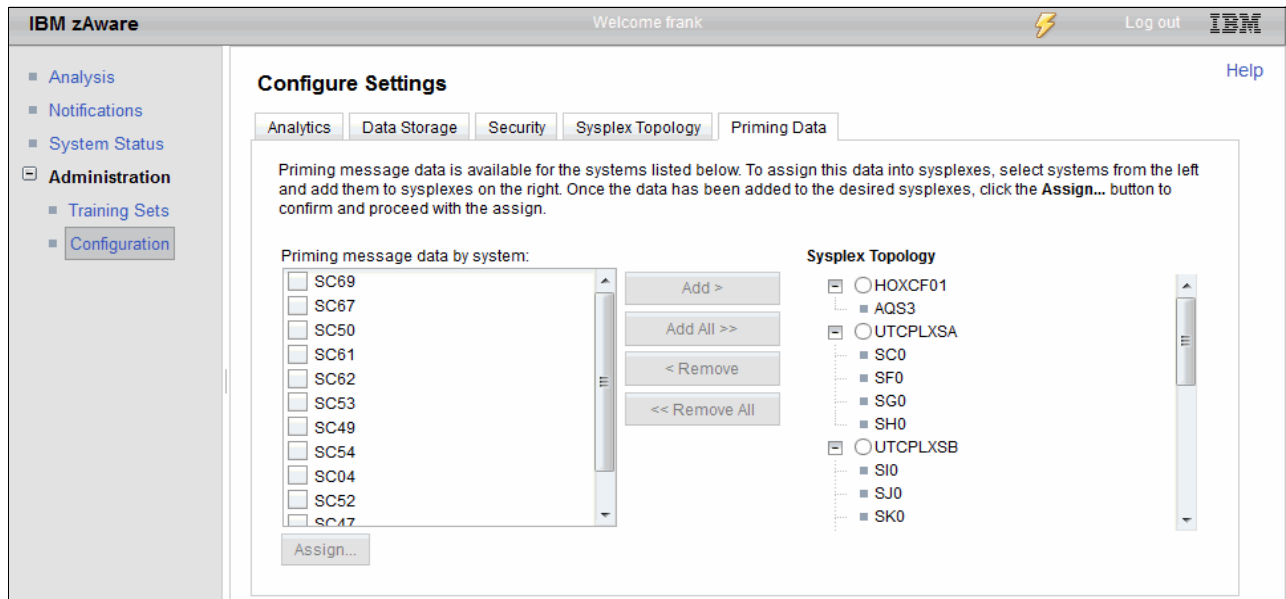


Figure 2-26 Assign bulk data window

The monitored client that runs the bulk data load job must have a connection to the IBM zAware application. Information about managing connections to IBM zAware is provided in 5.4, “Managing connections from monitored clients” on page 165.

Chapter 13 in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623, explains how to perform the bulk data load.

Training

The final step in preparing IBM zAware for use is to build the model of normal system behavior from the historical data. The Training Sets tab on the IBM zAware GUI presents a list of systems that have data in the instrumentation data database.

To create the initial model for each system, you would normally select the system that you want to train, and then select the Request Training option in the Actions drop-down, as shown in Figure 2-27 on page 75.

Tip: You might find that IBM zAware appears to be selective about which systems are able to be successfully trained. This is particularly the case for test or development systems. However, remember that IBM zAware is trying to emulate what you do when presented with a system problem, which is, identifying what is different about the system now, compared to its normal state. The best way that IBM zAware has of creating a reliable model of the normal state of a system is to require that monitored systems exhibit consistent behavior and a statistically significant number of messages.

Successfully creating a model for *every* system, even one that produced few messages or exhibited no consistency, would simply result in IBM zAware marking normal messages as being anomalous. If it behaved in this manner, you would soon learn to ignore its results, thus reducing its value as a reliable tool.

The topic of identifying suitable systems for IBM zAware to monitor is discussed further in 3.2, “Selecting which systems to monitor with IBM zAware” on page 96.

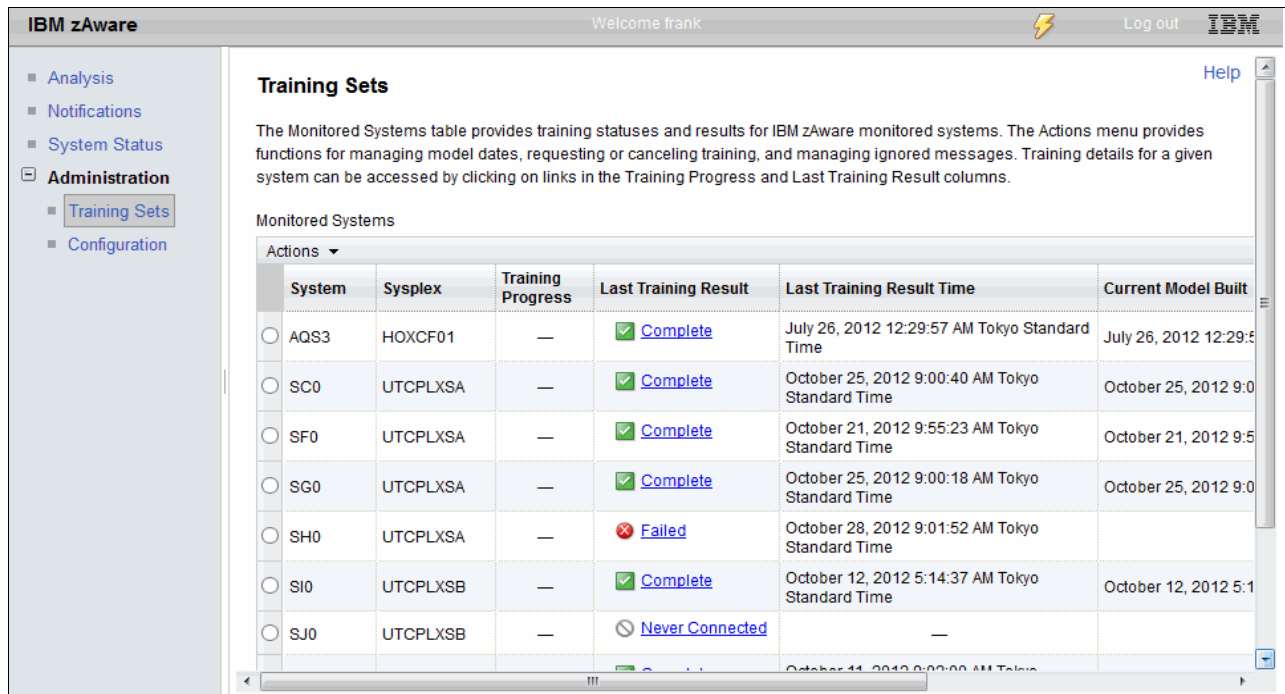


Figure 2-27 Training Sets window

IBM zAware training divides the historical data in the Instrumentation data database into 10-minute slices.

Within those time slices, it performs the following tasks:

- ▶ Categorizes message ids.
- ▶ Determines the frequency of messages being issued by determining the percentage of intervals that contain the message and the number of times within the interval the message occurred.
- ▶ Attempts to identify groups of messages that consistently appear together.

Ninety days of message data is the recommended minimum amount of data for training. This increases the likelihood that clusters of messages will occur multiple times and be identified as a valid pattern. Having data that represents mid-week and weekends, month-end, and any other special processing periods increases the value of your model.

Regardless of how many days of message data you use, it is important that there are enough unique message IDs and enough repetition to allow IBM zAware to discern clusters of messages and to create a realistic model.

The initial training will probably be initiated manually. Subsequent training, to update the model with current message data, can be performed manually or (more likely) on a schedule. Refer to 2.6.3, “Achieving maximum benefit from IBM zAware” on page 81 for a discussion about which method is more useful for you.

2.6.2 Using IBM zAware

After you perform the preparation steps, you will have a model of the “normal” message activity of a system, and be sending realtime message data to the IBM zAware LPAR. To help you obtain the best value from IBM zAware, the following sections explain some of the

processing that IBM zAware performs, and then discuss what you can expect to see on the IBM zAware GUI Analysis results panels and how to use that information.

When a model of the system exists and realtime message data is being sent to IBM zAware, every two minutes the analytics engine compares the current data to the model. The results of the comparisons are output to the XML files and to the IBM zAware GUI. When performing the analysis, IBM zAware applies the following criteria to each message ID so that it can assign an anomaly score to that message:

- ▶ How frequently is this message ID seen in the model database? Messages that appear infrequently are, by their nature, more anomalous than ones that are issued every few minutes.
- ▶ Is there any domain-specific information for this message ID? For example, IBM zAware might know that this message is quite common and is purely informational and therefore is generally considered to be “background noise.” Or, it might know that this message can be an indication of a significant system problem.
- ▶ If this message is part of a cluster, then are the other messages that normally appear in that cluster also present?
- ▶ Is the frequency with which the message is appearing significantly different than the normal frequency with which it is seen?

When investigating a potential problem on your system, these are the same types of questions that you as an experienced system programmer apply to the logs that you are reviewing. And IBM zAware can process huge volumes of data in a short time. But you can use intuition and knowledge of other environmental factors that IBM zAware might not be aware of. So the objective is to help you combine your experience with IBM zAware’s strengths to more quickly identify potential problems. IBM zAware’s GUI is designed to help you achieve that.

Contents of the Analysis view

The results of the analysis of the realtime message data is shown on the IBM zAware Analysis view window, as shown in Figure 2-28 on page 77. Notice that each monitored client has its own row, with each row containing a number of bars. The height and the color of the bars on the Analysis view present two types of information, as explained here:

- ▶ The *height* of the bar is based on the number of unique message IDs for that system in that interval.

For example, if messages ABC123I, DEF123I, and GHI456E are issued, that counts as three different messages. If message ABC123I was issued three times, that only counts as one message. So, a high bar indicates that many unique messages were issued in that interval.

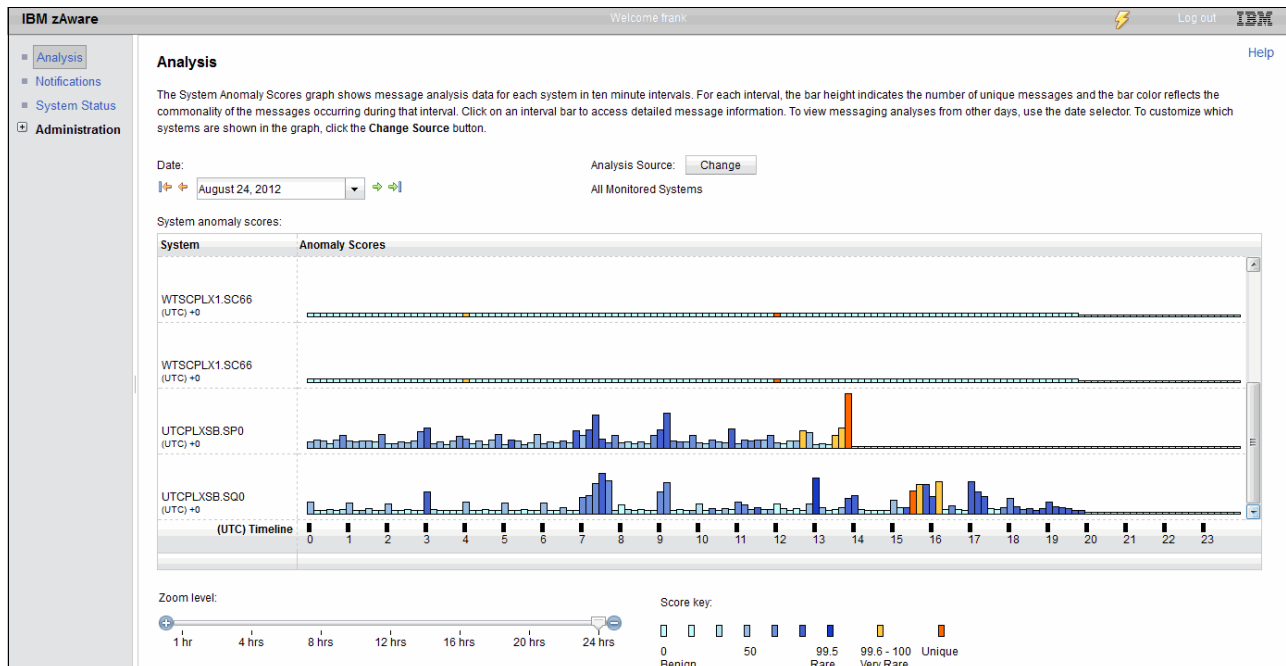


Figure 2-28 IBM zAware Analysis view

- The *color* of the bar represents how close the messages in the corresponding interval are to the model of that system.

Factors such as messages being issued out of context, messages being issued at a higher rate than expected, messages that were not seen in the model, or messages that IBM zAware knows are important messages will increase the interval anomaly score.

A low interval anomaly score will result in light blue bars. Higher interval anomaly scores will result in darker blue, yellow, or orange bars. In general, intervals with dark blue, yellow, or orange bars would be considered to be more “interesting”. A key (shown in Figure 2-29) describing the scores associated with the different colors is contained in the bottom-right part of the Analysis view window.

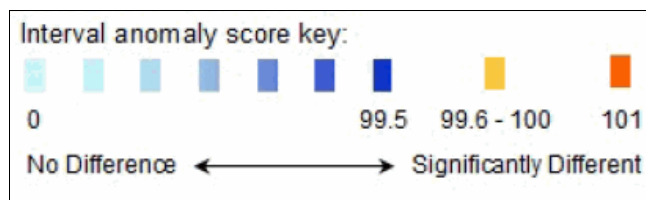


Figure 2-29 Interval anomaly score key

Note that there are two sets of anomaly scores. Each *message ID* in an interval is assigned an anomaly score (which is a value between 0 and 1). Additionally, the *interval* is assigned an anomaly score (which is a value between 0 and 101). The interval anomaly score for an interval is the one that is reflected in the color of the bar for that interval.

The interval anomaly score is based on the anomaly scores of the messages in that interval. However, it is not simply the sum of the individual message anomaly scores. Rather, the interval anomaly score represents an overall view of how widely the messages in the interval differed from the normal behavior for that system. A high score indicates unusual message IDs or unusual patterns of message IDs compared to the system model.

The anomaly score for a specific message ID is shown on the Interval View window (which is described in “Interval View window” on page 79). It is an indication of how different the behavior of the specific message ID is from the normal behavior of the message ID during the training period.

During training, IBM zAware determines whether this message ID is issued as part of a pattern of messages, how frequently within a 10-minute interval it is issued, and how many intervals contain the unique message ID. If the message is issued more often within a 10-minute interval than observed in the past, the score will be higher. If the message did not occur in many intervals or not in any intervals in the past, then the score will be higher. If the message is usually issued as part of a pattern of messages but is issued by itself, the score will be higher.

The observed behavior is fitted to two different statistical distributions and these distributions are used to calculate a portion of the message anomaly score:

- ▶ The Poisson distribution, applied to the likelihood of a message appearing in a group of messages
- ▶ The Bernoulli distribution, applied here to the probability of a message occurring or not occurring

IBM domain knowledge is combined with the observed behavior of the message traffic to construct a final message anomaly score. IBM adds their domain knowledge to make sure that certain critical messages like those associated with actions like sysplex partitioning or the occurrence of an SVC dump receive the correct score, no matter what the behavior of the z/OS image was during the training period.

For example, the messages issued when an SVC dump is taken are marked as important and generate a higher message anomaly score even if during the training period a large number of SVC dumps occurred (this would normally reduce the score associated with those messages). Similarly, message IDs that appear to occur at random in most, if not all, intervals and that reflect random normal behavior of the z/OS image are identified and marked so that the message anomaly score for that message ID is low.

The interval bars on the GUI normally represent a 10-minute interval. The most recent bar is updated every two minutes and might represent less than 10 minutes of data. Every two minutes, IBM zAware analyzes the data for the last two minutes. Depending on the results of that analysis, the color and height of the bar representing the current interval might change. For example it might be light blue at the beginning of the period, then dark blue, and then orange. At the end of the 10-minute interval, the XML that represents the information behind each bar is saved in the IBM zAware file system and will never be updated again. The XML files are kept for the length of time specified in the “Analysis results retention time” field on the Analytics tab of the Configuration Settings window.

Note that the 10-minute period is a fixed window for analysis. At the beginning of each interval, IBM zAware starts its analysis again, but does not carry any information forward from the last interval. It also does not use any information for previous intervals from the instrumentation data database. This means that a cluster of messages that spans two 10-minute intervals might appear to not match any cluster in the model. This could affect the anomaly score of those messages and the interval anomaly score for the two intervals, by making them higher than they would normally be.

To investigate a problem at a specific time, select the bars around that time to determine whether IBM zAware has detected any anomalous message activity. However, to verify that everything is working as expected, select the bars that are dark blue, yellow, or orange to determine what it is about those intervals that makes them significantly different than the normal behavior of that system.

Interval View window

After you identify the interval or intervals that you are interested in, click the bar representing that interval. This brings you to the Interval View window shown in Figure 2-30.

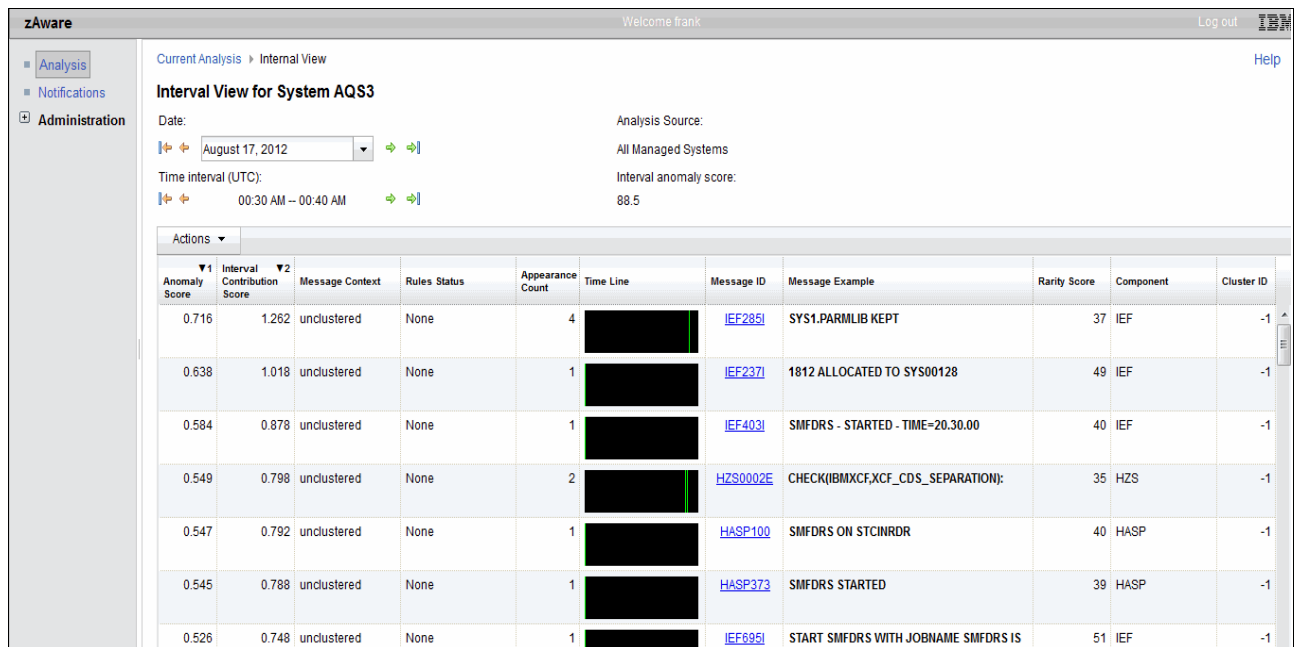


Figure 2-30 Interval View window

The Interval View contains a row for each message ID that occurred in that interval. Each message ID has a time line that shows where the message occurred in the 10-minute period. If the message ID appeared multiple times in the interval, there will be only one row for that message ID but there will be multiple lines in the time line column, one line for each occurrence of the message.

The variables that affect the message anomaly scores are shown in the Interval View window. The associated columns are explained in Table 2-4.

Table 2-4 Interval View descriptions

Column heading	Description
Interval Contribution Score	This shows how this message has contributed to the overall anomaly score for this interval. The higher the Interval Contribution Score, the more influence that message has on the interval anomaly score.

Column heading	Description
Message Context	This indicates whether this message appears in any clusters contained in the model. The possible values are: <ul style="list-style-type: none"> ► NEW - This means that this message was not found in the model database for this system. ► UNCLUSTERED - This means that the message is in the model, but IBM zAware did not identify any other messages that always appear together with this message. ► IN_CONTEXT - This means that the message has appeared in a cluster in the model and in this interval, the message was issued together with the other messages in that cluster. ► OUT_OF_CONTEXT - This means that this message was issued in the interval, but other messages in the cluster were not issued in this interval.
Rules Status	This shows the application of domain knowledge, where metadata has been stored about the context of a message. The possible values (in order of significance) are: <ul style="list-style-type: none"> ► CRITICAL - the rule indicates that this message might be important when diagnosing a problem. ► IMPORTANT - the rule identifies this message is likely to indicate a problem. ► INTERESTING - the rule identifies this message is indicative of a diagnostically useful event, such as a health check exception. ► None - no rule was found for this message. ► NON-INTERESTING - the rule for this message indicates that it is not a significant message.
Appearance Count	This shows how often the message appeared in the 10-minute interval ^a .
Rarity score	This is an indicator of how often this message was issued during the model interval. A score of 101 means that the message was not found in the model. A low score means that this message was found frequently in the model.
Cluster ID	If this message is part of a cluster, the cluster ID identifies the cluster. All the messages in a cluster have the same Cluster ID.

a. Some message IDs can be issued with differing message text. Because IBM zAware does not look at the text of a message, if the same message ID is issued twice, with two different message texts, the message ID will have an Appearance Count of "2".

The messages in the interval view are presented sorted on their anomaly score, with the most anomalous messages at the top. However, you can sort on any column by clicking the heading for each column and selecting whether you want to sort in ascending or descending order. Using the ability to sort the report based on each column, you can easily find the following items:

- Messages that IBM zAware observed did not exist in the model for that system (by sorting the "Rarity Score" column).

These might be interesting candidates to add to your automation or your message suppression table. They might also be messages warning you of the existence of a single point of failure, for example.

- Messages that were issued a large number of times (by sorting the “Appearance Count” column).

These can help you identify a problem in a particular component or with a particular data set.

- Messages that might be impacting your system (by sorting the “Anomaly Score” and “Interval Contribution Score” columns).

These can make you aware of messages that IBM has determined to be high impact (based on its domain knowledge about specific message IDs) or messages that IBM zAware has determined to be highly anomalous. It can also highlight messages that are normal messages, but that are being issued much more frequently than normal.

- The relationship between different components (by sorting the “Time Line” column).

By showing the messages in the chronological order that they were issued, you might spot relationships between messages or components that you were not aware of.

- You can also use the “Cluster ID” column. Messages that IBM zAware has determined are issued as a group will have the same Cluster ID.

If messages from different components (shown in the “Component” column) have the same Cluster ID, that might indicate a relationship that you would not expect. Message IDs with a Cluster ID of -1 are messages that are not clustered; that is, they do not appear to be consistently related to other messages.

For a detailed description of the meaning of the columns in the interval view, refer to *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

2.6.3 Achieving maximum benefit from IBM zAware

Use the following tips to quickly achieve the maximum benefit from IBM zAware:

- Use bulk load to prime the data.

Using bulk load to prime the data allows IBM zAware to start performing message analysis as soon as the training is done.

If you do not use bulk load to prime the data, IBM zAware needs to upload the data in realtime and training cannot occur until enough data is available to create a valid model.

Because 90 days of data is recommended for training to begin, performing a bulk load can save a considerable amount of time.

- Rebuild the model by manually training, when necessary.

If there have been changes in the message traffic on your system, determine whether they are caused by changes such as a new release of the operating system, new or changed middleware, or new or changed applications. If IBM zAware is flagging the new messages as anomalous and you are satisfied that the messages are normal, you can initiate a training of that system to update the model.

Retraining consideration: You must wait until at least after midnight of the day the anomalies occurred because training uses only full days of data. The effect of retraining depends on the stability of the system and the degree to which the new messages cluster. Therefore, waiting more than one day to collect additional data allows the patterns to be more obvious, which can result in a more valid model.

- Exclude days containing unusual events from the model.

If an incident or an outage occurs for which you do not want the anomalous behavior to become a part of the normal model, you can exclude those days from the next IBM zAware training. If the anomalous behavior is allowed to be a part of the model, similar incidents in the future might not be recognized as an anomaly by IBM zAware.

to do this, you need to be cognizant of when the next model is scheduled to occur so that you can exclude the days desired before training occurs automatically.

- ▶ Do not worry about “holes” in your data.

If connections are quiesced or not reconnected, messages issued during that period will not be in the instrumentation data database, and therefore will not be included in the model the next time the model is updated. However, this is no cause for concern. The data used by IBM zAware should not be considered to be critical to things like transaction logs. Because using at least 90 days of data is recommended, missing data for a few hours does not invalidate the model.

2.7 Message analysis with the various functions and products

In 2.4, “Runtime Diagnostics” on page 32, the seven types of behavior detected by Runtime Diagnostics are discussed, including component message analysis.

In 2.5, “Predictive Failure Analysis” on page 42, the six PFA checks are discussed, including the message arrival rate.

In 2.6, “IBM System z Advanced Workload Analysis Reporter” on page 71, how IBM zAware’s analysis uses OPERLOG messages to detect anomalous system behavior is discussed.

Those sections highlight that there are three separate functions and products that all use messages. This might seem confusing, leading you to the following questions:

- ▶ Why are there multiple functions and products that use messages in their analysis and diagnostic behavior?
- ▶ What are their differences?
- ▶ Which one should I be using and in what situations?

As stated in 1.2.4, “Analytics can help address these issues” on page 6, it takes more than one view of your system to determine whether it is functioning properly. The individual functions and products use different types of diagnostics and analysis to achieve the same ultimate goal, which is high availability on your systems and reduced mean time to recovery. This section focuses on the differences in these functions and products as they relate to their message processing.

2.7.1 Runtime Diagnostics critical message analysis

Runtime Diagnostics is intended to be used when you think something might be wrong on your system. It does quick, on-demand, processing to help you *diagnose* system problems. It determines whether critical messages were issued in the last hour, or if the other Runtime Diagnostics events related to global resource contention or address space execution exist.

The critical component message analysis in Runtime Diagnostics answers the following questions:

- ▶ In the last hour of OPERLOG data, do any of the messages in the IBM-defined list exist?

- For the messages for which additional analysis is done, are the combinations of messages and other things required occurring?

This list of messages that Runtime Diagnostics detects was compiled by IBM development and service personnel based on the messages most often found in dumps sent to IBM from systems that experienced failures. It also contains messages identified by component owners who deemed the messages to be indicative that a critical function was experiencing a failure. This list is updated as necessary so that Runtime Diagnostics can detect more and more critical component events.

Because OPERLOG is used in the message processing, Runtime Diagnostics can search the OPERLOG for messages originating on any system in the sysplex by specifying the system name on the SYSNAME parameter when Runtime Diagnostics is invoked. However, to determine whether any system in your sysplex has issued these messages in the last hour, you need to issue the **f hzr,analyze** command for each system in your sysplex.

Runtime Diagnostics does not continually monitor for these messages and does not perform any processing until invoked.

Armed with the results of Runtime Diagnostics, you then have an idea where to perform deeper diagnostics to resolve the problem. The results are also often used as preparation for a “bridge call.”

Runtime Diagnostics message processing information:

- Runtime Diagnostics looks for a hardcoded list of messages in the last hour of the OPERLOG.
- Runtime Diagnostics is intended to be used “after the fact” when you think a system problem exists.

2.7.2 PFA message arrival rate check

The PFA message arrival rate check is designed to detect a potentially damaged address space or system based on the answer to the following question:

- Based on the trends created with 1 hour of data, 24 hours of data, and 7 days of data, is the current message arrival rate within the range expected at this time?

The PFA message arrival rate check relies solely on the rate of WTO and WTOR messages within the collection interval, divided by the amount of CPU used in the collection interval for the system on which PFA is running.

By performing analysis and comparisons on the message rate, PFA can detect whether the current rate is too high (which could mean that an address space or the entire system is damaged) or too low (which could mean that an address space or the entire system is hung).

The model used in the comparisons is a *predictive* model based on current trends. That is, the model is updated on a regular basis to incorporate the most recent trends over three time ranges to detect workload changes. It provides you with the list of address spaces that might be the cause of the problem to use to help diagnose the problem, and proactively alerts you to the abnormal behavior using exceptions through IBM Health Checker for z/OS.

This check applies domain knowledge by excluding all messages issued by any address space starting with JES by creating an EXCLUDED_JOBS file in the check's /config directory when PFA is installed. You can customize PFA to exclude messages from other address spaces by adding them to this file as needed.

The message arrival rate check is not aware of individual messages or patterns of messages. It detects an abnormality based on the calculated rate being too high or too low when compared to the expected rates. Three time range predictions are generated to reduce unwarranted exceptions due to workload changes.

PFA message processing information:

- ▶ PFA looks for abnormally high or low message arrival *rates*, normalized by CPU utilization, to detect address spaces or the system being hung or damaged.
- ▶ PFA includes the current data in the historical data when creating a new model so that current, predictive trends can be established.
- ▶ PFA is intended to detect and alert you to the abnormal behavior before it can be seen externally and to provide the potential address spaces that are the source of the problem to help you diagnose the problem.

2.7.3 IBM zAware message analysis

IBM zAware message analysis uses message IDs and pattern recognition. The data used in the training creates a model of the normal behavior of the message traffic on the system.

Using the normal behavior as the model, IBM zAware can detect anomalies due to the following reasons:

- ▶ Messages not previously found in the historical data.
- ▶ Messages that are rare when compared to the historical data.
- ▶ A specific message ID is issued too frequently when compared to its historical data.
- ▶ Messages that are out of context when compared to the patterns found in the historical data

IBM zAware's message analysis answers the following question:

- ▶ Was the message activity in the interval significantly different than the model of normal behavior?

This processing can detect anomalies on the system that can eventually result in a system outage or a soft failure. It can also help you diagnose problems on your system when you think a problem is occurring by highlighting messages that might otherwise have gone unnoticed. For example, if z/OS system programmers are examining the system log for unusual messages, would they know which IBM DB2® messages are unusual? IBM zAware detects anomalous messages and brings them to the attention of anyone who looks at it. The programmers might not know what the message indicates, but can at least bring it to the attention of their DB2 colleagues.

IBM zAware is independent of the operating system, which allows it to be used even when the systems being investigated are not operating, assuming that the network connectivity to let you logon to IBM zAware is in place. And if it is not in place, that in itself could be valuable diagnostic information.

IBM zAware message processing information:

- ▶ IBM zAware understands message IDs and message patterns.
- ▶ IBM zAware resides outside of z/OS, which means it can be used when the system is non-functional.
- ▶ IBM zAware analysis is based on models that depict historical normal behavior for the system, rather than on predictive trends.
- ▶ IBM zAware is intended to be used to validate major software changes, detect anomalies before they are externally visible, and help identify the cause of the problem after the fact when you think the system is experiencing a problem.

2.7.4 Comparison summary

Each of these tools has specific strengths and is designed to be used in particular circumstances. This section provides this information in a table format to make it easier to compare one product or function to another.

Message analysis detailed comparison

Table 2-5 compares the message processing performed by Runtime Diagnostics, PFA, and IBM zAware.

Table 2-5 Comparisons of Runtime Diagnostics, PFA, and IBM zAware message analysis processing

Function	Metric used	Data source	Intent	Analysis method	Output
Runtime Diagnostics	Hardcoded list of critical component messages.	<ul style="list-style-type: none">▶ OPERLOG▶ LPAR-specific	<i>Diagnose</i> a problem by finding specific messages in OPERLOG after event was reported.	When invoked, search OPERLOG back for 1 hour (if available) to find a critical message on the targeted system and perform additional analysis on some of them.	Runtime Diagnostics Critical Message Event sent to the console (default) with actions suggested.

Function	Metric used	Data source	Intent	Analysis method	Output
PFA	Rate of the count of WTO and WTORs, divided by the CPU used in the collection interval. (Not counted by individual message ID).	<ul style="list-style-type: none"> ▶ Console address space (before Message Flooding Automation) ▶ LPAR-specific 	<ul style="list-style-type: none"> ▶ <i>Detect</i> and <i>alert</i> you to a problem before it is externally visible by detecting potentially damaged address spaces or system and issuing a WTO (which could be processed by your automation product) before your business is impacted. ▶ Helps you <i>diagnose</i> a problem by identifying the potential address spaces causing the problem. Information is externalized using IBM Health Checker for z/OS. 	<ul style="list-style-type: none"> ▶ Predictive, trending model of historical and current rates stored for three time ranges (to distinguish workload variances) creates an expected rate for this point in time. ▶ Applies statistical comparison algorithms using the current rates and expected rates and issues an exception if the current rate is statistically abnormal when compared to the expected rate. ▶ Invokes Runtime Diagnostics to corroborate “too low” condition. ▶ Compares top 10 individual jobs. ▶ Compares other persistent jobs. ▶ Compares non-persistent jobs. ▶ Compares total system. ▶ Performs comparisons every 15 minutes by default. 	IBM Health Checker for z/OS exception sent as WTO (default).
IBM zAware	Scores based on message pattern analysis, using message IDs and message clusters, against learned historical message data for each system.	<ul style="list-style-type: none"> ▶ OPERLOG (before Message Flooding Automation). ▶ LPAR-specific 	<ul style="list-style-type: none"> ▶ Make sure system is likely to work (<i>validate</i> new workloads, new releases). ▶ <i>Detect</i> events before your business is impacted by detecting message anomalies. ▶ <i>Diagnose</i> a problem by identifying the potential cause of the problem after event was reported. 	<ul style="list-style-type: none"> ▶ Creates a behavioral model of normal, historical behavior based on message ID and by message pattern. ▶ Applies analytics to determine how far message activity deviates from the model of normal behavior. ▶ Results updated every two minutes. 	<ul style="list-style-type: none"> ▶ IBM zAware GUI shows scores and results. ▶ Can view all LPARs at once. ▶ API can be used to make results visible to automation products.

Message analysis functions individual benefits

Runtime Diagnostics, PFA, and IBM zAware all provide useful functions to help you achieve high availability. But each performs different processing and offers individual benefits, as listed in Table 2-6.

Table 2-6 Runtime Diagnostics, PFA, and IBM zAware message analysis individual benefits

Function	Benefits	Other considerations
Runtime Diagnostics	<ul style="list-style-type: none"> ▶ Simple setup. ▶ Simple command interface. ▶ Output provides recommended actions. ▶ Can search through volumes of messages and isolate critical messages quickly. ▶ Only consumes CPU time when needed. ▶ Applies domain knowledge. 	<ul style="list-style-type: none"> ▶ IBM-defined list is not modifiable. ▶ No proactive detection capability (diagnostic only). ▶ Returns only messages found in last hour (must be used in real-time immediately after symptoms detected). ▶ Not able to use if system is unresponsive. ▶ Must be run for each LPAR (when needed).
PFA	<ul style="list-style-type: none"> ▶ Reasonably simple setup. ▶ Identifies address spaces potentially causing problem. ▶ Can detect potentially hung address space or system based on rate that is too high or too low. ▶ Attempts to understand workload changes across time so that unwarranted exceptions are not produced when workloads change. ▶ Alerts operator through WTO (by default). ▶ Alert can be acted on by your automation product. ▶ Can be customized to exclude erratic address spaces from all processing. ▶ Requires minimal data to model and little storage for persistence. ▶ Counts all WTO and WTOR messages for this LPAR, including ISV and application-generated messages (except for address spaces specifically excluded) ▶ Self-learning based on individual system workload. ▶ Remodels as necessary to improve understanding of recent trends. ▶ Exclusion list is modifiable. ▶ Applies domain knowledge and excludes some address spaces by default (such as JES* address spaces and CONSOLE, which cause “noise”). 	<ul style="list-style-type: none"> ▶ Relies solely on rates - has no message ID knowledge. ▶ Has no understanding of which messages are critical. ▶ Does not provide an option to exclude times when problems were occurring from next model. ▶ Requires z/OS system resources to run ▶ Analysis occurs after every collection (15 minutes by default), but is configurable ▶ Not able to use if system unresponsive. ▶ Must be running on each LPAR you want analyzed.

Function	Benefits	Other considerations
IBM zAware	<ul style="list-style-type: none"> ▶ Independent of the operating system; can be used even if system is unresponsive. ▶ Consumes minimal additional z/OS system resources. ▶ Analyzes by message ID and the group of messages that they normally appear with (this is referred to as “clustering” in IBM zAware terminology). ▶ Analyzes messages by count to determine whether rate is higher than expected ▶ Can analyze large volumes of messages and isolate intervals of anomalous behavior quickly. ▶ Finds anomalous behavior of newly changed workloads, newly installed products, or changes due to a new release based on message ID and patterns. ▶ Tells you about <i>anomalies</i>, in contrast to Runtime Diagnostics, which looks for a set of known problems. IBM zAware tells you what is different, and you determine whether it is a problem. ▶ Near real-time analysis with the IBM zAware GUI, which is refreshed every two minutes. ▶ The IBM zAware APIs can be used to automatically monitor for intervals with an anomaly score above a threshold that you specify. ▶ Can exclude data from next model by day. ▶ Groups analysis results by sysplex (not consolidated; each LPAR is analyzed separately). ▶ Analyzes all messages in OPERLOG for each LPAR, including ISV, suppressed, and application-generated messages. ▶ Model can be updated to reflect a “new normal.” ▶ Analysis of messages can be done for problems that started days or weeks ago (if datis available). ▶ Applies domain knowledge; gives more weight to critical messages and removes specific messages as “noise” (such as JES2 messages). 	<ul style="list-style-type: none"> ▶ More effort required to set up than Runtime Diagnostics or PFA. ▶ Cannot exclude specific address spaces or specific messages. ▶ Cannot detect message traffic that is unusually low or messages that are missing. ▶ The message text listed in the interval view might not be representative of the message related to the problem (for message IDs that have multiple possible message texts). ▶ Requires more data for training and more storage for persistence. ▶ Must be retrained if workloads or system changes significantly from model. ▶ Each LPAR analyzed must be connected to IBM zAware. ▶ No automated alerts (by default), but can be automated through APIs and XML output.

2.8 Additional tips to achieve high availability

The following section contains additional tips for using the various functions and products to help you achieve high availability.

- ▶ Use IBM Health Checker for z/OS to *avoid* problems.
 - Start IBM Health Checker for z/OS at IPL.
 - Do not ignore the results of the migration checks.
 - Investigate exceptions and take appropriate action. Aim to have Health Checker run with zero exceptions.
 - Automate exceptions issued by IBM Health Checker for z/OS.
- ▶ Use Runtime Diagnostics to quickly *diagnose* problems.
 - Invoke Runtime diagnostics when:
 - The help desk or operations staff reports a problem on the system.

- You need to get ready for the “bridge call” or for a meeting in the “war room.”
- PFA detects abnormal behavior.

In z/OS 1.13, start Runtime Diagnostics at IPL. If you do not start Runtime Diagnostics, PFA cannot detect when an address space or the system might be hung.

- ▶ Use PFA to detect and alert you to potential failures.
 - Investigate PFA exceptions and take appropriate action based on PFA reports.
 - Tune PFA checks for your installation.
 - Automate the PFA Health Check exceptions after the checks have been tuned.
- ▶ Use the z/OSMF Incident Log to help you gather diagnostic data and send it to IBM service.
- ▶ Use IBM zAware to detect message anomalies and diagnose problems
 - Investigate occurrences of message anomalies.
 - Retrain IBM zAware after anomalies caused by changed environments are investigated and at least one full day has passed.
 - Use IBM zAware when a system problem is detected and you need to pinpoint when the problem started to occur and what the source of the problem could be.
 - Use IBM zAware to get an indication whether the problem is affecting multiple systems or only a single system, or whether anomalous message activity on another system might be related to the problem you are investigating.
 - Use IBM zAware when the system is unresponsive.
- ▶ Use the availability tools provided by z/OS components to find problems within those components.
- ▶ Automate critical messages.
 - Keep your automation current. Automation can only detect messages you define to it.
 - Automate new messages each release.
 - Automate new messages delivered through PTFs.
 - Automate z/OS, middleware, ISV, and application messages.

2.9 Sample scenarios

The following section defines sample problem scenarios and suggests the functions and products you can use to detect and diagnose the problem.

Note that providing sample scenarios for z/OS component functions and specific examples of using z/OSMF or the z/OSMF incident log task are beyond the scope of this book. There is also little mention in this section of scenarios using IBM Health Checker for z/OS. This does not imply that these functions are less important, but they are not the focus of this document. For detailed information about those topics, consult the following publications:

- ▶ *z/OSMF Configuration Guide*, SA38-0652
- ▶ *z/OS Management Facility*, SG24-7851
- ▶ *IBM Health Checker for z/OS*, SA22-7994
- ▶ *Exploiting the IBM Health Checker for z/OS Infrastructure*, REDP-4590

Also refer to the *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623, for more detailed information about using IBM zAware and its GUI interface for detecting and diagnosing system problems.

2.9.1 The system is unresponsive

If the system is too sick for Runtime Diagnostics, PFA, or any other functions or products dependent on the operating system to be operational, IBM zAware is available for diagnostics to see if anomalous messages were issued on the failing system.

Using IBM zAware's GUI, identify times of anomalous behavior to help identify the source of the problem.

2.9.2 A sysplex problem exists in which all LPARs are affected

Assume that you have a large sysplex of more than twenty LPARs. All the LPARs seem to be exhibiting abnormal behavior, but PFA has not issued any messages.

Searching the OPERLOG yourself is an option if you know exactly what to search for, or you are able to digest the huge number of messages a large sysplex can produce in a short time. Unfortunately, the root cause of the problem might be one message buried among thousands and thousands of messages.

Runtime Diagnostics might be helpful, but to obtain the full picture of where the problem originated, you might have to invoke Runtime Diagnostics on every LPAR. And the message indicating the issue might not be one of the messages Runtime Diagnostics detects, or it might have been issued more than an hour ago.

You could simply IPL the LPARs one at a time until the problem is resolved. But this activity is quite disruptive and history shows that you would need to IPL more than half of the LPARs on average to find the source.

In this case, IBM zAware is the most logical place to start diagnosing this scenario. In IBM zAware's Analysis view, there might be messages that are issued frequently, causing the bars in the IBM zAware output to be orange in color the closer you get to the current time, because there are now multiple problems occurring.

However, the source of the problem might be more readily identified by looking for a bar *prior* to the problem being known that identifies the following anomalous behavior:

- ▶ A unique message; that is, a message that did not exist in the model database was issued
- ▶ One or more out-of- context messages was issued
- ▶ Some message was issued more frequently than is normal

The fact that the IBM zAware GUI shows all LPARs in a sysplex makes it easier to identify a problem that exists across the sysplex. It can also help you identify a trigger event on a system other than the one that is currently having a problem.

2.9.3 Software changes have been made to your system

Both IBM Health Checker for z/OS and IBM zAware provide useful, but different, ways to avoid and detect abnormal behavior after significant software changes on your system.

Use IBM Health Checker for z/OS to avoid problems

IBM Health Checker for z/OS can detect configuration settings that are not compatible if a check exists for it. If the software change is a new release of z/OS, many checks exist in the operating system to detect whether migration actions have been performed. Do not ignore the results of these checks, as they are provided specifically to help you avoid problems after migration.

Use IBM zAware to detect and diagnose problems

IBM zAware can be quite useful in helping you to determine whether software changes in your installation are behaving correctly. Consult IBM zAware whenever a significant change is made in your installation.

When new software levels are installed, or when system settings or the system configuration have been altered, use IBM zAware to answer the following questions:

- ▶ Are new, unusual messages being issued during periods immediately following the changes to your system?
- ▶ Are more messages issued than expected?

When you are satisfied with the behavior of your new environment, use IBM zAware training functions to update the model. The newly collected data is included in the new model so that IBM zAware does not erroneously identify new normal messages as being anomalous.

2.9.4 IBM zAware detects an anomaly

If IBM zAware detects an anomaly (as reported by the height and color of the bar on the GUI), the cause might be a message that has never been seen before, a message that appears out of context, or a message that occurs more times than expected. Use the IBM zAware GUI to drill down to obtain more details about the anomaly:

- ▶ Investigate the messages with the largest interval contribution score.
- ▶ Check to see whether the messages were for the same component, subsystem, or product. This might indicate the component, subsystem, or product is misbehaving and might need to be restarted.

For additional information, use Runtime Diagnostics on the system exhibiting the message anomaly.

- ▶ Correlate Runtime Diagnostics events with the message anomalies.
- ▶ Perform actions recommended in the Runtime Diagnostics events.

2.9.5 PFA message arrival rate check exception issued for a high rate

When the PFA message arrival rate check issues an exception for a message rate that is too high, the exception typically occurs due to a burst of messages. PFA can detect the address space causing the burst in some cases, but it has no knowledge of the message IDs that were issued.

IBM zAware complements the PFA message arrival rate check because it provides the message IDs and the exact times of those messages, and provides deeper analysis to help you identify the source of the burst that was detected by PFA.

In addition, a high message arrival rate might actually be sympathy sickness caused by another problem. Using IBM zAware, you can identify whether there were any unique messages or other anomalies prior to the high rate.

For further problem determination, Runtime Diagnostics can be used to identify whether there were other causes of the high rate of messages. For example, an address space that is looping might be the source of the high message arrival rate.

When an exception occurs in the PFA message arrival rate check for a rate that is too high, follow these steps:

1. Examine the PFA report in SDSF.
 - Note the time of the exception.
 - Note the address spaces listed.
2. In IBM zAware, find the LPAR for which PFA issued the exception.
 - Using the time of the exception noted in step 1, find the interval in which the exception occurred and drill down to determine more information about the anomaly in that interval.
 - On the sysplex view, look at the message intervals prior to the exception. Determine if there were any unique messages identified or any other anomalous behaviors that could be the true source of the error.
 - Review the message details in the message documentation for the messages identified. Follow the directions provided by the message to continue to diagnose and fix the problem.
3. If the problem has not yet been identified or resolved, use Runtime Diagnostics to see if there is another reason for the high message arrival rate.
 - Invoke Runtime Diagnostics on the system that issued the PFA exception.
 - Determine whether any events are issued for the address spaces identified in step 1 or if any events have been issued for other address spaces for reasons that could be the source of the high message arrival rate.

2.9.6 PFA exception issued for a low rate

When PFA issues an exception for a message arrival rate that is too low, an SMF arrival rate that is too low, or an enqueue request rate that is too low, Runtime Diagnostics has already been invoked and the results are in the report available in SDSF. Therefore, the starting place for your investigation should be the PFA report.

Most often for a rate that is too low, Runtime Diagnostics detects some form of contention such as ENQ contention, GRS latch contention, or z/OS UNIX latch contention, or it detects high local lock suspension.

If the problem seems to be sympathy sickness, use IBM zAware to determine whether there was a unique message prior to the PFA exception that could identify the source of the problem as described in step 2 of section 2.9.5, “PFA message arrival rate check exception issued for a high rate” on page 91.

2.9.7 PFA exception issued for a high SMF arrival or high ENQ request rate

When PFA issues an exception for an SMF arrival rate that is too high or an enqueue request rate that is too high, the starting place for your investigation should be the PFA report. PFA attempts to identify the address spaces that are the cause of the high rates and prints them on the report.

For further problem determination, Runtime Diagnostics can be used to identify whether there were other causes for the high rates. For example, an address space that is looping might be the source of the high SMF arrival rate or the high ENQ request rate.

For additional diagnostics, IBM zAware can be consulted to determine whether there are any other message anomalies on the system.

When an exception occurs in the PFA SMF arrival rate check or the ENQ request rate check for a rate that is too high, follow these steps:

1. Examine the PFA report in SDSF.
 - Note the time of the exception.
 - Note the address spaces listed.
If the exception is for the SMF arrival rate check, review the SMF records sent by the address spaces identified on the report and examine the system log to determine what caused the increase in SMF activity.
2. Use Runtime Diagnostics to see whether it detects an event that might be the reason for the high rate.
 - Invoke Runtime Diagnostics on the system that issued the PFA exception.
 - Determine whether any events are issued for the address spaces identified in step 1 or whether events have been issued for other address spaces for reasons that could be the source of the high rate.
 - If Runtime Diagnostics issues an event for the address space with the high rate, the most common events issued are loop events or high CPU events, which is likely the reason for the high rates. The recommended action is to cancel the job.
3. If the problem has not been resolved, use IBM zAware to see whether there any message anomalies prior to the exception.
 - In IBM zAware, find the LPAR for which PFA issued the exception.
 - Using the time of the exception noted in step 1, find the interval in which the exception occurred and drill down to determine more information about the anomaly in that interval.
 - On the sysplex view, look at the message intervals prior to the exception. Determine whether there were any unique messages identified or any other anomalous behaviors that could be the true source of the error.
 - Review the message details in the message documentation for the messages identified. Follow the directions provided by the message to continue to diagnose and fix the problem.

If the problem seems to be sympathy sickness, use IBM zAware to determine whether there was a unique message prior to the PFA exception that could identify the source of the problem, as described in step 2 of 2.9.5, “PFA message arrival rate check exception issued for a high rate” on page 91.

2.9.8 Runtime Diagnostics message event detected

In this scenario, Runtime Diagnostics has been invoked and has issued a critical component message event which provides actions recommended for each message event.

If you need to acquire more information about the problem prior to taking the recommended action (which might be destructive), use IBM zAware to further analyze the interval when the message was actually issued, and the intervals prior to the message being issued.

Because Runtime Diagnostics searches back one hour in the OPERLOG (or less if an hour's worth is not available), it is guaranteed that Runtime Diagnostics message events include only messages that were issued in the hour prior to Runtime Diagnostics being invoked.

2.9.9 PFA and Runtime Diagnostics examples

The following are simply a few examples of PFA exceptions that have occurred on IBM systems used in internal, large test environments.

These systems were not specifically testing PFA and Runtime Diagnostics. That is, they have PFA running simply to detect system abnormalities and Runtime Diagnostics was active so that PFA could invoke it and so that it could be used for additional diagnostics. In other words, these are situations that can also happen in your installation.

- ▶ The PFA_ENQUEUE_REQUEST_RATE check issued an exception for an abnormally high rate.
 - Investigation showed that a HOLDACTION had not been performed after maintenance was applied.
- ▶ The PFA_LOGREC_ARRIVAL_RATE check issued several exceptions while workloads were running at a high stress level.
 - The EREP report showed many OC4 abends in a particular module.
 - The problem was that a HIPER PTF had not been applied.
- ▶ Both the PFA_ENQUEUE_REQUEST_RATE and the PFA_SMF_ARRIVAL_RATE checks indicated that their rates were too low on many LPARs in a 9-way sysplex.
 - The sysplex was experiencing a hang condition in a critical address space.
 - The problem was detected before it would have been seen externally through other mechanisms, and the problem was fixed quickly without requiring an IPL.
- ▶ The PFA_ENQUEUE_REQUEST_RATE issued an exception for the total system ENQ rate being too low because of latch contention in a critical system address space.
 - The problem was resolved without the need for an IPL.
- ▶ The PFA_LOGREC_ARRIVAL_RATE check caught a problem where one of the middleware started tasks was continuously dumping.
- ▶ The PFA_JES_SPOOL_USAGE check caught a problem that at first glance seemed insignificant.
 - There was a task that started, used a little spool space, and then ended.
 - The check sent an exception when the task was using 14 track groups instead of the expected 0 track groups.
 - This exception seemed unwarranted at the time because 14 track groups is so little. However, after further investigation, it was discovered that tracing had been turned on and the tester thought it was subsequently disabled, but it was not.
 - Further investigation discovered that there were over 1 million lines of trace output in the spool for the task.
- ▶ Both the PFA_MESSAGE_ARRIVAL_RATE check and the PFA_ENQUEUE_REQUEST_RATE check detected rates that were too low for a critical, persistent address space.
 - Runtime Diagnostics corroborated the PFA exception by determining that the address space had enqueue contention and was hung.



Planning for an IBM zAware implementation

This chapter provides high-level information that will help you identify the hardware, system setup, and personnel resources required for a successful IBM zAware implementation so you can plan your implementation in advance. When you are ready to start the installation, such preparation will help you avoid unforeseen issues or delays and enable you to derive value from IBM zAware as quickly as possible.

Refer to *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623, to obtain the latest and most detailed information about planning an IBM zAware implementation. The guide is available on IBM ResourceLink

3.1 Planning overview

This chapter summarizes the items to consider to plan for the resources that are needed to complete a successful and timely install of IBM zAware. Each consideration is addressed in more detail in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

3.2 Selecting which systems to monitor with IBM zAware

When selecting systems that are to be monitored with IBM zAware, it is important to bear in mind the capabilities and objectives of IBM zAware, and the importance to your business of the monitored clients.

To maximize the value of the information that it provides to you, IBM zAware attempts to build a model that is as close as possible to an actual representation of the “normal” behavior of that system. Looking at the message activity for simply a few hours or days will not include all the messages that you are likely to see on that system over a month or many months.

Therefore, IBM zAware has a set of criteria that must be met before it will build a model of a system. The requirements at the time of writing are:

- ▶ The set of messages that are used to train an LPAR must contain somewhere between 300 and 400 unique message IDs.
- ▶ Each of those messages must occur at least once in three different intervals during the period that is being analyzed.
- ▶ Production systems tend to show a high degree of message clustering. A message set that results in a large number of single-message clusters means that most messages are appearing individually and not in repeating patterns. This is an indication that IBM zAware could have difficulty identifying anomalous messages and therefore it will not create a model.

For example, a production z/OS system is an obvious candidate for monitoring with IBM zAware. The messages that are produced by those systems are typically many and varied. They also tend to be consistent and are more likely to occur in repeating patterns, and the availability of those systems is vital to the enterprise. These attributes mean that production systems are typically well suited to monitoring by IBM zAware.

Another system type is Quality Assurance (QA). One of the objectives of QA systems is to identify changes between the current release of a software product or an application and a new release. The ability of IBM zAware to identify anomalous messages is ideally suited to that role, so those are also good candidates for monitoring with IBM zAware.

Test and Development systems might be good candidates, depending on how you use them. If they are in the same sysplex as production systems, they might generate messages that are of interest from the perspective of the production systems. Also, if your applications tend to issue console messages, anomalous messages that might be issued due to application changes might be valuable, particularly if those messages identify a situation that can cause a problem in a production environment.

Alternatively, system programmer sandbox systems might seem ideal for initial testing of IBM zAware and getting familiar with it. However, they might not be good candidates for monitoring with IBM zAware because the workload characteristics of those systems are too variable for a model of what is “normal” to be built. Also, the availability of those systems is

not as critical to the enterprise. If you plan to use a system programmer system to drive initial testing of IBM zAware you might find that you need to load more than three months of syslog data to have sufficient message variety to be able to successfully create a model. Even that, however, is not a guarantee that IBM zAware will be able to build a model for that system.

Model creation consideration: If a system does not have enough message diversity or consistent patterns to be able to successfully create a model, you could “cheat” by writing a simple program to generate large numbers of messages. However, although this might be successful from the perspective of getting IBM zAware to create a model of that system, you need to consider whether this is a worthwhile exercise. If the day-to-day message traffic on a system is such that IBM zAware cannot determine what a “normal” day looks like, you will probably find that it will end up reporting many “anomalies” for that system that are in fact simply normal messages.

If you encounter problems trying to get a system to train successfully, refer to 5.5.2, “Creating and updating the system model” on page 168 for further guidance.

Another point to remember is that the monitored clients must have TCP/IP network connectivity to the IBM zAware LPAR. Similarly, any system that will communicate with IBM zAware’s Application Programming Interface (API) will also require TCP/IP access (using secure sockets) to the IBM zAware LPAR.

The monitored clients can be in the same CPC as IBM zAware or in a different CPC. They can also be running under IBM z/VM®. They can be running on any CPC that is supported by z/OS 1.13 or later, and they must be running z/OS 1.13 or later with the enabling PTFs for IBM zAware. Both JES2 and JES3 systems are supported (although the Bulk Data Load utility does not support archive data sets created from a JES3 DLOG).

IBM zAware prerequisites for monitored clients: All monitored clients must meet the following prerequisites:

- ▶ They must be running z/OS 1.13 or later.
- ▶ They must be running on any CPC that is supported by z/OS 1.13.
- ▶ The LOGR CDS must be formatted at the HBB7705 or later level.
- ▶ The APARs to enable System Logger IBM zAware support must be applied.
- ▶ The monitored client must have a TCP/IP connection to the IBM zAware LPAR.

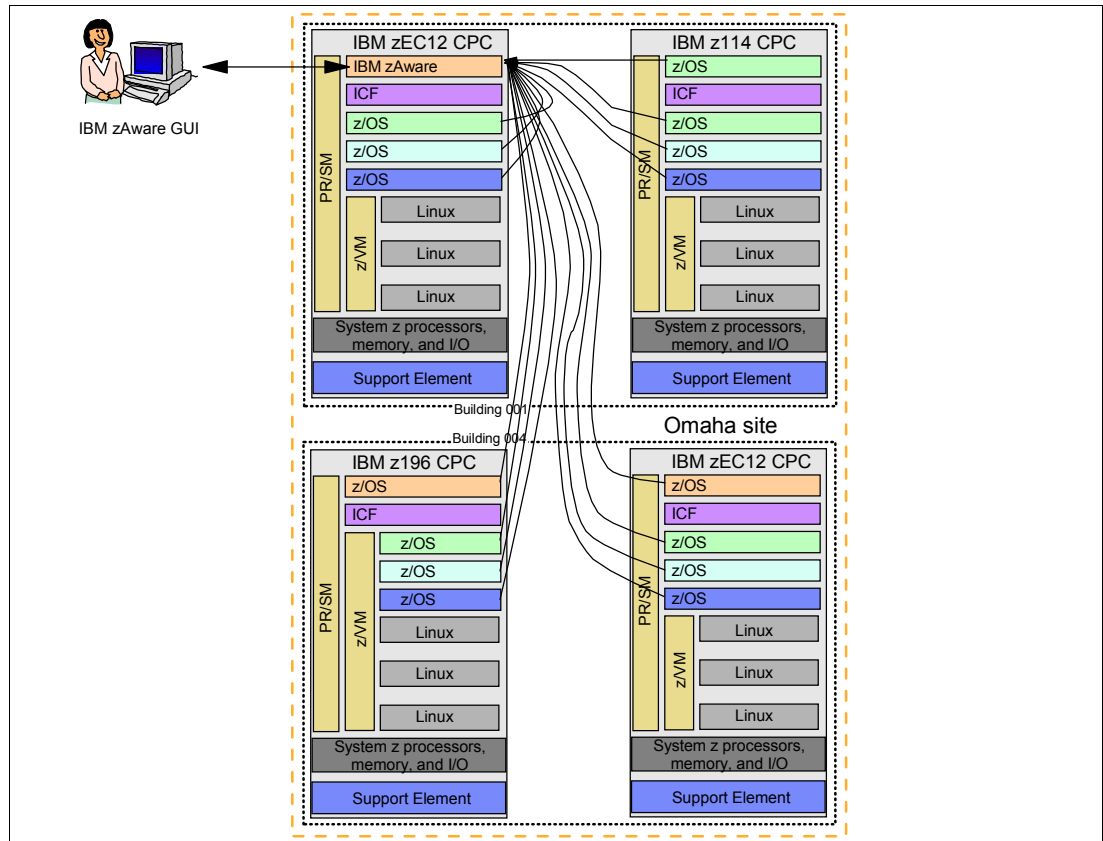


Figure 3-1 Sample IBM zAware configuration

Figure 3-1 contains a sample IBM zAware configuration. You can see that the monitored clients can be in the same CPC as IBM zAware or in different CPCs. They also can be in the same site or in multiple sites. The planning information provided here can help you configure a simple configuration or a more complex configuration such as the one shown.

3.3 Hardware resources

For a smooth and timely implementation of IBM zAware, ensure that the hardware configuration required to support and connect to IBM zAware is available.

3.3.1 CPU capacity

IBM zAware runs in an LPAR on a zEC12 or later CPC. Just like any System z LPAR, IBM zAware requires processing capacity, memory, disk storage, and connectivity.

IBM zAware is able to use either general purpose CPs or IFLs, and they can be shared or dedicated. If your configuration contains both types of PUs, and both have sufficient capacity to support IBM zAware, it is generally more cost effective to configure the IBM zAware LPAR to use IFLs. It is useful to start with shared PUs until you gain experience with IBM zAware and have a more accurate indication of how much capacity it will require to handle your message rates.

IBM zAware LPARs support up to 101 PUs, although it is not expected that anywhere near that much capacity will be required.

LPARs

Each IBM zAware LPAR is able to analyze the message stream from a large number of monitored clients. It is expected that most installations will only require a single IBM zAware LPAR. If you have business reasons for requiring multiple IBM zAware LPARs, it is possible to have multiple IBM zAware LPARs on the same CPC, but each will require its own disks and network connections. The CPU capacity required for each IBM zAware LPAR will depend on the volume of messages that IBM zAware has to process and the number of clients it is monitoring.

It is possible to have both test and production IBM zAware LPARs. Whether you deem this to be necessary depends on your philosophy about separation of test and production systems and why you want to have a test IBM zAware. There are several points to consider when making a decision about whether you will benefit from a test IBM zAware LPAR:

- ▶ Consider *why* you want to have a test IBM zAware LPAR. If your objective is to test the analytics function when you move to a new release of IBM zAware, then bear in mind that the message profile from your test systems is probably different than the message profile from your production ones. This means that any changes that you see with the new release in the test IBM zAware might or might not apply to your production IBM zAware.
- ▶ When you activate IBM zAware firmware updates on a CPC, *all* IBM zAware LPARs on that CPC will automatically re-IPL themselves. This means that it is not possible to run two IBM zAware LPARs with different code levels on the same CPC. So if your intent is to investigate the effect of IBM zAware firmware updates, you will need to run the test and production IBM zAware LPARs on different CPCs.
- ▶ You could use the bulk load utility to load several months' worth of messages from your production systems into the upgraded test IBM zAware. However, the results shown on the analysis window only relate to messages received in real time. No analysis results are shown for the time frame that is covered in the bulk load.

Thus, unless you are willing to bulk load production messages *and* then move your production systems over to the test IBM zAware, you will not be able to see the effect of the new IBM zAware release on your production messages.

Messaging consideration: A monitored client can only connect to one IBM zAware LPAR at a time. This means that you cannot have a z/OS system feeding messages in real time to both test and production IBM zAware LPARs.

- ▶ There is no capability to delete all data for just one system or just one sysplex from IBM zAware. The controls for how long data is kept in IBM zAware apply to all systems and all sysplexes.

This means that if you start with one IBM zAware LPAR and then decide that you want to move a subset of systems to a new IBM zAware LPAR, the data from those systems will remain in the first IBM zAware until it expires.

CPU capacity for IBM zAware LPAR

The amount of CPU capacity required by IBM zAware depends mainly on the volume of message data that is sent to IBM zAware and the number of monitored clients. There are two usage profiles for a IBM zAware LPAR:

- ▶ During normal operation, when IBM zAware is processing messages in real time as they arrive from the monitored clients.
- ▶ Performing a bulk load or updating the model. Additional CPU will be used by IBM zAware at these times. Because the LPAR will require more capacity during this activity, if the

LPAR is constrained, the update of the model or the bulk load will take more time to complete.

Chapter 5, “Maintaining and managing IBM zAware” on page 153 discusses how to identify the most appropriate frequency with which to update the IBM zAware model. Note that the frequency that you specify applies to all monitored clients, that is, it is not specified on a system-by-system basis.

To give you an indication of how much processing capacity is required, Table 3-1 shows the percent of a zEC12 IFL that was consumed for two different message rates and assuming a maximum of ten connected clients.

Table 3-1 CPU consumption of IBM zAware LPAR

Total Message rate per second	Utilization % of 1 zEC12 IFL during normal operation	Utilization % of 1 zEC12 IFL during bulk load and training
500	20	25
1500	40	80

IBM advises attaching no more than 10 monitored images when the maximum message rate in a 15-minute interval is approximately 1500. Otherwise, there is the potential to overrun the capacity of a single zEC12 IFL during initial bulk load and training. You can configure a second IFL to avoid this condition.

Note that these numbers are only rules of thumb for planning purposes. Actual IBM zAware LPAR CPU consumption can vary based on many variables such as the number of monitored clients, the number of unique message IDs, the amount of data in the model database, the amount of storage in the IBM zAware LPAR, the message rate, and so on.

If you start with a single shared engine and use IBM RMF™ to monitor the CPU usage of the IBM zAware LPAR, you should be able to determine the required capacity if you plan to add more monitored clients to the configuration. Note that if you need to increase the number of engines in the LPAR, it is necessary to deactivate and reactivate the IBM zAware LPAR. Even though it is possible to define both INITIAL and RESERVED CPs or IFLs in the LPAR profile, the IBM zAware GUI does not provide a mechanism for dynamically bringing additional engines online.

You can calculate your maximum message rate by selecting a peak workload interval and storing the OPERLOG messages. Then, choose a precise 10-minute interval from the stored peak message traffic and count the messages in the interval. Divide that message count by 600 to obtain the message rate per second. As a general rule of thumb, the upper bound on sustained message rates on z/OS systems averages about .1 message per second per MIPS. This is based on systems running batch, TSO, IBM CICS®, DB2, and IBM IMS™. Systems running WebSphere Application Server might be a little higher. And systems dedicated to IMS or DB2 tend to be a bit lower.

Alternatively, the IBM provided Message Analysis Program helps you identify the average and peak message rates for a system based on an analysis of the file you would use as input to a bulk load. This is discussed in “Capacity planning” on page 208.

CP capacity for z/OS LPARs

The additional z/OS CPU consumption for the monitored clients depends on a number of factors:

- Was that system already using OPERLOG?

- The average and peak number of messages that will be sent to IBM zAware from the monitored client.

IBM ran a number of measurements of systems that were sending data to IBM zAware. The *total* utilization of the z/OS systems varied from about 25 percent of 1 CP up to more than 20 CPs' worth of capacity. The total capacity used by the System Logger address space (which included its normal processing of OPERLOG *plus* its work to send the messages to IBM zAware) varied from .1 percent of 1 CP up to a maximum of 2 percent of 1 CP. Even when a bulk data load was running, System Logger utilization did not exceed 2 percent of 1 CP.

Using a System Logger that was only handling OPERLOG requests as the base, connecting System Logger to the IBM zAware LPAR resulted in roughly doubling the amount of CPU being used by System Logger. If System Logger was handling requests for other log streams, those requests would be unaffected by connecting to IBM zAware. Also, connecting System Logger to IBM zAware had no impact on CPU usage by the Console Address space if the system was already using OPERLOG.

OPERLOG uses Message Data Blocks (MDBs) that the CONSOLE address space is already creating for its own use, so there is no additional overhead involved in creating MDBs for the OPERLOG. The only additional CPU cost associated with the use of OPERLOG (which will be reduced if you are no longer writing to SYSLOG) is the use of System Logger services to write to OPERLOG. The writing to either log is charged to the CONSOLE address space, so you can monitor the CPU usage of that address space before and after turning on OPERLOG if you need to identify the additional cost associated with enabling OPERLOG.

Based on these experiences, we believe that it is fair to say that the impact on z/OS CPU utilization of sending that system's messages to IBM zAware will be negligible.

Memory

IBM zAware requires a base amount of memory for IBM zAware itself. In addition, each monitored client requires an additional amount of memory in the IBM zAware LPAR. At the time of writing, the recommended values are 4 GB for IBM zAware, plus about 256 MB extra for each monitored client. However, always check the level of *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623, that corresponds to your level of IBM zAware to obtain the latest IBM guidance regarding the required amount of memory. IBM zAware supports up to the maximum amount of memory that the CPC supports in a single LPAR.

At the time of writing, IBM zAware does not create any SMF-like information about its own resource utilization. The CPU consumption of the IBM zAware LPAR can be monitored using an RMF CPU report, but there is no mechanism to monitor memory use of the IBM zAware LPAR.

Lack of sufficient CPU capacity or memory is likely to show as poor response times when using the IBM zAware GUI. Adding memory to the IBM zAware LPAR will require the LPAR to be deactivated and reactivated.

Tip: After you have completed the initial setup, adding more monitored clients to IBM zAware is simple. However, it is easy to forget to add memory to the IBM zAware LPAR when you add monitored clients. So it is useful to create a small checklist of the steps needed to add a monitored client and include adding memory to the IBM zAware LPAR in that list. A sample checklist is contained in 5.9, "Checklist for adding a monitored client" on page 179.

3.3.2 Network connectivity

IBM zAware uses TCP/IP to communicate with the systems it monitors. Hipersockets can be used for communication if the monitored clients are in the same CPC as the IBM zAware LPAR.

Communication over an Open Systems Adapter (OSA) can be used if the monitored clients are in the same or a different CPC. IBM zAware supports multiple OSAs, and multiple IP addresses on a given OSA. The OSAs can be on the same or different subnets.

An Intra-Ensemble Data Network (IEDN) can be used to connect to the IBM zAware LPAR if the IBM zAware LPAR resides in a CPC that is part of a IBM zEnterprise® ensemble, and the monitored clients are also part of the ensemble. For more information about IEDNs, refer to the IBM Redbooks publication *Building an Ensemble Using IBM zEnterprise Unified Resource Manager*, SG24-7921.

The network bandwidth that will be required to connect the monitored clients to IBM zAware is based on the peak message rate of the systems. The message rate can be determined by analyzing a representative interval of syslog or OPERLOG data using the Message Analysis Program as described in Appendix A, “Syslog Message Analysis Program” on page 207.

Note that System Logger supports up to 99 GB of buffers for times when the rate of message production exceeds the available network bandwidth. So if the available network capacity is insufficient to handle intermittent message peaks, the only impact might be that it will take a little longer for messages to reach the IBM zAware LPAR. It is not desirable to be in a situation where there is constantly a backlog of messages queuing in the Logger buffers¹.

Regardless of the type of connection you use between the monitored clients and IBM zAware, remember that you will need at least one OSA adapter (preferably two, for availability) for access to the IBM zAware GUI. IBM zAware supports a maximum of 24 OSA adapters.

3.3.3 DASD storage

The IBM zAware LPAR requires CKD disk space² to store the following information about *each* monitored client:

- ▶ A model of the normal message activity for the system
This model information includes not only information about which messages are issued, but also how they are clustered (which messages tend to be issued around the same time). You control how many days of model information are kept in the database using the Training models retention time parameter in the IBM zAware GUI, as shown in Figure 3-2 on page 103. The default is 365 days.
- ▶ Information about messages that have been received since the model was last updated
This information will be used as input to the next time the model is updated. You can control how many days of this information is kept in the database using the Instrumentation data retention time parameter in the IBM zAware GUI. The default is 365 days.
- ▶ The information that is presented on the Analysis window of the IBM zAware GUI.

¹ The System Logger IBM zAware buffers are pageable and reside in above-the-bar private storage. You control the amount of storage for the buffers using the LOGBUFMAX keyword in the IXGCNFxx member of Parmlib.

² IBM zAware requires CKD disk. FBA is not supported.

An XML file is created for each 10-minute interval for each monitored client. The amount of time that these XML files are retained in the database is controlled by the Analysis results retention time parameter in the IBM zAware GUI. The default is 365 days.

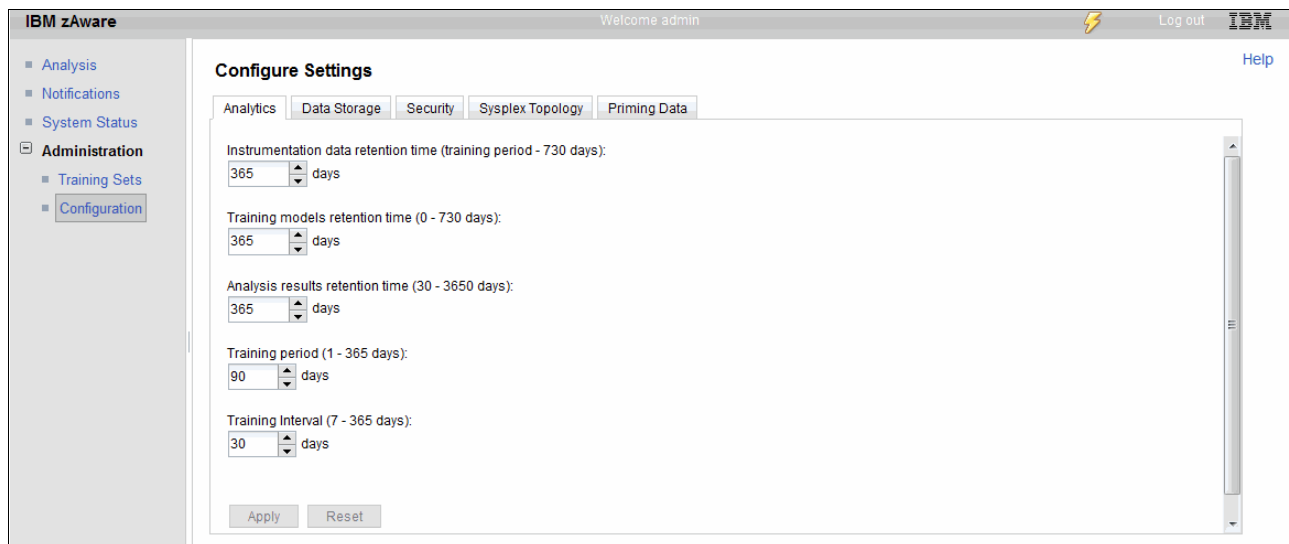


Figure 3-2 Configuring IBM zAware data retention periods

Note that the *actual* incoming messages are not stored in the IBM zAware disk pool. Each message is summarized and analyzed by IBM zAware when it arrives and the instrumentation data database is updated.

Because the amount of disk space that IBM zAware requires is based on the number of *unique* messages, and not the total number of messages, it can be challenging to accurately predict the amount of disk space that is required. Also, there is a relationship (although not a linear one) between the number of unique messages, the number of systems, and the amount of space that is required to hold that information.

Currently, the IBM guideline is to provide 500 GB for the use of IBM zAware, plus an additional 4 to 5 GB per monitored client. However, this value might vary from one system to another; it depends on how many months of information you want IBM zAware to keep. The IBM zAware GUI tells you how full the disk pool is, and you can easily add and remove devices dynamically, so you can fine-tune your configuration based on your own experiences. IBM zAware supports up to 256 CHPIDs, and there is effectively no limit on the amount of disk space that can be connected to it.

When the IBM zAware LPAR is activated, it can see any DASD device that has been defined as accessible to the IBM zAware LPAR in the active IOCDS. You then use the IBM zAware GUI to tell it which of those volumes it should use. IBM zAware supports dynamic I/O reconfiguration, so you can add DASD devices to the IBM zAware LPAR dynamically (assuming there is a z/OS or a z/VM LPAR on the IBM zAware CPC to dynamically activate

the new configuration). Pressing the Refresh button on the Data Storage Devices panel (as shown in Figure 3-3) results in the updated list of devices being displayed.

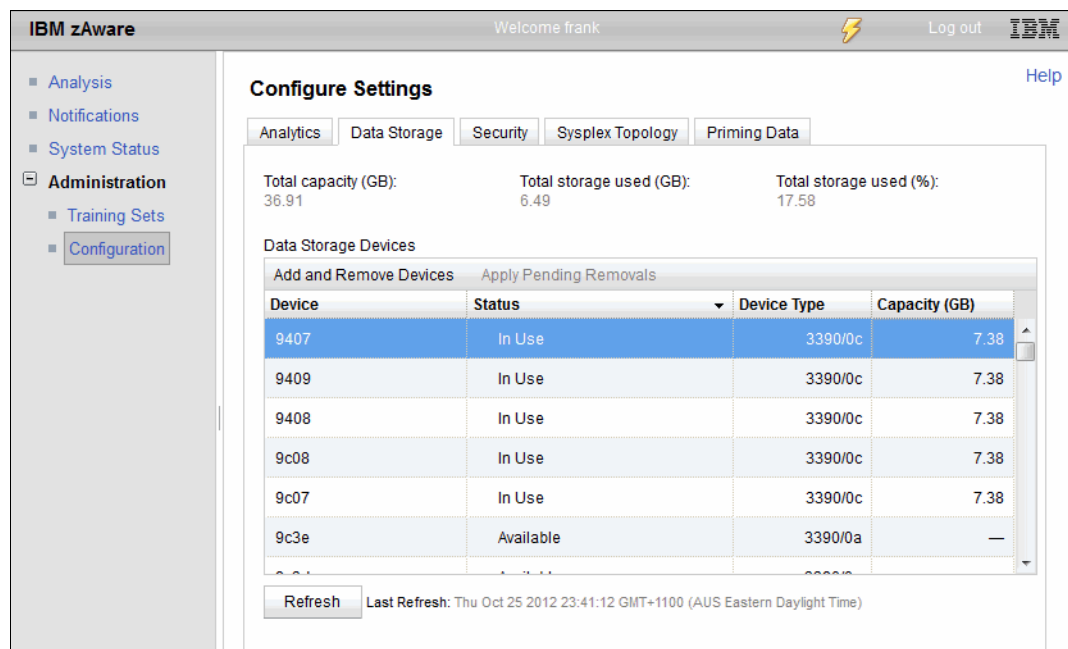


Figure 3-3 IBM zAware Data storage panel

Critical IBM zAware access considerations: When defining the IBM zAware LPAR in HCD, it is *critical* that you ensure the following points:

- ▶ Use a device candidate list to ensure that the IBM zAware LPAR can *only* access its own disks.
- ▶ Use a device candidate list to ensure that *only* the IBM zAware LPAR has access to the IBM zAware disks.
 - The one exception is if you want to back up the IBM zAware disks. In that case, the IBM zAware disks must be accessible to the z/OS system, but should be defined to that system as being offline at IPL. For more information about backing up the IBM zAware data, refer to “Backing up the IBM zAware file system” on page 161.
- ▶ If you have more than one IBM zAware LPAR, each IBM zAware LPAR should only have access to its own set of disks.

If an IBM zAware LPAR can access disks other than its own, it is possible that an IBM zAware user with admin authority could accidentally initialize one of those disks.

Note that using an OS CONFIG (a method frequently used to limit the devices that a z/OS LPAR can access) will *not* work for IBM zAware.

IBM zAware DASD backup considerations

When planning your IBM zAware environment, it is important to bear in mind that the information that resides in IBM zAware is not transactional data. This means that the data in IBM zAware does not have to be managed with the same degree of care that your typical z/OS data receives.

For example, if a system loses connectivity to IBM zAware and some number of console messages are not sent to IBM zAware, the impact of that is likely to be minimal in relation to the volume of messages that *do* reach the IBM zAware LPAR.

Furthermore, it is likely that all the information in IBM zAware can be recovered relatively easily (if necessary) by simply loading historical message information from your syslog archives for all monitored clients. This means that backing up or mirroring your IBM zAware data, as you typically do with all your z/OS data, might not be necessary. The subject of backing up your IBM zAware disks is discussed in more detail in 5.3.4, “Backing up the IBM zAware file system” on page 161.

3.3.4 z/OS requirements

The detailed z/OS requirements for IBM zAware are provided in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623. This section simply summarizes the z/OS considerations.

Providing message history

To provide IBM zAware with a representative view of the normal message traffic for a system, it is ideal to have about 90 days of syslog or OPERLOG data that will be bulk-loaded into the IBM zAware LPAR as part of the setup process. The likelihood of IBM zAware considering a “normal” message is an anomaly increases inversely with the number of days of messages you provide. A small number of days’ worth of data is more likely to result in IBM zAware considering a message to be an anomaly because it has not seen it before.

At the time of writing, the Bulk Data Load Utility does not support the use of a JES3 DLOG archive file. That means that you have two options for getting JES3 message data into IBM zAware to create the model database:

- ▶ If you are using OPERLOG, you can create an archive file using the IEAMDBLG program. The message format in OPERLOG is the same, regardless of whether the system is using JES2 or JES3.
- ▶ You can send message data in real time and wait until you have sent over enough data to be able to build a representative model.

Provide a representative view: Aim to keep at least 90 days of syslog or OPERLOG data to use to prime the IBM zAware database. This will result in more accurate anomaly detection when you start using IBM zAware immediately after installing it.

OPERLOG

The mechanism to move messages from z/OS to the IBM zAware LPAR is provided by System Logger. Therefore, to have messages sent to IBM zAware in real time, OPERLOG *must* be running on every system that you want to monitor with IBM zAware. In a multisystem sysplex, that means that the OPERLOG log stream must be defined as a CF log stream.

You can also use OPERLOG as a DASD-only log stream. However, this method is only suitable for a single system sysplex, because a DASD-only log stream is single system in scope and you can only have one OPERLOG log stream per sysplex. This means that if you make OPERLOG a DASD-only log stream, only one system can access it and therefore only that system can be an IBM zAware monitored client. See *z/OS MVS Setting Up a Sysplex*, SA22-7625, for information about DASD-only log streams.

Message capture: Even though OPERLOG is a sysplex-wide log stream, System Logger IBM zAware support captures the messages when they are written by each z/OS image. Therefore, every system that will be monitored by IBM zAware sends its own messages to the IBM zAware LPAR.

If you have systems that do not use OPERLOG or System Logger (a GDPS/PPRC Control System, for example), IBM zAware is unable to analyze the message activity on those systems.

Any messages that were suppressed by your MPFLST or Message Flood Automation or similar facilities will be passed to IBM zAware. However any messages that were *deleted* by those facilities will not be sent to OPERLOG, and therefore will not be visible to IBM zAware.

z/OS release requirements

To benefit from the IBM zAware monitoring capability, a system must be running z/OS 1.13 or later with the required enabling PTFs. Systems that are running z/OS 1.11 or 1.12, or systems running z/OS 1.13 without the enabling PTFs, can coexist in the same sysplex with a system that is connected to IBM zAware. However, they cannot use the IBM zAware functions. The changes that must be made to the OPERLOG log stream definition to support IBM zAware will be ignored by those systems.

z/OS systems that will be monitored by IBM zAware must be members of a monoplex or a multisystem sysplex.

System Logger planning

System Logger uses log streams for two aspects of working with IBM zAware:

- ▶ It captures messages that are bound for the OPERLOG log stream and sends them to IBM zAware in real time.
- ▶ When you use the bulk load utility to load historical data into IBM zAware, you define both a model log stream and a temporary log stream. When the utility copies the messages from the input data set to the temporary log stream, System Logger captures those messages and forwards them to IBM zAware.

You will need SAF access the IXGZAWARE_CLIENT resource in the FACILITY class to be able to update the OPERLOG log stream definition to indicate that it is to be sent to IBM zAware.

For the bulk load utility, if you use RACF or similar security products to protect access to log streams, ensure that both System Logger and the person that submits the bulk load job have access to the log streams used by the job.

The LOGR CDS should be formatted at level HBB7705³. However, this was delivered by z/OS 1.2, so there should be no considerations related to sharing the CDS with pre-z/OS 1.13 systems. Remember that the bulk load utility will require a model log stream and a real log stream, so ensure that the LOGR CDS has sufficient unused LSR records.

³ To determine the format level of the LOGR CDS, use the **D XCF,C,TYPE=LOGR** command. If the format level is older than HBB7705, format a new set of LOGR CDSs using the **ITEM NAME(SMDUPLEX) NUMBER(1)** keywords.

Tip: The provided sample bulk load job (member AIZBLK in SYS1.SAMPLIB) defines a model log stream in the first step. To execute with the optimum efficiency, ensure the following points:

- ▶ The LS_SIZE and STG_SIZE parameters are reasonable, based on the amount of data that you will be loading. The default values result in offload and staging data sets that are only 225 tracks. Using these values, in our case we were getting a new offload data set every 3 seconds, which is possibly more frequently than you might want.
- ▶ AUTODELETE(YES) is specified. This specifies that each offload data set should be deleted as soon as it is no longer required. If you do *not* specify this, all the offload data sets will be kept, potentially consuming a large amount of disk space.

3.4 Technical resources

In addition to the hardware and software that is required to run IBM zAware, some setup must be performed to enable it. This section helps you identify the technical skills that are required to complete the installation of IBM zAware.

3.4.1 Hardware Management Console

Because IBM zAware runs as an LPAR, you must create the definitions for the LPAR on the HMC. And because IBM zAware is a new type of LPAR, there is additional information that must be entered on the HMC that does not apply to other types of LPARs.

All of these tasks can be most efficiently completed by someone that is familiar with managing the HMC. It is also useful to have access to whomever is responsible for setting up the network definitions for the IBM zAware LPAR.

Update CPC Reset profile

Before you can create an Image Profile for the IBM zAware LPAR, the CPC Reset profile must be updated to specify the activation order for the newly defined IBM zAware LPAR.

Define Image Profile

The IBM zAware Image Profile is similar to the profiles for most other LPARs. However, in addition to the normal Processor, Memory, and other tabs, there will also be a Firmware tab for any LPAR defined to run in zAware mode. The firmware tab asks for the following information, so ensure that you have this available when defining the IBM zAware Image Profile:

- ▶ The CHPID, VLAN, IP address, and mask (in 2-digit format⁴) for each LAN adapter
- ▶ The default gateway IP address
- ▶ The IP address of the Domain Name Servers, if any, that the IBM zAware LPAR is to use

3.4.2 z/OS MVS

There are a number of tasks that must be completed to prepare for IBM zAware that are likely to fall under the jurisdiction of the z/OS technical support group.

⁴ If your network staff provides you with a 4-byte subnet mask (255.255.240.0, for example), there are a variety of subnet mask converters on the Internet to help you convert that mask to the 2-digit number that is required by IBM zAware (20, for example).

Hardware Configuration Definition

Hardware Configuration Definition (HCD) should be used to define the IBM zAware LPAR and its devices and network adapters. Therefore, whoever is responsible for maintaining the hardware definitions needs to update HCD. If the disks that IBM zAware will use are already defined to other LPARs, creating explicit device candidate lists to limit the LPARs that can access those disks will require additional time.

It might also take an amount of time to roll out the hardware configuration changes to all affected CPCs. Plan to create the hardware definitions and roll out the changes in advance of the start of the IBM zAware installation to avoid unnecessary delays.

IXGCNFxx member in Parmlib

The information that System Logger requires to be able to communicate with IBM zAware is provided in the IXGCNFxx member of Parmlib. This member was initially delivered by z/OS 1.13, so you might not be using it yet. A sample member is provided in the IXGCNFXX member of SYS1.SAMPLIB. You can copy that to Parmlib and use it as a base for the changes you will make for IBM zAware.

The IXGCNFxx member can be activated either by adding the IXGCNF=xx keyword to your IEASYS member and performing an IPL, or dynamically, by issuing the **SET IXGCNF=xx** system command. Note that by default, the IXGCNF00 member (if it exists) is not used. To have an IXGCNFxx member used at the next IPL, you must explicitly specify it in the IEASYS member.

If implementing changes such as this can take a long time in your environment, plan to make this change in advance of starting the IBM zAware installation. The new functions enabled by updating that member will not have any effect until you explicitly start the connection to IBM zAware using a System Logger command.

3.4.3 Network

All communication with IBM zAware, whether to send messages to IBM zAware, to logon and use its GUI, or to use the IBM zAware API, is through TCP/IP connections. Because network configurations vary so widely from one installation to another, it is difficult to provide information that is applicable in all cases. However, the network requirements are not complex and can be easily understood by anyone familiar with managing your IT network.

z/OS port for IBM zAware

System Logger uses TCP/IP to send data to the IBM zAware LPAR. This means that it requires an available port on the z/OS system. The UNIX System Services that System Logger uses to establish a connection to IBM zAware automatically selects an available port. You do not need to define a port number to System Logger, and there is no way to influence which port it will use.

Connectivity

The IBM zAware LPAR can be connected using Hipersockets, OSA adapters, or IEDN. You need to allocate a number of IP addresses for use by the IBM zAware LPAR, so plan them in advance. To avoid single points of failure, configure the IBM zAware LPAR with at least two OSA CHPIDs.

Firewall

Nearly all communication with the IBM zAware LPAR is initiated from outside the LPAR. IBM zAware uses the following ports:

80	For HTTP access
443	For HTTPS access
2001	For access from monitored clients

Any firewall rules that might be required will be based on enabling outside systems to come in through those ports.

If you plan to use an LDAP server, the IBM zAware LPAR will initiate a session with the LDAP server. If there is a firewall between the IBM zAware LPAR and the LDAP server, you must permit the IBM zAware LPAR to use the port that is used by your LDAP server. As part of the installation process, you will be required to provide the IP address or host name and port number to IBM zAware (if you plan to use LDAP). Additionally, if a domain name is used instead of an explicit IP address for the LDAP server, then the IBM zAware LPAR must be able to reach a DNS server for host name-to-IP resolution.

3.4.4 Security

There are two aspects to security when planning for a IBM zAware implementation. One aspect is the SAF resources that System Logger will need access to send the message data to IBM zAware. The other aspect is that you will need to define user IDs that can be used to logon to the IBM zAware LPAR. You will also need to specify the level of authority of IBM zAware users to make changes to the IBM zAware LPAR.

z/OS resources

Because System Logger is the task that is sending all the data to the IBM zAware LPARs, the z/OS access considerations are centered on the ability of IBM zAware to access the resources that are required for this process.

For detailed information about the access requirements for the IBM zAware-related SAF resources, refer to “Define authorization to System Logger resources” in *z/OS MVS Setting Up a Sysplex*, SA22-7625, and *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

z/OS security considerations for the IBM zAware API

The IBM zAware API currently requires communication using Secure Sockets Layer (SSL). If you are going to communicate with the API from a z/OS system, that system must be enabled for Application Transparent - Transport Layer Security (AT-TLS).

Describing the setup of AT-TLS is beyond the scope of this book. However, we provide information explaining the changes we made to enable AT-TLS for IBM zAware in our environment in Appendix B, “Activating TCP/IP AT-TLS” on page 213. Additionally, you can find more information about setting up AT-TLS in the Redbooks publication *IBM z/OS V1R13 Communications Server TCP/IP Implementation: Volume 4 Security and Policy-Based Networking*, SG24-7999.

IBM zAware security

To access the IBM zAware GUI, users must have a IBM zAware user ID and a password. The user IDs can be defined in a local repository, in an LDAP server, or in both places. To keep things simple, we strongly advise that you use just one of these methods rather than having some IDs defined in LDAP, and some in the local repository.

After you have defined the IDs and passwords (either in the local repository or in the LDAP server), you use the IBM zAware GUI to assign a role to each ID. Roles can be either “user” or “admin.” There is also the master user ID that is defined on the Firmware tab in the IBM zAware LPAR’s image profile. That user ID automatically has admin authority. For information about the differences between the user and admin roles, refer to *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

If you use LDAP to define your users, also use a high availability LDAP configuration. If the LDAP server is down, it will not be possible to logon to IBM zAware using any ID that is defined in LDAP. You might also want to define just one or two user IDs in the local repository that can be used if the LDAP server is unavailable. Additionally, we suggest that if you use LDAP as the repository for your IBM zAware IDs, then none of those IDs have Administrator authority on the LDAP server.

For more information about the local repository and the use of the IBM zAware GUI, refer to 4.4.2, “User authentication to use the IBM zAware GUI” on page 131.

Tip: Do *not* use LDAP IDs that have administrator authority on the LDAP server. Doing so can lead to confusion if you accidentally define the same user ID on both the IBM zAware GUI panels and on the LDAP server.

Also, avoid defining the same user ID on both the LDAP server and the IBM zAware GUI. If you define a few user IDs on the GUI as a fallback in case LDAP is unavailable, ensure that you use user IDs that do not exist on the LDAP server.

3.5 Using IBM zAware

The perspective of system message traffic that is provided by IBM zAware is quite different than anything you are familiar with the System z systems management sphere. Fully appreciating the information that is presented by IBM zAware will require time to get familiar with it.

Thought needs to be given to which of your staff will be the primary users of IBM zAware. That is, will it be operators, system management personnel, system programmers, or possibly all three groups? Regardless of who it will be, you need to plan for time for education to allow IBM zAware users to efficiently navigate around the IBM zAware GUI and understand both the information they are presented with and why IBM zAware considers that information to be an anomaly. All of this will take time.

Additionally, someone must be responsible for monitoring and managing IBM zAware, and asking questions such as: is it still working as expected, and does it have sufficient disk space? If there are days when the activity on one system was not representative of a “normal” day, you might want to exclude that day from the next IBM zAware training cycle.

These are just some of the tasks required to effectively manage IBM zAware, so identify who will have that responsibility, and allow time for them to become familiar with their responsibilities. The management of IBM zAware is discussed at length in Chapter 5, “Maintaining and managing IBM zAware” on page 153.

3.6 Monitoring IBM zAware availability

The expected usage of IBM zAware is that you will use it if you detect something unusual happening in your environment. More than likely, you will not have someone watching IBM zAware on a constant basis. This means that there is a risk that IBM zAware has stopped processing the message traffic for one or more systems but no one has noticed, or not noticed until they need to use IBM zAware, at which point it is too late.

For this reason, we advise that you consider using the IBM zAware API to monitor IBM zAware on an ongoing basis, to ensure that it is still working, and that it is still receiving message data from every monitored client. 6.3, “IBM Tivoli NetView for z/OS” on page 196 contains information about sample NetView execs that perform this checking.

The most likely cause of a “problem” is that the monitored clients disconnected for some reason and failed to reconnect successfully. The following list includes several of the events that cause the connection between IBM zAware and monitored clients to be dropped:

- ▶ When you IPL a z/OS system, the connections from that system will be stopped. After the IPL, System Logger will automatically try to reestablish the connections, regardless of whether the connection was active at the time the system was IPLed.
- ▶ When you assign bulk loaded data to a sysplex, the connections to all monitored clients will be terminated by IBM zAware. Normally, the links should be automatically restarted by System Logger. However, problems can be encountered during this process.
- ▶ When you apply firmware maintenance to IBM zAware, IBM zAware will relPL itself and the connections to the monitored clients will be dropped. Again, the links should be automatically restarted by System Logger, but problems can be encountered during this process.

Presumably you will have automation in place on each monitored client that will query the state of the IBM zAware connections, and take corrective action if a connection is down. Nevertheless, it is still advisable to perform some automatic monitoring of IBM zAware itself to ensure that it is still running *and* analyzing the message data.

3.7 Accessing the IBM zAware GUI

We realize that some enterprises lock down their workstations so that no additional software can be installed on them. The only workstation software that is required by IBM zAware is a supported browser (Internet Explorer or Firefox). It is not necessary to install any client software or make any other changes that are likely to not be allowed.



IBM zAware installation

This chapter describes our experiences with installing IBM zAware. The detailed installation information is provided in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623. However, providing a working example can be helpful in guiding you through the installation steps.

This chapter describes all the steps involved, from the beginning of the installation process through the point where historical message data is loaded into IBM zAware.

Specifically, the following topics are discussed:

- ▶ Overview of the installation
- ▶ Customizing the LPAR activation profile
- ▶ Sample network configuration
- ▶ Security considerations
- ▶ Setting up System Logger
- ▶ Connecting to the IBM zAware server
- ▶ Running the bulk load client

4.1 Overview of the IBM zAware installation process

This chapter provides an overview of the steps involved in installing IBM zAware, followed by a description of our experience. To give an idea of the effort involved and the elapsed time, our installation was completed in slightly over half a day. Spending a little more time on advanced planning, particularly in the network area, would have allowed us to complete the installation in less than half a day.

To understand the information that IBM zAware bases its analysis on, it is important to be familiar with z/OS message flow because that determines which messages IBM zAware will see. When a Write To Operator (WTO) message is issued, it goes through Message Processing Facility (MPF) and Message Flood Automation (MFA) processing, and is queued to a message data space, as shown in Figure 4-1. The message is then delivered to MCS consoles, SYSLOG, OPERLOG, and so on.

When a message is sent to OPERLOG, and the OPERLOG log stream is defined with the appropriate attributes, System Logger queues the message to be sent to IBM zAware. Therefore, you need to customize System Logger, the OPERLOG log stream definition, and TCP/IP settings to enable the flow of messages from OPERLOG to IBM zAware.

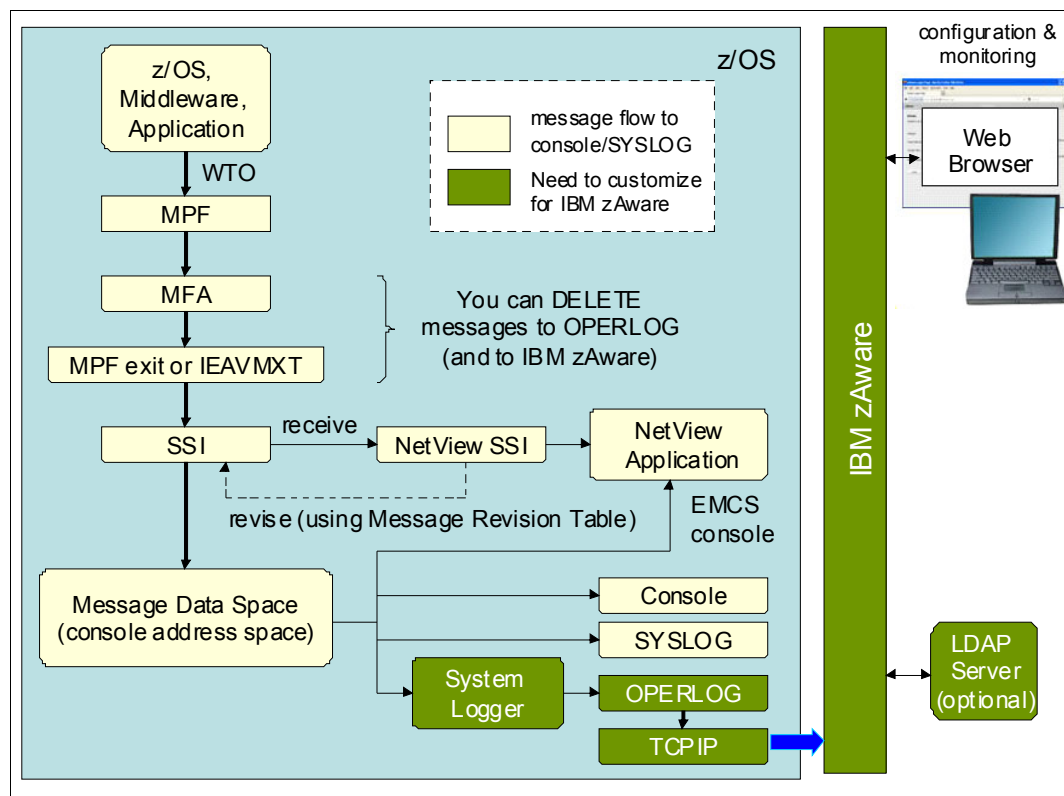


Figure 4-1 Overview of z/OS message flow

When the message data is received in the IBM zAware LPAR, IBM zAware analyzes the message and reports anomalous message behavior. You can view the analysis results on the IBM zAware GUI using your web browser. You need to customize the IBM zAware LPAR setting and the security environment to control access to the IBM zAware server.

Installing IBM zAware involves several steps. Each step is covered in detail in the following sections.

IBM zAware LPAR

Configuring the IBM zAware server involves the following four main steps:

1. Update your hardware configuration.

Use HCD to update your hardware definition to include the IBM zAware LPAR and its storage and network devices.

2. Customize an Activation Profile.

Before you can perform any customization on IBM zAware, you must define the image profile of the IBM zAware LPAR. You need to update the following tabs in the image profile on the HMC:

- General
- Processor
- Storage
- Firmware

To define this as an IBM zAware LPAR, you select the LPAR mode of “zAware” on the “General” tab. Then you specify how many processors and how much memory you assign for IBM zAware, and use the “Firmware” tab to provide the network information that enables IBM zAware to communicate with your browser and the systems that it will monitor.

3. Activate the IBM zAware LPAR.

After you customize an Activation Profile, you can activate the IBM zAware LPAR. It takes a few minutes for the LPAR to initialize.

4. Configure the IBM zAware server.

After the IBM zAware LPAR is activated successfully, you can access the IBM zAware server through your web browser. Before IBM zAware can perform any analysis, you must define storage devices for the IBM zAware server using the administration menu in the IBM zAware GUI.

Network

To deliver the message data from the monitored systems, you need to define network devices for the IBM zAware LPAR in the IOCDs. You also need to customize the TCP/IP profile and the hosts file of monitored systems to support the connection to the IBM zAware server.

Security considerations

This step includes two main sets of actions:

1. RACF definitions for System Logger

You need to set up the authority that the z/OS System Logger requires to send data to the IBM zAware server. You also need to define the SAF accesses that are required for the z/OS Bulk Data Load Utility.

2. User Authentication for the IBM zAware GUI

You define users to access the IBM zAware server through a local file-based repository or an LDAP directory. The “Role Mapping” menu on the IBM zAware GUI is used to distinguish between IBM zAware users and administrators.

Prepare System Logger

To prepare System Logger for use with IBM zAware, you must update the OPERLOG log stream definition in the Logger CDS. You also need to define new parameters in the IXGCNFxx Parmlib member for System Logger to have the information it requires to be able to connect to IBM zAware.

Connect

After you finish the previous steps you can connect monitored systems to the IBM zAware server by issuing System Logger **SETLOGR** commands. You can also check the status using System Logger **DISPLAY** commands and the IBM zAware GUI.

Run the Bulk Data Load Utility

To build a realistic model of normal system behavior, IBM zAware requires message data for a representative period of time. In general, 90 days of message data (from syslog or OPERLOG archives) should be sufficient to let IBM zAware build its model. You use the Bulk Data Load Utility to send the historical message data to the IBM zAware server. Before running the Bulk Data Load Utility, however, you need to establish the TCP/IP connection from the system to the IBM zAware server.

From the viewpoint of the IBM zAware installation, the definitions that must be updated to start the IBM zAware analysis are listed in Table 4-1.

Table 4-1 Definitions for IBM zAware

Component	Definition	Where to define	Reference
IOCDS	IBM zAware LPAR	HCD or IOCP and HMC	4.2, "IBM zAware LPAR" on page 117
	Storage devices		
	Channel paths		4.3, "Network" on page 128
Activation Profile	Image Profile	HMC	4.2, "IBM zAware LPAR" on page 117
IBM zAware server	Configuration	IBM zAware GUI	4.2, "IBM zAware LPAR" on page 117
	Training data	Monitored z/OS system and IBM zAware GUI	4.7, "Bulk Data Load Utility" on page 146
OPERLOG	OPERLOG log stream definition, CONSOLxx and so on	Monitored z/OS system	4.5, "Prepare System Logger" on page 140
System Logger	Logger CDS		
	IXGCNFxx		
	IEASYSxx		
	RACF		4.4, "Security considerations" on page 130
TCP/IP	Profile member		4.3, "Network" on page 128
	Hosts file		
User Security	Authority to access to the IBM zAware GUI	IBM zAware GUI, Security product	4.4, "Security considerations" on page 130

4.2 IBM zAware LPAR

Setting up the hardware configuration definitions for the IBM zAware LPAR includes the following activities:

1. Define the IBM zAware LPAR and devices in your IOCDs.

To start, make sure that the IBM zAware LPAR is defined in the IOCDs. Also, the storage devices that the IBM zAware server will use must be defined, and the IBM zAware LPAR needs to be in the access list of those devices.

As discussed in 3.3.3, “DASD storage” on page 102, we strongly advise that you set up an explicit device candidate list so that the only storage devices that the IBM zAware LPAR has access to are its own devices.

In this example we did not use an explicit device candidate list because ours was merely a test environment. However, in an actual production environment, such a list should definitely be used. Although slightly time-consuming to set up the first time, after the definitions are in place future maintenance should be straightforward.

In our configuration, we use an LPAR name of “A1A” and storage device numbers that range from 9487 to 9489 (Figure 4-2).

```
RESOURCE PARTITION=((CSS(0),(A0A,A),(A0B,B),(A0C,C),(A0D,D),(A*
    0E,E),(A0F,F),(A01,1),(A02,2),(A03,3),(A04,4),(A05,5),(A*
    06,6),(A07,7),(A08,8),(A09,9)),(CSS(1),(A1A,A),(A1B,B),(A*
    A1C,C),...))

CHPID PATH=(CSS(0,1,2,3),44),SHARED,*
PARTITION=((CSS(2),(A21),(=))),*
NOTPART=((CSS(0),(A0B,A0C,A0D,A0E,A0F),(=)),(CSS(1),(A1B*
    ,A1D,A1E,A1F),(=))),SWITCH=61,PCHID=520,TYPE=FC
.....

CNTLUNIT CUNUMBR=9480,*
PATH=((CSS(0),44,45,46,47),(CSS(1),44,45,46,47),.....),*
UNITADD=((00,256)),CUADD=9,UNIT=2107
IODEVICE ADDRESS=(9480,128),UNITADD=00,CUNUMBR=(9480),*
STADET=Y,UNIT=3390
```

Figure 4-2 Example of the IOCP definition

You also need to define the network devices for the IBM zAware LPAR in your IOCDs. Figure 4-3 shows the sample IOCP definition of a QDIO OSA.

```
CHPID PATH=(CSS(0,1,2,3),00),SHARED,*
PARTITION=((CSS(2),(A21),(=))),*
NOTPART=((CSS(0),(A0B,A0C,A0D,A0E,A0F),(=)),(CSS(1),(A1B*
    ,A1D,A1E,A1F),(=))),PCHID=574,TYPE=OSD
CNTLUNIT CUNUMBR=2040,*
PATH=((CSS(0),00),(CSS(1),00),(CSS(2),00),(CSS(3),00)),*
UNIT=OSA
IODEVICE ADDRESS=(2040,015),UNITADD=00,CUNUMBR=(2040),UNIT=OSA
IODEVICE ADDRESS=204F,UNITADD=FE,CUNUMBR=(2040),UNIT=OSAD
```

Figure 4-3 OSA definition for the IBM zAware LPAR in IOCP

Figure 4-4 shows the IBM HiperSockets™ definition statements. If the monitored clients and the IBM zAware LPAR are in the same zEnterprise Ensemble, you can use a channel path identifier of OSX for the IEDN. Make sure the IBM zAware LPAR is included in the access list.

The CHPIDs that are used for the OSA adapter and HiperSockets (00 and F4, in this case) will be specified when you set up the IBM zAware image profile on the HMC. All LPARs that will communicate with the IBM zAware LPAR using HiperSockets should be defined in the access list for that CHPID (F4).

```
CHPID PATH=(CSS(0,1,2,3),F4),SHARED,*
PARTITION=((CSS(2),(A21),(=))),*
NOTPART=((CSS(0),(A0B,A0C,A0D,A0E,A0F),(=)),(CSS(1),(A1B*
,A1D,A1E,A1F),(=))),TYPE=IQD
CNTLUNIT CUNUMBR=7A00,*
PATH=((CSS(0),F4),(CSS(1),F4),(CSS(2),F4),(CSS(3),F4)),*
UNIT=IQD
IODEVICE ADDRESS=(7A00,008),CUNUMBR=(7A00),UNIT=IQD
```

Figure 4-4 HiperSockets definition for the IBM zAware LPAR in IOCP

2. Customize the IBM zAware LPAR Activation profile.

Before you activate the IBM zAware LPAR, you need to customize its Image profile. Select the IBM zAware LPAR and click **Customize/Delete Activation Profiles** task (Figure 4-5) on the HMC.

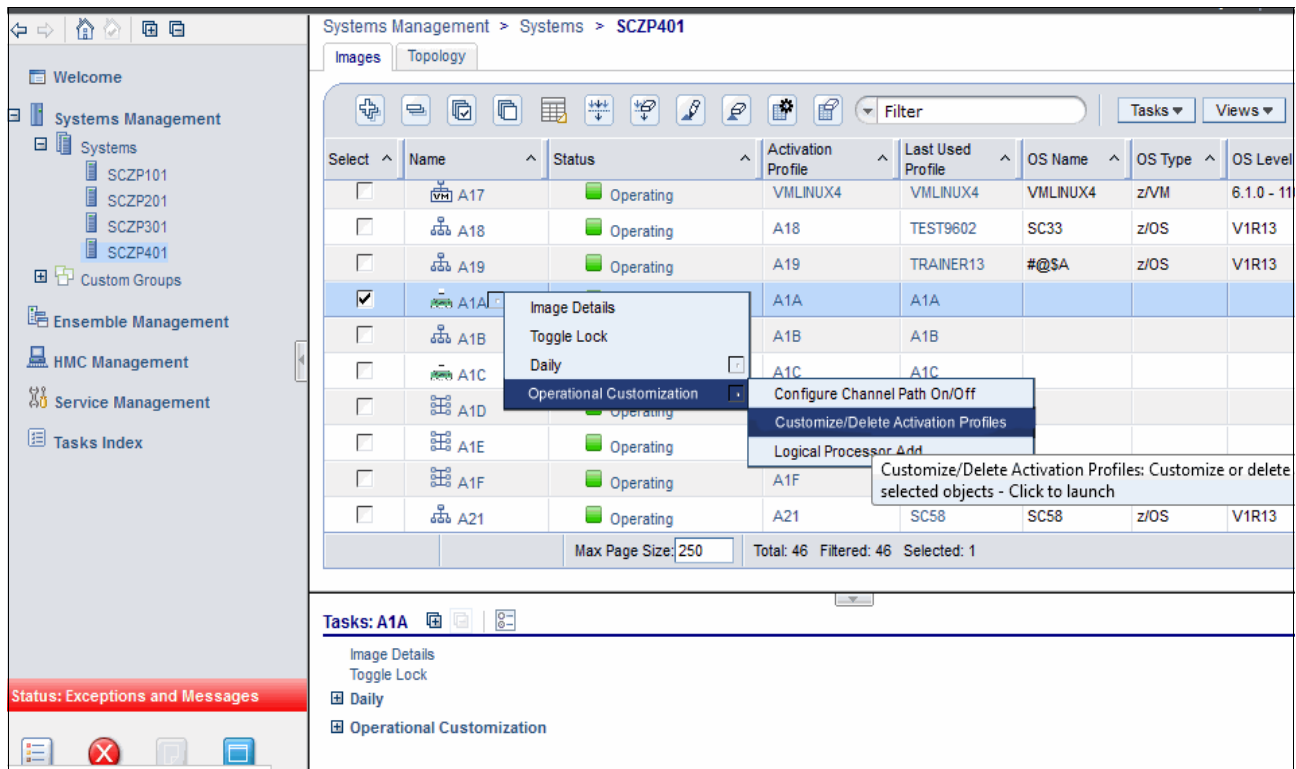


Figure 4-5 Select "Customize/Delete Activation Profiles" task

Select the image profile of the LPAR and click the **Customize profile** button as shown in Figure 4-6.

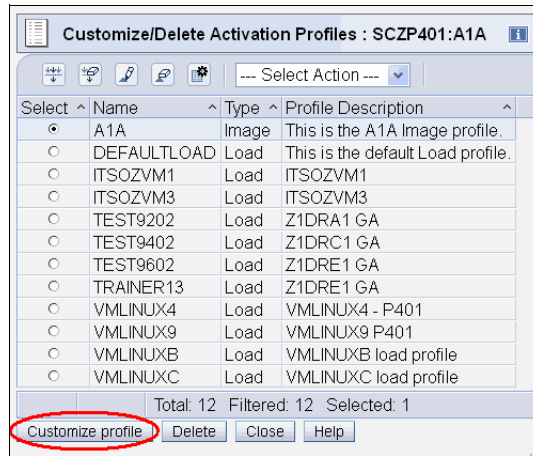


Figure 4-6 Customize profile

On the General tab, there is a new LPAR mode option of “zAware” for a zEC12 LPAR. Select **zAware** mode as shown in Figure 4-7.

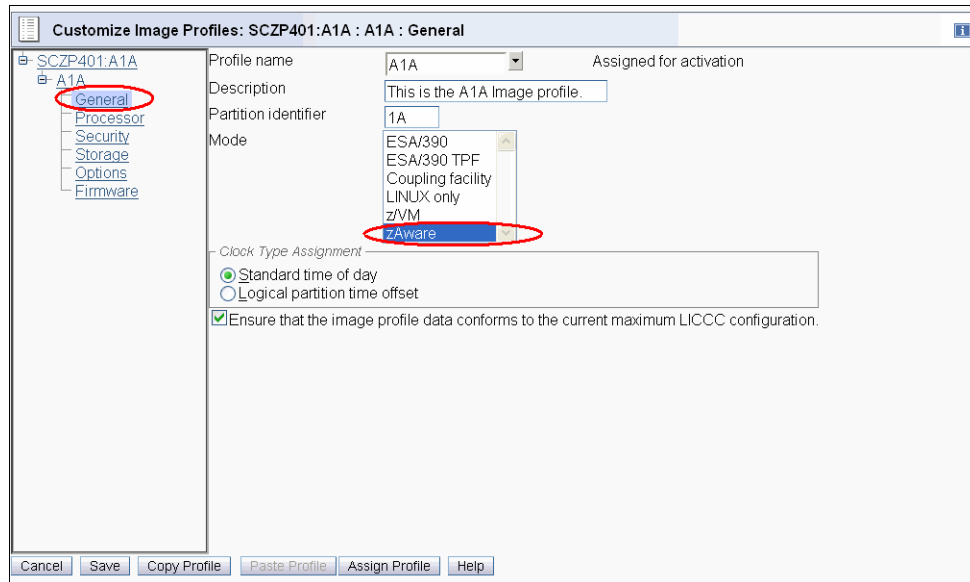


Figure 4-7 Customize General tab

After you select “zAware” mode, a new “Firmware” tab appears. On the Firmware tab, you need to define the network information for the IBM zAware server and a master user ID that has authority to perform any task that is available through the IBM zAware GUI. The definitions for other users who can access to the IBM zAware server will be done later, as described in 4.4, “Security considerations” on page 130.

Figure 4-8 on page 120 shows the values we used to set up our IBM zAware LPAR. In this example, the user ID is FRANK and there are two network adapters. Note that the user ID is not case-sensitive, but the password *is* case-sensitive. Work with your network colleagues to get the information you need to complete the Firmware tab.

Customize Image Profiles: SCZP401:A1A : A1A : Firmware

SCZP401:A1A

A1A

General
Processor
Security
Storage
Options
Firmware

Host name : ZAWARE2
Master user ID : FRANK
Master password :
Confirm master password :
Network Adapters

Select	CHPID	VLAN	IP address	Mask/Prefix
<input checked="" type="radio"/>	f4		10.1.110.3	24
<input type="radio"/>	0		9.12.5.128	20

Default gateway : 9.12.4.1
DNS Servers

Select	IP address
<input type="radio"/>	9.12.6.7

Cancel
Save
Copy Profile
Paste Profile
Assign Profile
Help

Figure 4-8 Customize Firmware tab

In this sample configuration, CHPID 00 is used for the OSA adapter and CHPID F4 is used for the HiperSockets connection. The CHPIDs used here must match those defined in the IOCDs.

The host name and IP address that you enter here might need to be defined in the z/OS hosts files on the monitored systems, as discussed in 4.3, “Network” on page 128.

On the “Processor” tab, you specify the processor assignment for IBM zAware (Figure 4-9). Select general purpose CPs or IFLs, and dedicated or non-dedicated. In our example, we defined our IBM zAware LPAR with one shared IFL.

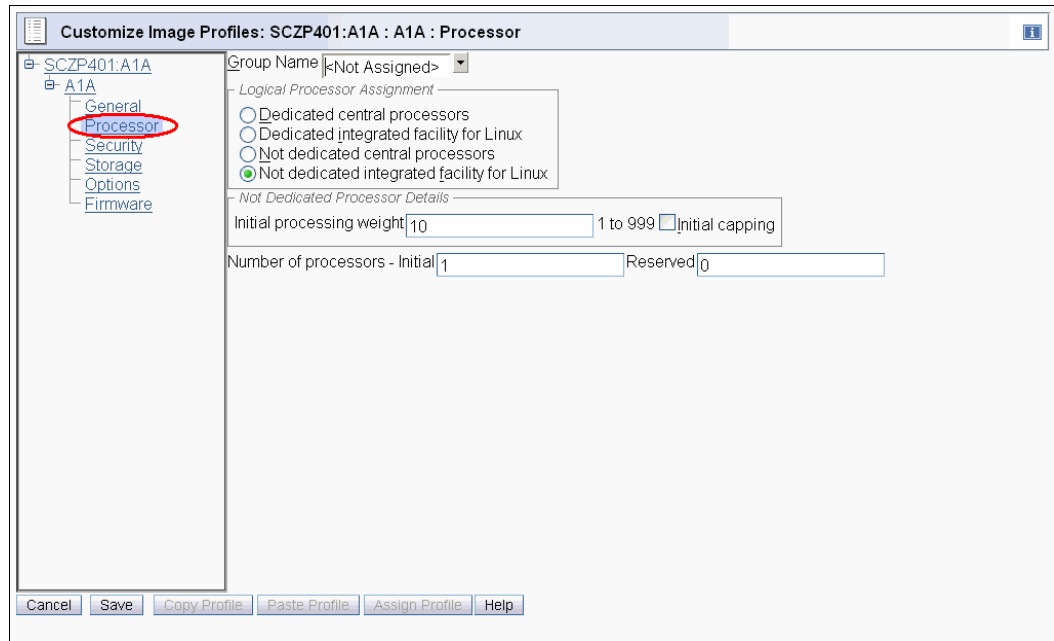


Figure 4-9 Customize Processor tab

On the “Storage” tab, you need to specify the amount of memory that you want to use for the LPAR. If you need to change the amount of memory available to the LPAR at a later time, you will need to deactivate and reactivate the IBM zAware LPAR.

To determine the required hardware resources for IBM zAware, check *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.



Figure 4-10 Customize Storage tab

3. Activate the IBM zAware LPAR.

To start the IBM zAware server, select the IBM zAware LPAR on the HMC and click **Activate** (Figure 4-11).

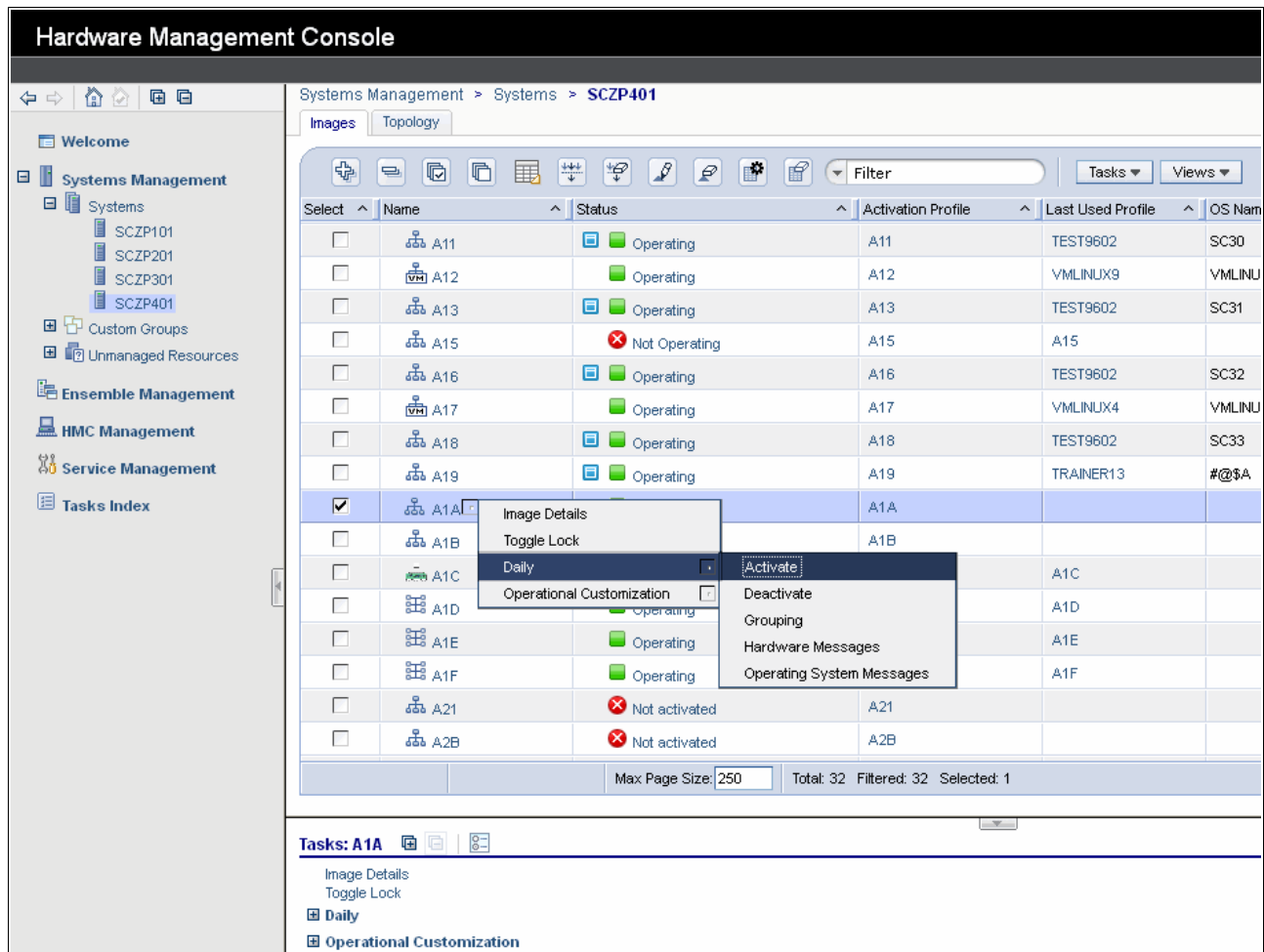


Figure 4-11 Activate the LPAR

The confirmation is displayed as shown in Figure 4-12. Confirm the information and click the **Yes** button.

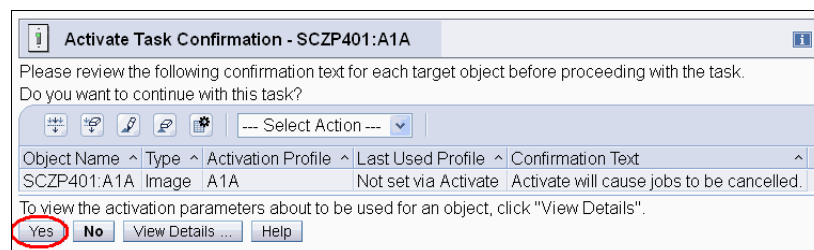


Figure 4-12 Perform Activate Task

Note that it takes a few minutes for the activation to complete (Figure 4-13).

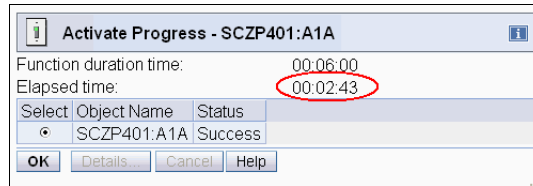


Figure 4-13 Complete LPAR activation

When the initialization completes, press **OK** and you will be returned to the list of LPARs. Confirm that the status of the LPAR is now *Operating*, as shown in Figure 4-14.

Tip: After the IBM zAware LPAR activates, it can take an additional few minutes for the WebSphere server to complete its initialization. Therefore, wait a few minutes before attempting to log on to IBM zAware.

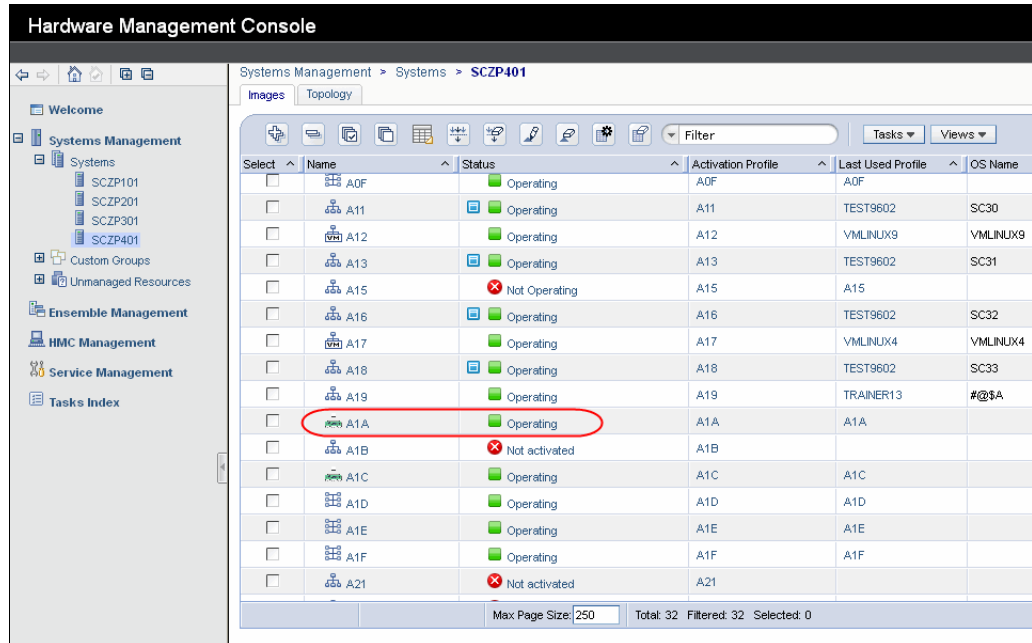


Figure 4-14 Confirm the LPAR status

4. Configure the IBM zAware Server.

After the IBM zAware LPAR has been activated and you have given it a few minutes to complete its initialization, you should be able to log on the IBM zAware GUI. Enter the IP address or host name of the IBM zAware LPAR in your web browser. Use the “ip_address” or “host_name” that were specified on the Firmware tab of the image profile in the previous step:

https://ip_address/zAware/

Or use:

https://host_name/zAware/

After you enter the address, the logon panel is displayed. Enter the Master user ID and password that you specified in the image profile (Figure 4-15); keep in mind that the password is case sensitive.

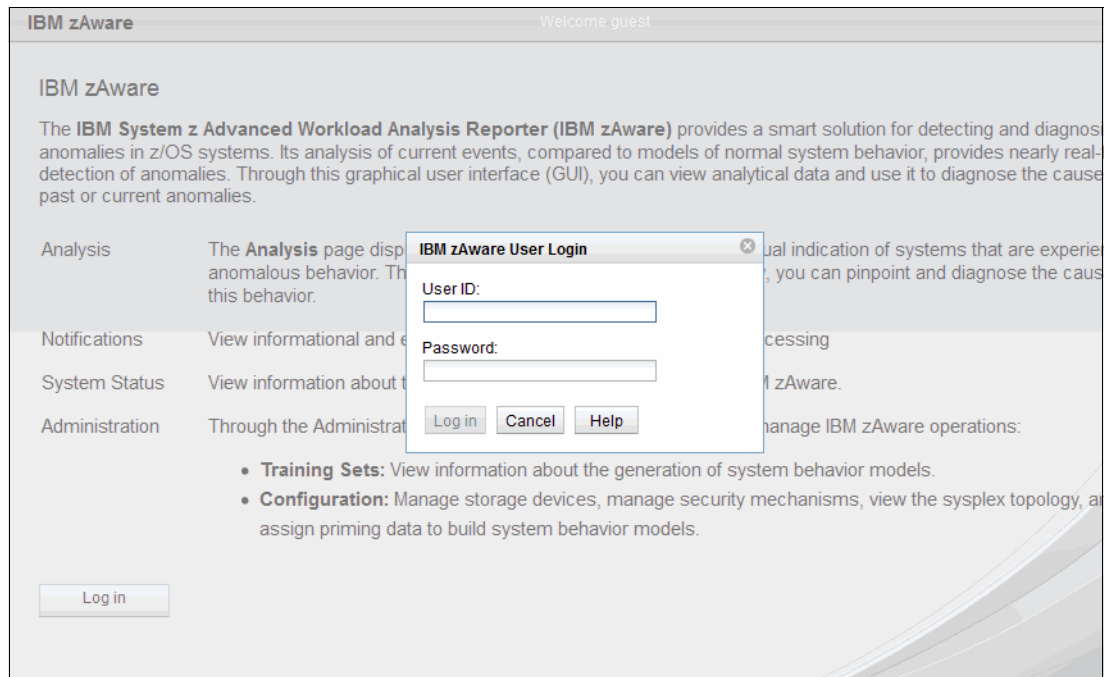


Figure 4-15 IBM zAware initial panel

When you log on the IBM zAware server for the first time, an error message is displayed as shown in Figure 4-16. You can ignore the message and click the **OK** button.

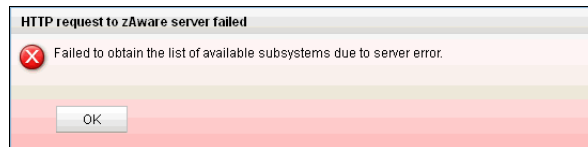


Figure 4-16 Warning information

To receive message data from the monitored systems and build the model, you need to configure the storage devices for the IBM zAware server. Select **Administration** → **Configuration** on the left menu.

zAware Welcome FRANK

Configure Settings

Analytics Data Storage Security Sysplex Topology Priming Data

Total capacity (GB): 0.00 Total storage used (GB): 0.00 Total storage used (%): NaN

Data Storage Devices

Add and Remove Devices Apply Pending Removals

Device	Status	Device Type	Capacity (GB)
9b49	Available	3390/0a	—
9b48	Available	3390/0a	—
9b41	Available	3390/0a	—
9b40	Available	3390/0a	—
9b43	Available	3390/0a	—
9b42	Available	3390/0a	—
9b45	Available	3390/0a	—
9b44	Available	3390/0a	—
9b47	Available	3390/0a	—
9b46	Available	3390/0a	—
9ad2	Available	3390/0a	—

Refresh Last Refresh: Thu Jul 12 2012 14:27:56 GMT-0400 (Eastern Daylight Time)

Figure 4-17 Add and Remove Devices

On the “Data Storage tab”, click the **Add and Remove Devices** tab at the top of the list of devices (see Figure 4-17 on page 125). This will bring you to another panel where you specify the storage devices that IBM zAware is to use.

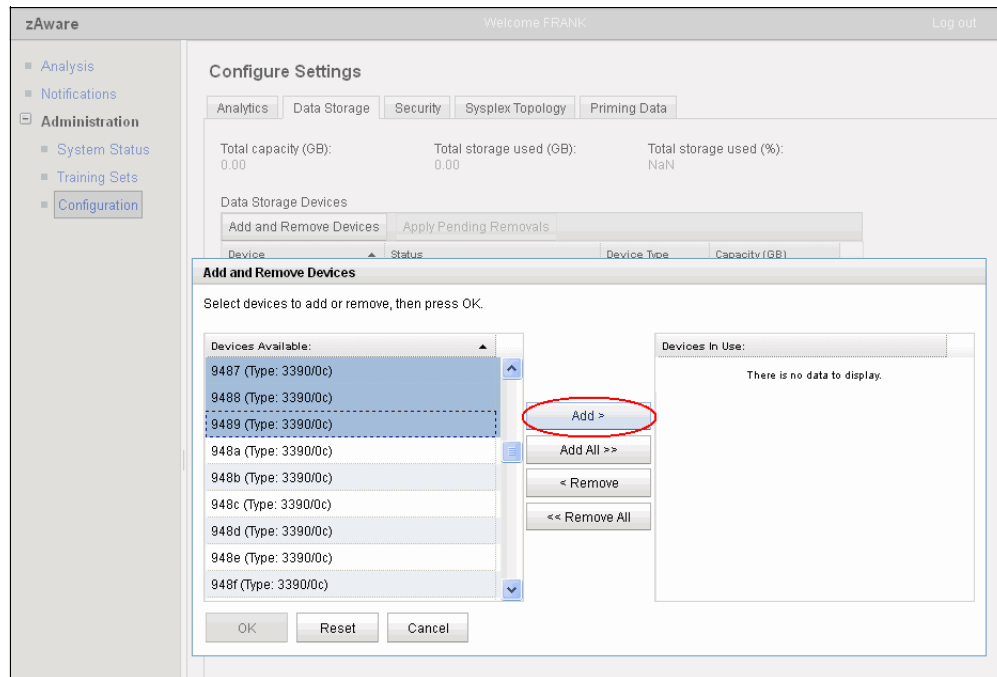


Figure 4-18 Select the devices

On the Add and Remove Devices panel, you can sort the list of devices by clicking the **Devices Available** heading. Select the devices you want to use and click the **Add** button as shown in Figure 4-18. As part of the process of adding a storage device, IBM zAware will always initialize that device.

Ensure that you select the correct devices: IBM zAware does not ask you to reconfirm the devices you select. Therefore, it is important to ensure that you select the correct devices.

To reduce the chance of accidentally initializing the wrong devices, set up the IBM zAware LPAR in HCD so that it can only access its own devices.

Confirm that the correct devices are now listed in the Devices in Use window on the right side. Then click the **OK** button as shown in Figure 4-19.

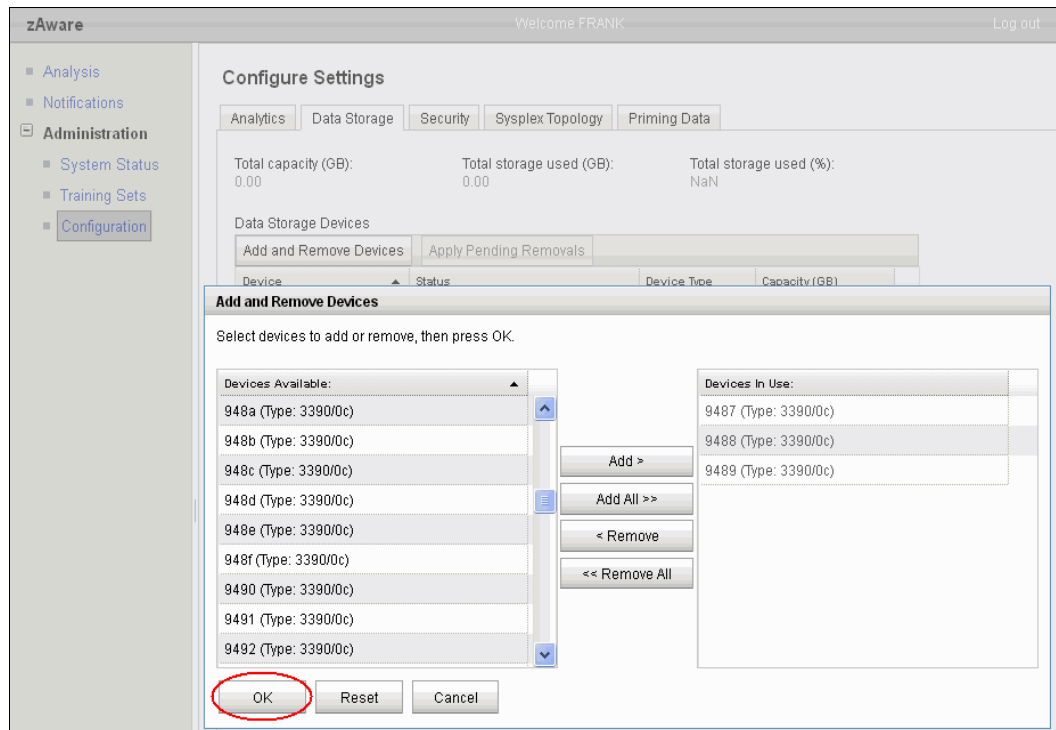


Figure 4-19 Add the selected devices

It might take several minutes to initialize each device, depending on the size of the volume. If the volume was previously used by IBM zAware, the initialization process will detect that and the initialization will take less time. Figure 4-20 shows that the device status changes to Being Added while the device is being initialized.

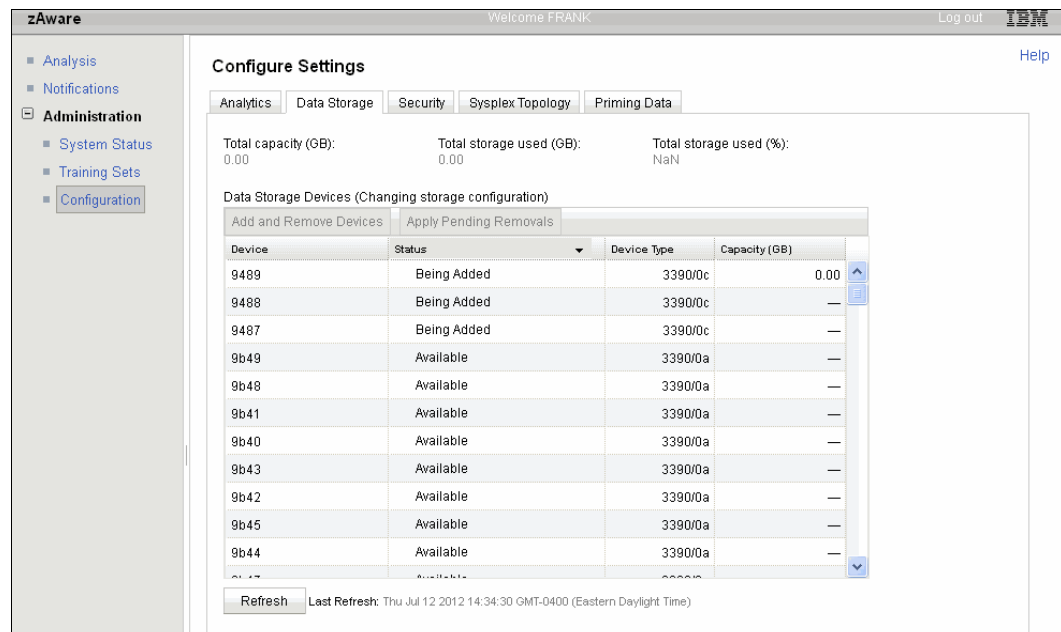


Figure 4-20 Device status sorted by status

When the initialization of the device completes, the status changes to In Use (Figure 4-21).

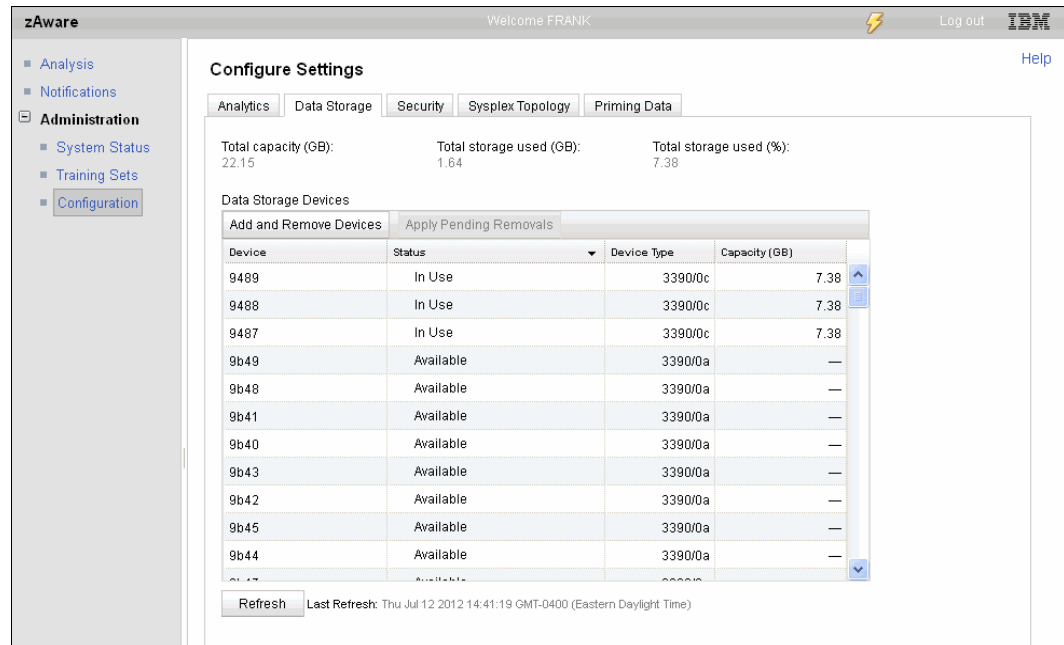


Figure 4-21 The status of added devices

You can customize the “Analytics” and “Security” tabs on the Configuration menu at this point. However, now that you have the Master userID set up and some storage devices have been added to IBM zAware, it is not necessary to make those changes until later.

The “Analytics” tab in the Configuration Settings window is where you control how long IBM zAware is to retain various types of information in its database, and how often it is to train the systems. For now, it is probably best to use the default values provided by IBM. When you have more experience with IBM zAware, you might want to come back later and fine-tune those values. But remember that the values apply to *all* monitored systems; you cannot keep data for one system for 300 days and data from a different system for 600 days, for example.

The customization of the “Security” tab is described in 4.4, “Security considerations” on page 130.

The other tabs, such as “Analysis,” “System Status,” and “Training Sets” will not contain any information until you connect some systems to IBM zAware.

4.3 Network

To connect the monitored clients to the IBM zAware server, you can use HiperSockets, OSA adapters, or IEDN. In this section, we describe how we set up our network connections using HiperSockets and OSAs (see Figure 4-22 on page 129).

We use HiperSockets for the monitored systems that are in the same CPC as the IBM zAware LPAR, and OSAs for the systems in a different CPC.

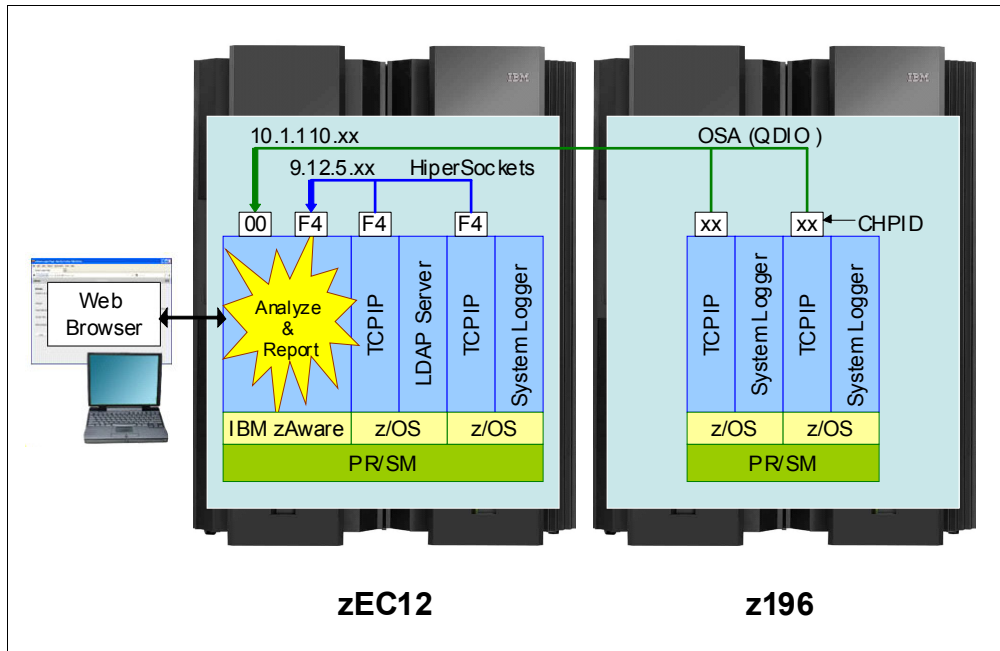


Figure 4-22 Sample network configuration

In this section we show sample TCP/IP settings for one of the monitored systems. The network configuration of the IBM zAware LPAR was defined on the image profile as shown in the previous step.

TCP/IP setting

To connect to the IBM zAware server you might need to customize the TCP/IP settings on the monitored z/OS systems. Figure 4-23 shows a sample profile definition for the OSA connection.

```

DEVICE OSA2400 MPCIPA
LINK   OSA2400LNK IPAQENET      OSA2400
.....
HOME
    9.12.4.xx OSA2400LNK
.....
BEGINROUTES
ROUTE 9.12.4.0 255.255.252.0 = OSA2400LNK MTU 1500
ROUTE DEFAULT 9.12.4.1 OSA2400LNK MTU 1500
ENDROUTES
.....
START OSA2400

```

Figure 4-23 Example TCP/IP profile for the QDIO OSA connection

If you add new OSA devices for the IBM zAware connection, you must also define the TRL major nodes in VTAMLST (see Figure 4-24) and activate them. The OSA devices need to be defined in the IOCDs and the IODF data set.

```
OSA2400  VBUILD TYPE=TRL
OSA2400P TRLE  LNCTL=MPC,
              READ=2400,
              WRITE=2401,
              DATAPATH=(2402,2403,2404,2405),
              PORTNAME=OSA2400,
              MPCLEVEL=QDIO
```

Figure 4-24 Example TRL major node definition for the QDIO OSA connection

Figure 4-25 shows a sample definition for the HiperSockets connection. For this connection, you do not need to define a TRL major node.

```
DEVICE  IUTIQDF4    MPCIPA NOAUTORESTART
LINK    IUTIQDF4LNK IPAQIDIO IUTIQDF4
....
HOME
10.1.110.x IUTIQDF4LNK
....
GATEWAY
10          =          IUTIQDF4LNK  1500          0.255.255.0  0.1.110.0
....
START IUTIQDF4
```

Figure 4-25 Example TCP/IP profile for the HiperSockets connection

To resolve the host name of the IBM zAware server you might need to update the local hosts file. Refer to *z/OS Communications Server IP Configuration Guide*, SC31-8775, to see how the host name is resolved in your environment. Figure 4-26 shows our example `/etc/hosts` file.

```
9.12.4.xx pst1.itso.ibm.com PST1
9.12.4.xx pst2.itso.ibm.com PST2
9.12.4.xx pst3.itso.ibm.com PST3
9.12.9.xxx msys.itso.ibm.com MSYS
10.1.110.x zaware2.itso.ibm.com zaware2
127.0.0.1 localhost
```

Figure 4-26 Example `/etc/hosts` file contents

4.4 Security considerations

This section includes the following topics:

- ▶ RACF definitions on the monitored systems
- ▶ User Authentication to use the IBM zAware GUI

4.4.1 RACF definition on the monitored systems

System Logger requires one or more of the following items:

- ▶ A UNIX System Service segment in its RACF user ID profile
This is required because System Logger will be using TCP/IP services through UNIX System Service.
- ▶ Being defined with a Trusted user ID
If this is done, no further accesses need to be granted to System Logger.
- ▶ Access to the SYSPLEX.OPERLOG log stream
- ▶ Access to the log stream that will be used by the bulk load utility
That log stream name is specified in the bulk data load job.

Additionally, the following people must have update access to the IXGZAWARE_CLIENT resource in the FACILITY class (this is a new resource, added as part of the IBM zAware support):

- ▶ The person who will run the job to modify the OPERLOG log stream definition to add the ZAI and ZAIDAT attributes.
- ▶ The person who will define the model log stream that will be used by the Bulk Data Load job.
- ▶ The person who will run the bulk data load job.
 - That person will also require authority to define new log streams because the bulk load job creates a model log stream.
 - The person will also require read access to the syslog or OPERLOG archive data set containing the messages that are to be sent to IBM zAware.

For more information about the RACF considerations for System Logger resources, refer to “Define authorization to System Logger resources” in *z/OS MVS Setting Up a Sysplex*, SA22-7625.

4.4.2 User authentication to use the IBM zAware GUI

You can define the user authentication for the IBM zAware GUI by one or both of the following methods:

- ▶ Local file-based repository
- ▶ LDAP server directory

Select one strategy and define nearly all your user IDs in that repository; see “IBM zAware security” on page 109 for guidance about defining IBM zAware user IDs. If you use LDAP server, see 4.4.3, “LDAP setup” on page 134.

To manage your IBM zAware user IDs using the local file-based repository, logon to the WebSphere Integrated Solutions Console using the following address:

https://ip_address/ibm/console

Enter the IBM zAware master user ID and password that you defined in the image profile for that LPAR, as shown in Figure 4-27 on page 132. User IDs and their roles are defined separately. The user ID is defined either in the local file-based repository or in LDAP. The role

of each user ID (user or admin) is then defined using the IBM zAware GUI, as described in “Role mapping” on page 137.

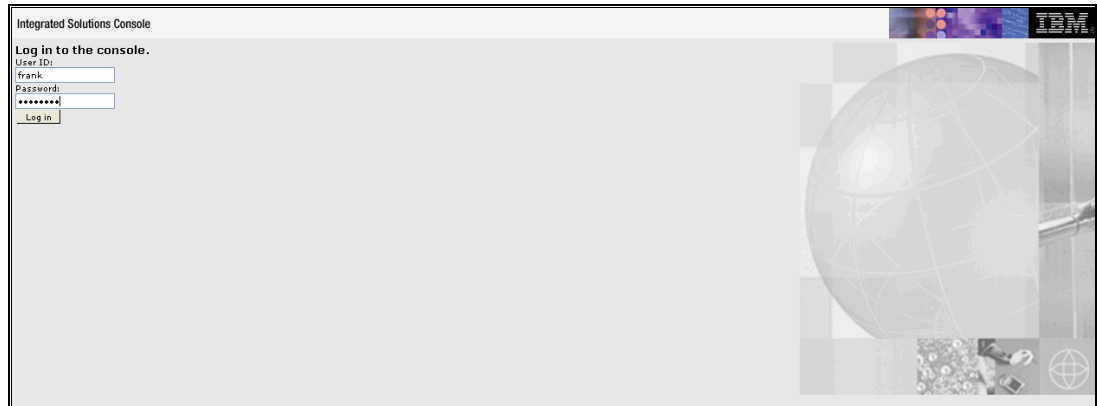


Figure 4-27 Logon panel of the Integrated Solutions Console

The Integrated Solutions Console is used to define user IDs in the local file-based repository. For example, to add a user, select **Users and Groups** → **Manage Users** on the left menu and click the **Create** button as shown in Figure 4-28.

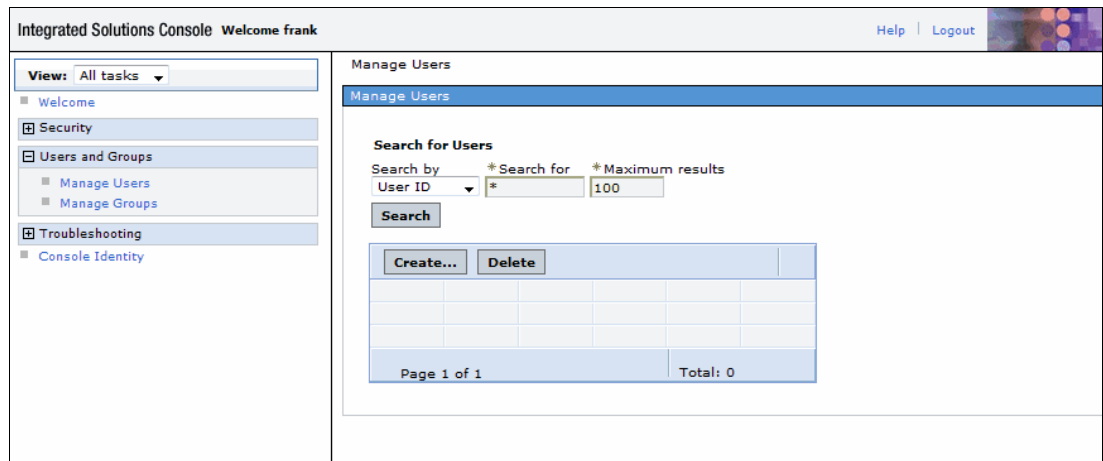


Figure 4-28 Manage Users window

On the Manage users window, enter the user ID, the person’s first name, last name, email address and a password. Click the **Create** button to add the user, as shown in Figure 4-29.

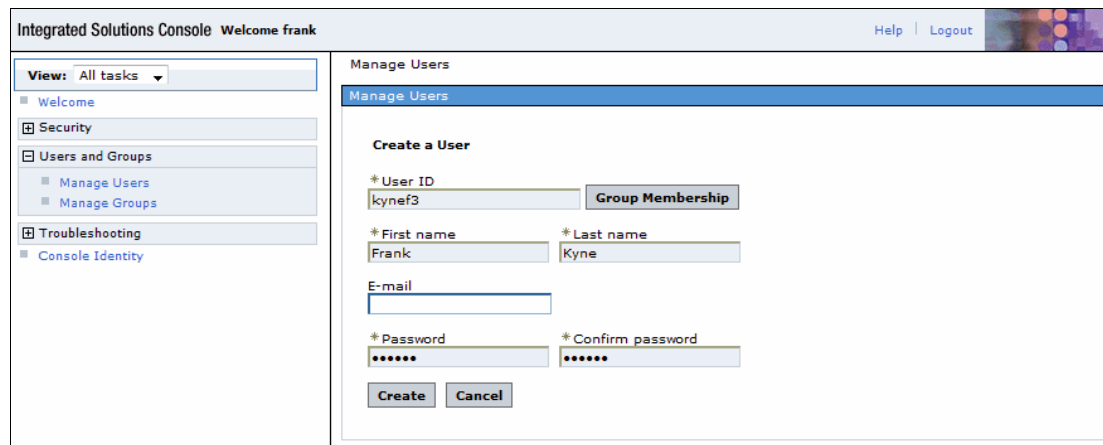


Figure 4-29 Creating a user ID

You can confirm that the user was added correctly using the Manage Users menu as shown in Figure 4-30.

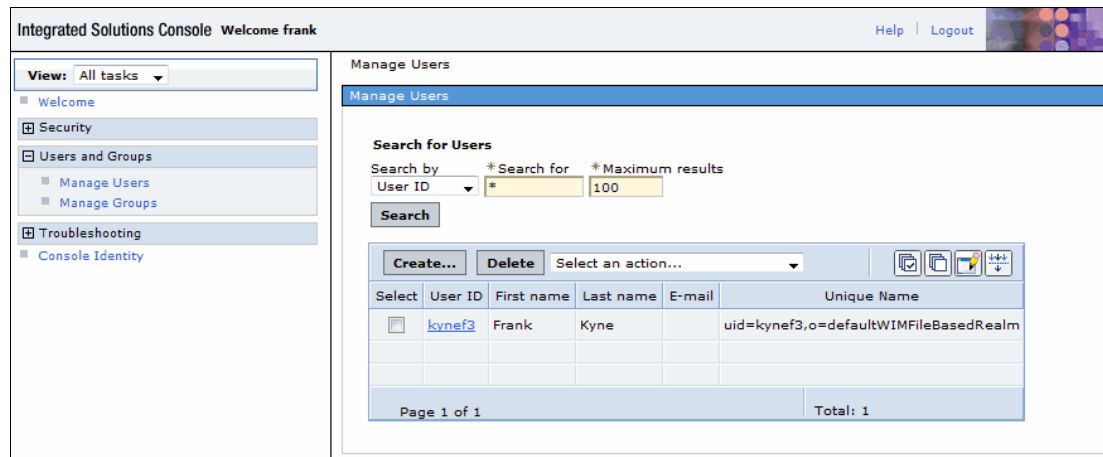


Figure 4-30 Confirm an added user

To display the list of users, click the **Search** button on the Manage Users menu. Notice that the resulting list of user IDs includes users defined on the LDAP server (see Figure 4-31 on page 134).

You can determine which user IDs are defined in the local file-based repository, and which are defined in the LDAP server, by looking at the “Unique Name” associated with each user ID.

- User IDs that are defined in the file-based repository appear as shown here:

uid=frank,o=defaultWIMFileBasedRealm
- User IDs that are defined in LDAP appear as shown here:

uid=angela,ou=itso,o=ibm

To avoid confusion, it is best to manage all your user IDs in one repository or the other. Defining user IDs in both places increases the complexity of managing the user IDs.

Restriction: Although WebSphere supports multiple repositories for user ID definitions, it explicitly does *not* support defining the same user ID in more than one repository.

Doing so will result in unpredictable behavior, including the inability to use the user ID that is defined in both repositories.

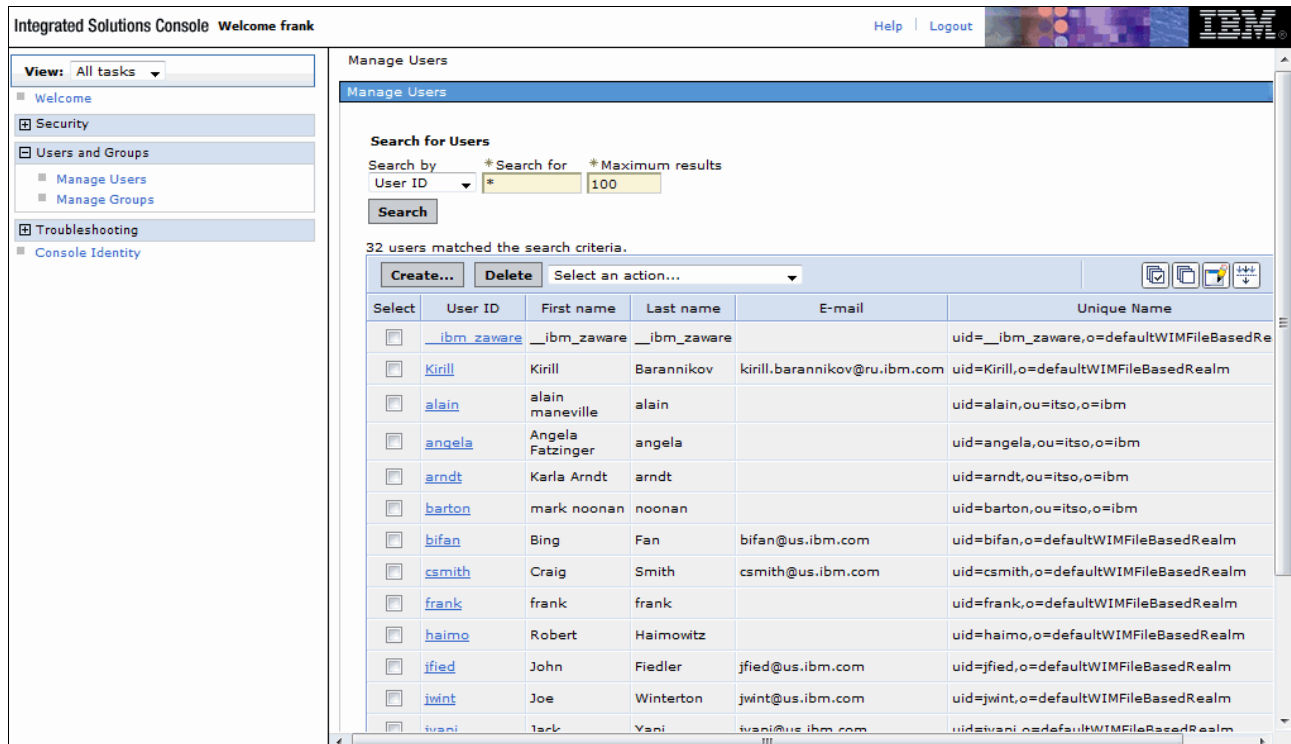


Figure 4-31 Displaying the list of users on the ISC console

IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide, SC27-2623 describes the local file-based security in more detail.

4.4.3 LDAP setup

LDAP data uses a tree structure of user information, as shown in Figure 4-32 on page 135. When IBM zAware is configured to use LDAP, the IBM zAware administrator can fetch user entries from the LDAP server and assign the IBM zAware security roles to the fetched users.

After a user is assigned a security role, that user can then authenticate to the IBM zAware GUI using their LDAP entry and password.

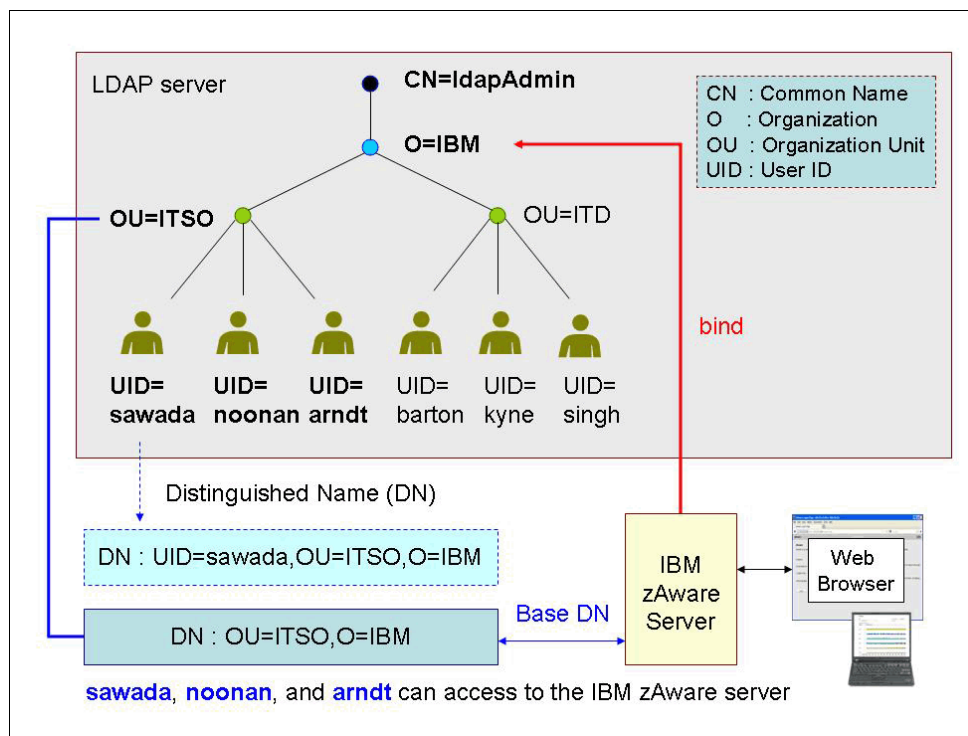


Figure 4-32 LDAP directory information tree

To fetch LDAP user entries, IBM zAware requires a valid bind distinguished name and bind password. The user entries that IBM zAware will fetch are determined by the Base distinguished name, login attribute and User object class fields that are specified in the LDAP Settings panel in the IBM zAware GUI.

After an LDAP user has been assigned a role, that user can then use its LDAP entry to authenticate to the IBM zAware GUI. IBM zAware will issue a bind to the LDAP server using the data entered by the user (ID and password) in the IBM zAware GUI login prompt. If the bind is successful, IBM zAware allows the user access.

The user ID that is entered in the IBM zAware User Login panel is determined by the Login attribute field of the LDAP Setting panel. For example, if the Login attribute field is set to UID and the user has an LDAP entry with the distinguished name of **FRANK,ou=ITSO,O=IBM**, then the user must use **FRANK** as the login name on the IBM zAware login prompt.

Figure 4-33 on page 136 and Figure 4-34 on page 137 show an example of the LDAP Settings window in the IBM zAware GUI.

You specify the host name and port number of an LDAP server to which IBM zAware will communicate with on the LDAP Settings tab within the Security tab on the IBM zAware GUI Configuration Options window. The “Bind distinguished name” and “Bind password” is the LDAP entry that IBM zAware will use to bind to LDAP for searches and retrieval of user entries. If you do not specify this information, the server binds anonymously. You also specify “Base distinguished name” to specify the portion of the LDAP directory tree for which IBM zAware will search for entries, the “Login attribute” which specifies what LDAP attribute will

be used as the login attribute on the IBM zAware login prompt, and the “User object classes” to specify the type of LDAP objects that will be searched and retrieved by IBM zAware.

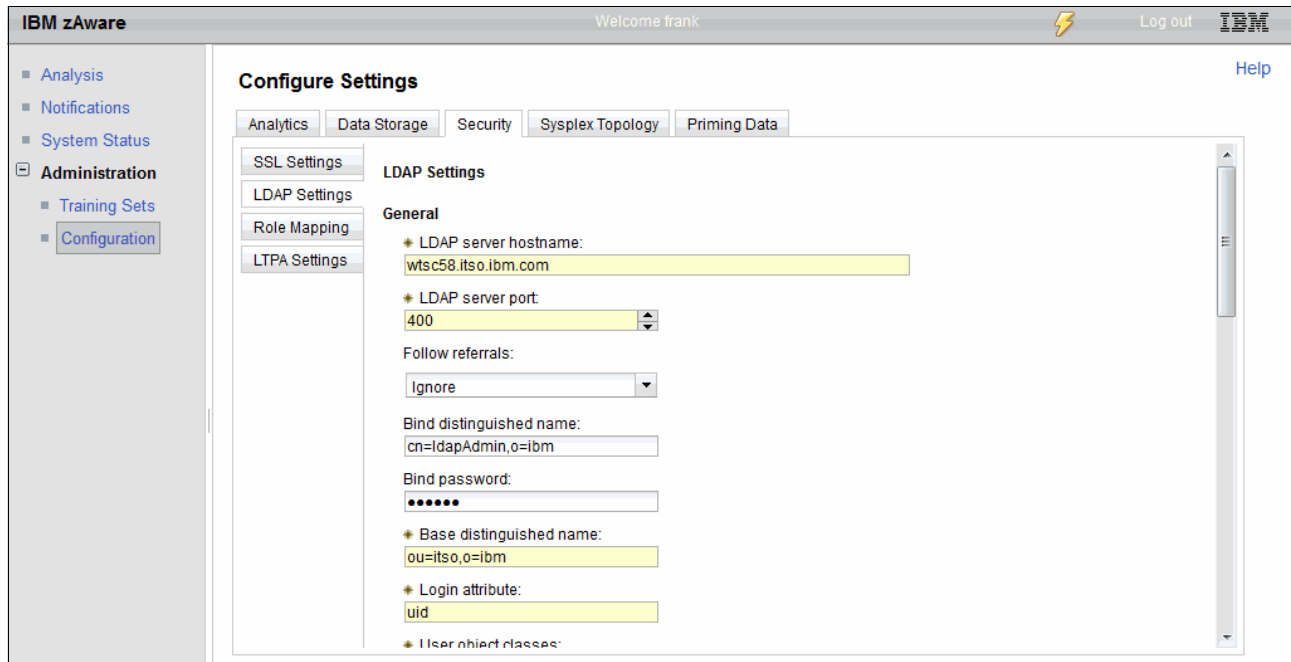


Figure 4-33 LDAP general information settings

If you are using LDAP groups, IBM zAware can also retrieve group objects from LDAP. Typically, LDAP groups contain a list of user distinguished names. For the group setting you are required to input “Group object class,” which specifies the type of group entry to retrieve, and “Group member attribute” information, which specifies the attribute within the group that represents the user entry.

If you manage users by groups, you also specify the “Group search bases,” which represent the portion of the directory tree to be searched, as shown in Figure 4-34.

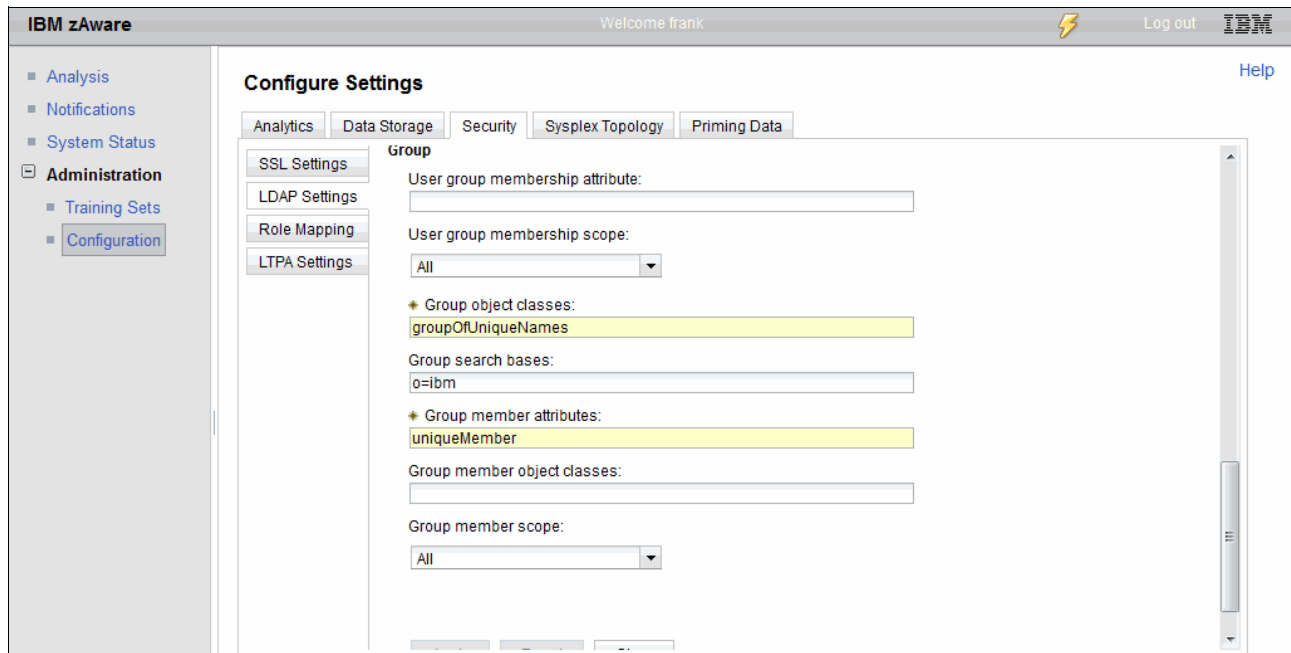


Figure 4-34 LDAP group information settings

Updates to the LDAP Settings panel require a restart of the IBM zAware security server. After you enter the LDAP information and click the **Apply** button, the information panel is displayed as shown in Figure 4-35. Clicking **OK** and restarts the relevant IBM zAware process. Note that restarting the security server does *not* cause the connections to the monitored clients to be stopped.

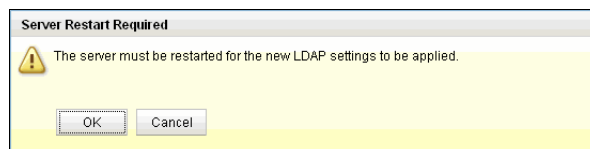


Figure 4-35 Server Restart Required window

The IBM zAware GUI Security tab is also where you assign (map) roles to defined users.

Role mapping

Users who access the IBM zAware GUI must be mapped to one of two roles. One is “administrator” and the other is “user”. An administrator user ID can perform all IBM zAware tasks, but a User user ID cannot use, or see, most of the items on the Administration menu.

To display the roles that are assigned to the defined user IDs, or to assign a role to a user ID, you must log on to the IBM zAware GUI using a user ID with administrator authority. The user ID that you defined in the LPAR’s image profile (known as the master user ID) is automatically defined as an administrator.

Select **Administration** → **Configuration** in the left-side menu and then click **Role Mapping** on the Security tab, as shown in Figure 4-36.

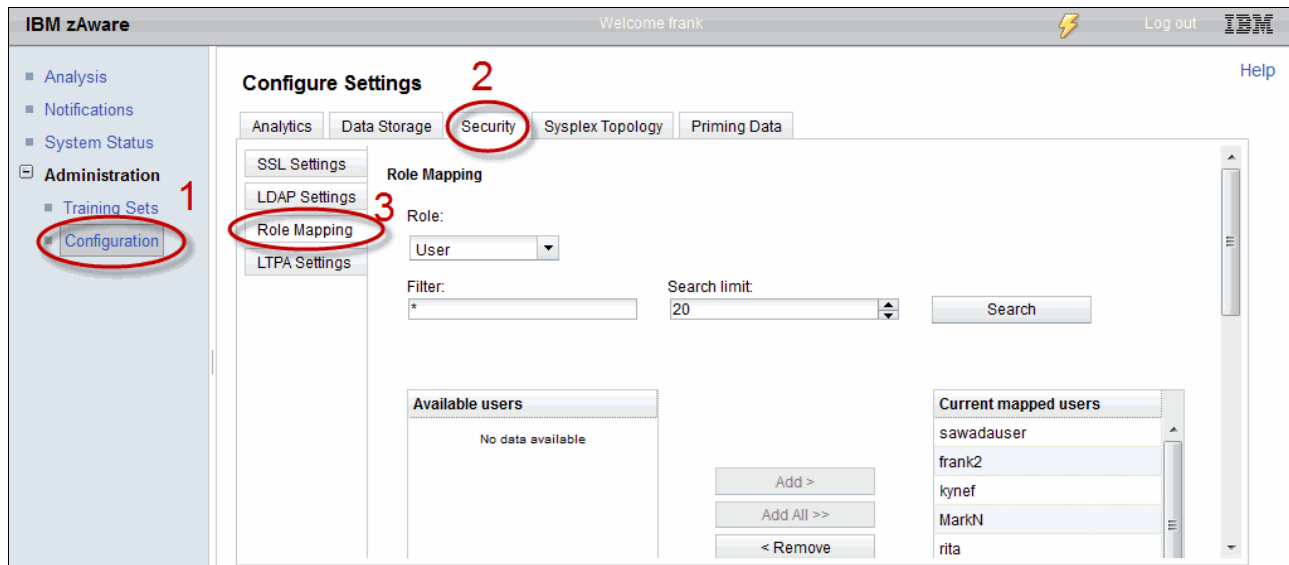


Figure 4-36 Role mapping menu

You can display the list of each type of user ID by clicking the **Role** drop-down and selecting either User or Administrator and then pressing the **Search** button. The resulting display shows a list of all defined users in the Available users box and the list of User or Administrator user IDs (depending on which option you selected) in the Current Mapped Users box (Figure 4-37).

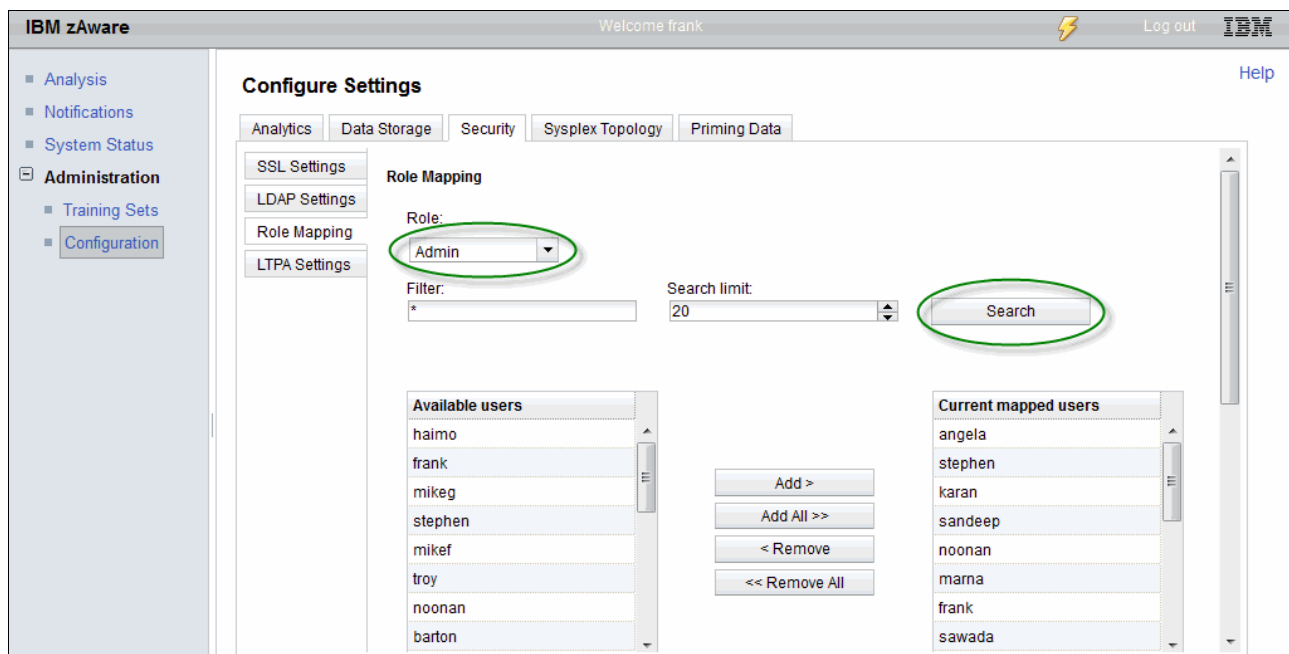


Figure 4-37 Displaying the administrator user IDs

When you add a user ID and want to assign it a role, select the role that you want to assign to that user ID and click the **Search** button. Select the user ID to be added in the Available users box and click the **Add** button, as shown in Figure 4-38.

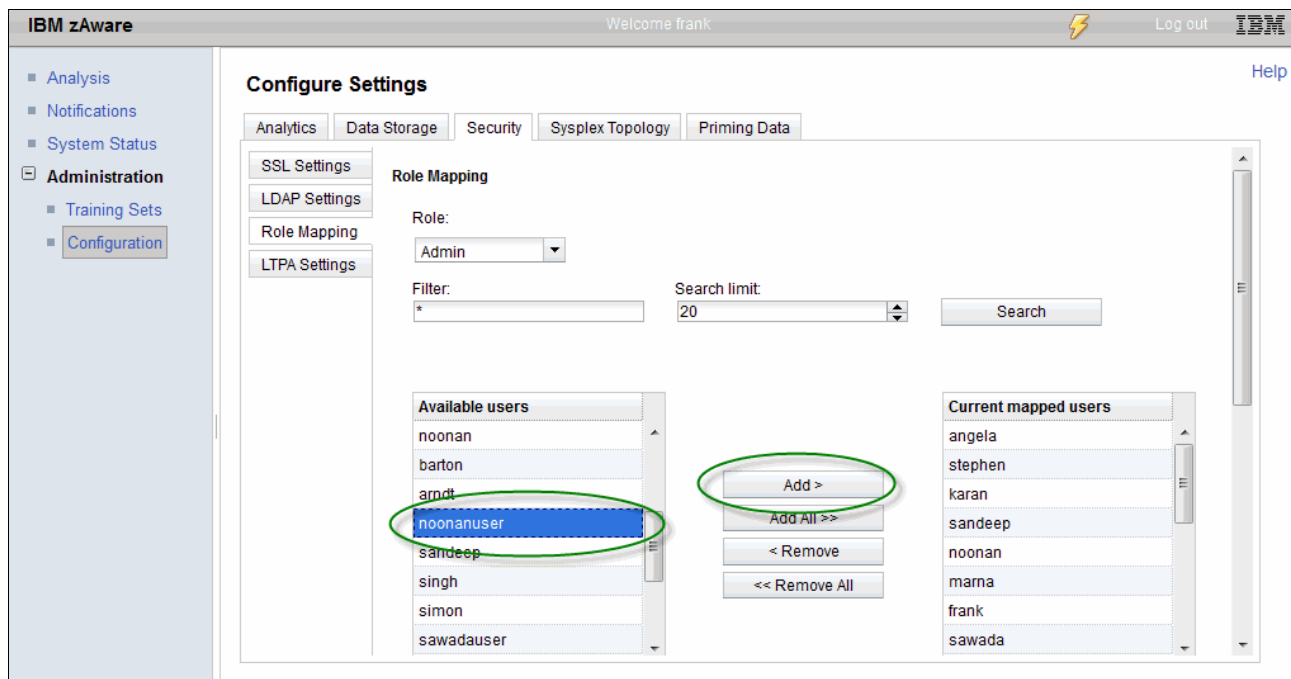


Figure 4-38 Assigning a role to a user ID

When you have finished selecting the user IDs that you want to change, click the **Apply** button to finalize the change for the user ID roles, as shown in Figure 4-39.

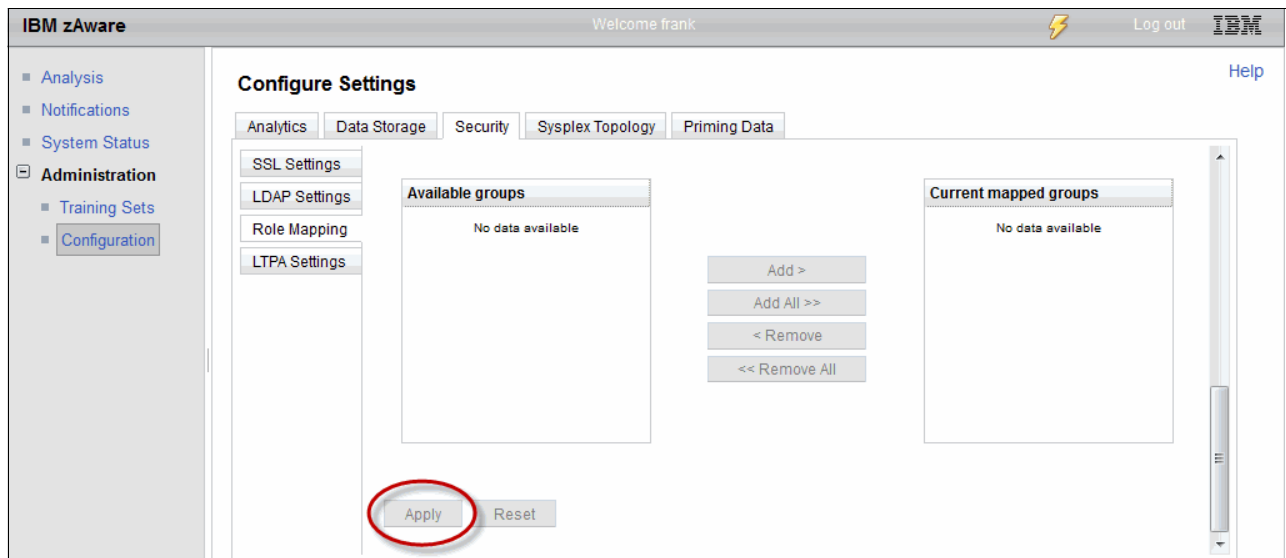


Figure 4-39 Apply the change of user role information

The IBM zAware security server must be restarted to apply the user ID role changes. After clicking the **Apply** button, a Server Restart Required window is displayed, as shown in

Figure 4-40. When you click **OK**, the security server will restart automatically; this will take about one minute.

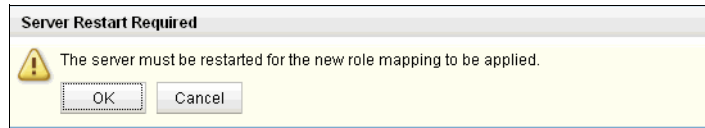


Figure 4-40 *Server Restart Required* window

4.5 Prepare System Logger

This topic describes the steps required to customize System Logger to send message data to the IBM zAware server.

Prepare OPERLOG

If you do not currently use OPERLOG, you *must* enable it on all systems that you want to monitor with IBM zAware. We do not cover setting up OPERLOG in this book. However, if you are not using OPERLOG, you can find information about setting it up in the IBM Redbooks document *S/390 Parallel Sysplex: Resource Sharing*, SG24-5666, and in *z/OS MVS Setting Up a Sysplex*, SA22-7625.

Update the OPERLOG log stream

After you set up OPERLOG, you need to update the OPERLOG log stream definition in the LOGR CDS to indicate that the information in that log stream should be sent to IBM zAware. Figure 4-41 shows a sample set of JCL to update the log stream definition.

```
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSIN DD *
DATA TYPE(LOGR) REPORT(YES)
UPDATE LOGSTREAM NAME(SYSPLEX.OPERLOG)
ZAI(YES)
ZAIDATA('OPERLOG')
```

Figure 4-41 *Sample JCL to update LOGR CDS*

Update IXGCNFxx parmlib member

You must also customize the System Logger Parmlib member (IXGCNFxx) to include the information required to connect to the IBM zAware server. Sample statements are shown in Figure 4-42.

```
/* IBM zAware Connection */
ZAI
SERVER(9.12.5.xxx)
PORT(2001)
LOGBUFMAX(2)
LOGBUFWARN(90)
LOGBUFFULL(MSG)
```

Figure 4-42 *Sample IXGCNFxx member*

Tip: Be sure to include the ZAI keyword on the line before the SERVER keyword.

To ensure that System Logger always comes up with the information it needs to connect to IBM zAware, make sure that your IEASYSxx Parmlib member points to the correct IXGCNFxx member, as shown in Figure 4-43.

If you do not specify IXGCNF=xx in the IEASYSxx member, the system will not default to IXGCNF00. You must explicitly state which IXGCNF member you want System Logger to use.

```
IOS=00,  
IXGCNF=xx,  
LPA=(&LPALST.),
```

Figure 4-43 Example of IEASYSxx member

You can also update System Logger dynamically to use this information using the following command:

```
SET IXGCNF=xx
```

You can use the same command to activate a change if you update any of the keywords in the IXGCNFxx member, or if you create a new member and want to switch to that one.

To verify that System Logger has the information it requires to connect to IBM zAware, issue the following command:

```
DISPLAY LOGGER,IXGCNF
```

The output from this command is shown in Figure 4-44.

```
D LOGGER,IXGCNF  
IXG607I 15.11.06  LOGGER DISPLAY 899  
LOGGER PARAMETER OPTIONS  
KEYWORD          SOURCE    VALUE  
-----  
....  
ZAI  
SERVER    IPV4  SET (xx) 9.12.5.xxx  
PORT      SET (xx) 2001  
LOGBUFMAX SET (xx) 02  
LOGBUFWARN SET (xx) 90  
LOGBUFFULL SET (xx) MSG
```

Figure 4-44 Response to DISPLAY LOGGER,IXGCNF command

To verify that the OPERLOG log stream definition has the correct attributes to be used with IBM zAware, issue the following command:

```
DISPLAY LOGGER,STATUS,ZAI
```

The output from this command is shown in Figure 4-45.

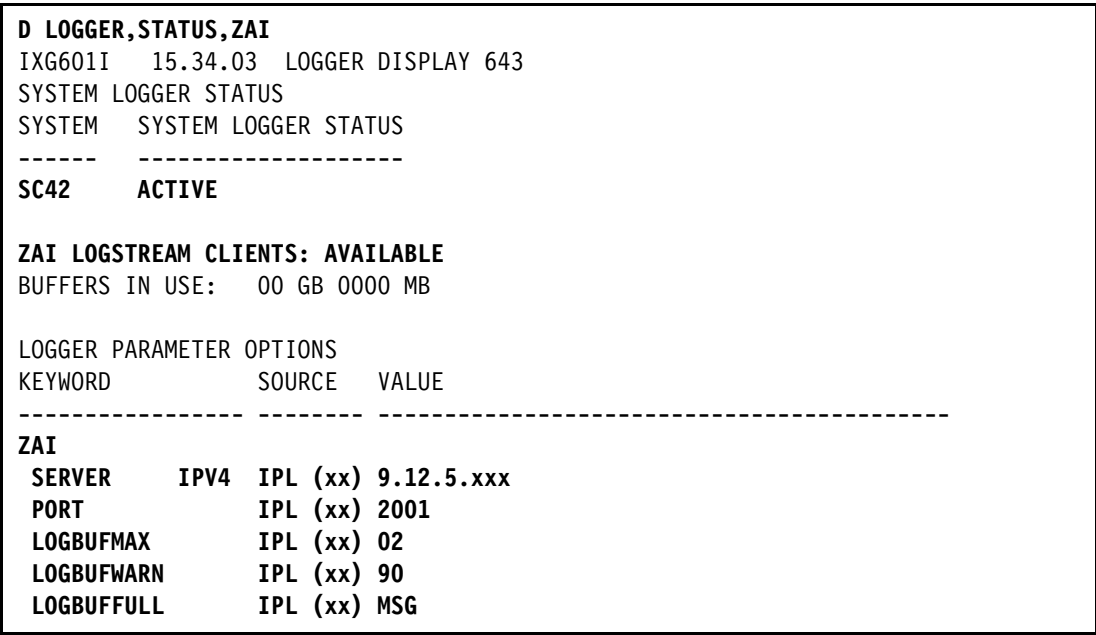


Figure 4-45 Syslog output for DISPLAY LOGGER,STATUS,ZAI command

4.6 Connecting to IBM zAware

To connect to the IBM zAware server, issue the following command on each monitored system:

```
SETLOGR FORCE,ZAICONNECT,LSNAME=SYSPLEX.OPERLOG
```

The response to this command is shown in Figure 4-46.

```
SETLOGR FORCE,ZAICONNECT,LSNAME=SYSPLEX.OPERLOG
IXG651I SETLOGR FORCE ZAICONNECT COMMAND ACCEPTED 013
FOR LOGSTREAM=SYSPLEX.OPERLOG
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 014
FOR LOGSTREAM SYSPLEX.OPERLOG
STATUS: ATTEMPTING SOCKET CREATE
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 015
FOR LOGSTREAM SYSPLEX.OPERLOG
STATUS: SOCKET CREATE SUCCESSFUL
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 016
FOR LOGSTREAM SYSPLEX.OPERLOG
STATUS: ATTEMPTING SOCKET CONNECT
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 017
FOR LOGSTREAM SYSPLEX.OPERLOG
STATUS: SOCKET CONNECT SUCCESSFUL
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 018
FOR LOGSTREAM SYSPLEX.OPERLOG
STATUS: INITIATING SOCKET VALIDATION
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 019
FOR LOGSTREAM SYSPLEX.OPERLOG
STATUS: SOCKET VALIDATION SUCCESSFUL
IXG380I ZAI LOGSTREAM CLIENT ESTABLISHED 020
FOR LOGSTREAM SYSPLEX.OPERLOG
```

Figure 4-46 Response to SETLOGR FORCE,ZAICONNECT command

Note that if you issue the **SETLOGR FORCE,ZAICONNECT** command and there is already an active connection to the IBM zAware LPAR, the connection will be restarted. Any messages that are queued, waiting to be sent to the IBM zAware LPAR, will be retained. Transmission will continue when the connection restart completes.

You can check the connection status with the following command:

```
DISPLAY LOGGER,STATUS,ZAI,VERIFY
```

The response to this command is shown in Figure 4-47 on page 144.

Command consideration: The **DISPLAY LOGGER,STATUS,ZAI,VERIFY** command establishes a connection with the IBM zAware server and changes the Instrumentation Data Type status (shown on the IBM zAware GUI) to null. If the connection to the IBM zAware LPAR is already active, using this command will cause that connection to be restarted.

The **DISPLAY LOGGER,STATUS,ZAI,VERIFY** command was designed to provide a test mechanism to ensure the network path to IBM zAware was available. Avoid using the **VERIFY** parameter if there is an active connection to IBM zAware.

D LOGGER,STATUS,ZAI,VERIFY

```
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 181
FOR DISPLAY ZAI,VERIFY
STATUS:  ATTEMPTING SOCKET CREATE
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 182
FOR DISPLAY ZAI,VERIFY
STATUS:  SOCKET CREATE SUCCESSFUL
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 183
FOR DISPLAY ZAI,VERIFY
STATUS:  ATTEMPTING SOCKET CONNECT
IXG601I  13.43.10  LOGGER DISPLAY 180
SYSTEM  LOGGER STATUS
SYSTEM  SYSTEM  LOGGER STATUS
-----  -----
SC43      ACTIVE
```

```
ZAI LOGSTREAM CLIENTS: ACTIVE
BUFFERS IN USE:  00 GB 0000 MB
ZAI VERIFY INITIATED,  CHECK FOR MESSAGES IXG37X, IXG38X
```

LOGGER PARAMETER OPTIONS

KEYWORD	SOURCE	VALUE
---------	--------	-------

ZAI

SERVER	IPV4	SET (xx) 9.12.5.xxx
PORT		SET (xx) 2001
LOGBUFMAX		SET (xx) 02
LOGBUFWARN		SET (xx) 90
LOGBUFFULL		SET (xx) MSG

```
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 184
FOR DISPLAY ZAI,VERIFY
STATUS:  SOCKET CONNECT SUCCESSFUL
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 185
FOR DISPLAY ZAI,VERIFY
STATUS:  INITIATING SOCKET VALIDATION
IXG386I ZAI LOGSTREAM CLIENT CONNECT ATTEMPT IN PROGRESS 186
FOR DISPLAY ZAI,VERIFY
STATUS:  SOCKET VALIDATION SUCCESSFUL
IXG380I ZAI LOGSTREAM CLIENT ESTABLISHED 187
FOR DISPLAY ZAI,VERIFY
```

Figure 4-47 Response to DISPLAY LOGGER,STATUS,ZAI,VERIFY command

You can verify that messages are being sent to IBM zAware from this system by issuing the following command:

```
DISPLAY  LOGGER,C,LSN=SYSPLEX.OPERLOG,D
```


The response to this command is shown in Figure 4-48. The response confirms that the log stream is connected to IBM zAware. It also provides information about the number of log blocks that were sent successfully, and the number (if any) that failed.

```

D LOGGER,C,LSN=SYSPLEX.OPERLOG,D
IXG601I 13.43.33 LOGGER DISPLAY 189
CONNECTION INFORMATION BY LOGSTREAM FOR SYSTEM SC43
LOGSTREAM          STRUCTURE          #CONN STATUS
-----
SYSPLEX.OPERLOG    SYSTEM_OPERLOG    000002 IN USE
  DUPLEXING: LOCAL BUFFERS
  GROUP: PRODUCTION ZAI CLIENT: YES - CONNECTED
  ZAIDATA: OPERLOG
    LOG BLOCKS SENT TO SERVER OK: 0000006302, FAILED: 0000000000
  JOBNAME: ZWARE3 ASID: 0025
    R/W CONN: 000001 / 000000
    RES MGR./CONNECTED: *NONE* / NO
    IMPORT CONNECT: NO
  JOBNAME: CONSOLE ASID: 000B
    R/W CONN: 000000 / 000001
    RES MGR./CONNECTED: *NONE* / NO
    IMPORT CONNECT: NO

NUMBER OF LOGSTREAMS: 000001

```

Figure 4-48 Response to `DISPLAY LOGGER,C,LSN=SYSPLEX.OPERLOG,D` command

If you need to disconnect the system from the IBM zAware LPAR, issue the following command:

```
SETLOGR FORCE,ZAIQUIESCE,LSNAME=SYSPLEX.OPERLOG
```

The response to this command is shown in Figure 4-49.

```

SETLOGR FORCE,ZAIQUIESCE,LSNAME=SYSPLEX.OPERLOG
IXG651I SETLOGR FORCE ZAIQUIESCE COMMAND ACCEPTED 727
FOR LOGSTREAM=SYSPLEX.OPERLOG
IXG382I ZAI LOGSTREAM CLIENT QUIESCED 728
FOR LOGSTREAM SYSPLEX.OPERLOG
REASON: SETLOGR COMMAND REQUEST.
IXG387I ZAI LOGSTREAM CLIENT CONNECTION SUMMARY 729
FOR LOGSTREAM SYSPLEX.OPERLOG
CONNECTION WAS ESTABLISHED AT: 07/17/2012 14:43:46 GMT
LOG BLOCKS SENT TO SERVER OK: 6386, FAILED: 0

```

Figure 4-49 Response to `SETLOGR FORCE,ZAIQUIESCE` command

After you disconnect from the IBM zAware server, use the following commands to verify the status:

```

DISPLAY LOGGER,STATUS,ZAI
DISPLAY LOGGER,C,LSN=SYSPLEX.OPERLOG,D

```

4.7 Bulk Data Load Utility

To be able to perform an analysis on messages arriving in real time, IBM zAware needs to have a model of the normal behavior of the system. Ideally the model will contain at least 90 days of messages. One way to achieve this is to feed messages in real time to the IBM zAware LPAR. However, that will result in a significant delay before you can start getting valuable information from IBM zAware.

A preferable option is to use syslog or OPERLOG archives to bulk load message data to IBM zAware.

In this section we describe how to use the Bulk Data Load Utility to send the past message data to the IBM zAware server, and then use training to create the model. This process involves the following steps:

- ▶ Prepare the input file to the Bulk Data Load Utility.
- ▶ Verify that the connection to IBM zAware is working.
- ▶ Perform the bulk load.
- ▶ Use the IBM zAware GUI to assign the system to the sysplex (this will cause IBM zAware to briefly drop the connection to all monitored clients).
- ▶ Ensure that the system successfully reconnected to the IBM zAware server.
- ▶ Train the system.

Next, we explain each step in more detail:

1. Prepare the input file to the Bulk Data Load Utility.

Typically, the message data that will be input to the Bulk Data Load Utility will come from either syslog or OPERLOG.

If you have existing sequential archive data sets containing data from syslog or OPERLOG, you can use them¹.

If you keep your message data in an OPERLOG log stream, use the IEAMDBLG program provided in SYS1.SAMPLIB(IEAMDBLG) to extract the required data into a sequential data set. This program reads records from an OPERLOG log stream and converts them to SYSLOG format. Sample JCL is shown in Figure 4-50.

You can also tailor the range of dates that will be extracted from the log stream. See the IEAMDBLG member in SYS1.SAMPLIB for information about filtering the data that is extracted from the OPERLOG log stream.

```
//COPYLOG JOB (999,P0K),MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=&SYSUID,  
//          CLASS=A,REGION=0M  
//SSS EXEC PGM=IEAMDBLG,  
//          PARM='COPY,HCFORMAT(CENTURY)'  
//SYSLOG DD DSN=KYNEF.OPERLOGS.JULY,  
//          DISP=(NEW,CATLG,DELETE),  
//          SPACE=(CYL,(99,199)),  
//          DCB=(RECFM=VB,BLKSIZE=0,LRECL=4096,DSORG=PS)
```

Figure 4-50 Sample IEAMDBLG job

¹ Note that the bulk data load exec does not support JES3 DLOG format. Therefore, the only way to perform a bulk load for a JES3 system is if you have OPERLOG archive files for that system.

If the message data is still in syslog files in the spool, you can use a product like SDSF or the external writer to save the syslog data to sequential data sets.

The provided sample REXX exec to upload the data to the IBM zAware server (AIZBLKE in SYS1.SAMPLIB) reads sequential data sets that contain SYSLOG data stored in hardcopy log 2-digit year (HCL) or 4-digit year (HCR) format. It also supports both VB and VBA format data sets.

2. Verify that the connection to IBM zAware is working.

You can perform the bulk load from any z/OS system that has access to the input data sets and has a connection to the IBM zAware LPAR. That is, it is *not* necessary for each system to load its own data. Before running the bulk load job, check that the connection is active with the following commands:

```
DISPLAY LOGGER,STATUS,ZAI
DISPLAY LOGGER,C,LSN=SYSPLEX.OPERLOG,D
```

Also check the connection status on the IBM zAware GUI by selecting the **System Status** option as shown in Figure 4-51. The connection should show a status of Active.

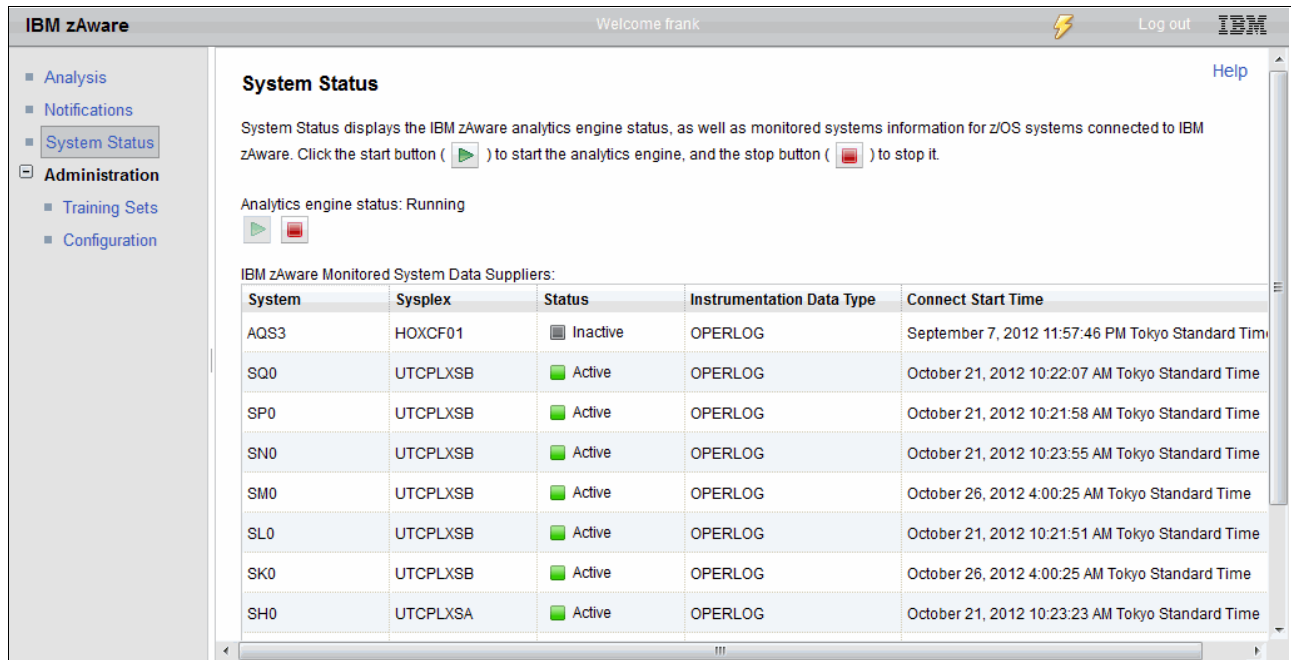


Figure 4-51 System Status on the IBM zAware GUI

3. Perform the bulk load.

The sample bulk load job is provided in SYS1.SAMPLIB(AIZBLK). Customize the sample job to match your data set names and submit the job. While the job is running, messages similar to the following are displayed:

```
IXG283I OFFLOAD DATASET IXGLOGR.BARTON.ZAI.LSTREAM.A0000024 576
ALLOCATED NEW FOR LOGSTREAM BARTON.ZAI.LSTREAM
CISIZE=4K, SIZE=11059200
IXG284I OFFLOAD DATASET IXGLOGR.BARTON.ZAI.LSTREAM.A0000023 577
DELETED FOR LOGSTREAM BARTON.ZAI.LSTREAM
```

When the job completes, check the actual output from the BULKLOAD step to ensure that it completed successfully (do not simply rely on looking at the return codes). Messages similar to those shown in Figure 4-52 are displayed.

```
READY
AIZBLKE BARTON.ZAI.CONTROL CREATECNTLS
An old copy of 'BARTON.ZAI.CONTROL' has been deleted.
Control dataset 'BARTON.ZAI.CONTROL' created.
AIZBLKE ended with a return code of 0.
READY
WHEN SYSRC(EQ 0) AIZBLKE BARTON.ZAI.CONTROL ADDSYSLOGDSN
KYNEF.OPERLOGS.JULY
Dataset 'KYNEF.OPERLOGS.JULY' added to 'BARTON.ZAI.CONTROL'
AIZBLKE ended with a return code of 0.
READY
WHEN SYSRC(EQ 0) AIZBLKE BARTON.ZAI.CONTROL DISPLAYCNTLS
Line 1 - /* List of SYSLOG files to be imported to zAware */
Line 2 - KYNEF.OPERLOGS.JULY
AIZBLKE ended with a return code of 0.
READY
WHEN SYSRC(EQ 0) AIZBLKE BARTON.ZAI.CONTROL IMPORT DUMMY BARTON.ZAI.LSTREAM
BARTON.ZAI.LSMODEL
Reading control dataset 'BARTON.ZAI.CONTROL'.
'KYNEF.OPERLOGS.JULY' is a non-ANSI data set.
Importing 2,500,000 records from 'KYNEF.OPERLOGS.JULY'.
All 2,500,000 records written.
Importing 129,951 records from 'KYNEF.OPERLOGS.JULY'.
All 129,951 records written.
269,662,371 bytes queued to zAware.
AIZBLKE ended with a return code of 0.
```

Figure 4-52 Sample output from BULKLOAD step

To confirm that the bulk load has completed, check the System Logger buffer usage using the following command:

```
DISPLAY LOGGER,STATUS,ZAI
```

4. Use the IBM zAware GUI to assign the system to the correct sysplex.

The message data that is sent to IBM zAware by the Bulk Data Load Utility contains the *system* name of each system (because that is contained in every message). However, it does not know the name of the sysplex that the system is a member of.

Therefore, you need to provide that information by assigning the new system to its correct sysplex using the **Administration** → **Configuration** → **Priming Data** tab.

Click each system you want to add to the sysplex in the **Priming message data by system** box.

System connection consideration: Before you can assign a system to a sysplex, at least one system in that sysplex must have connected to IBM zAware at least once.

Click the radio button next to the sysplex name for the systems.

Click the **Add** button. You will be presented with the confirmation window shown in Figure 4-53.

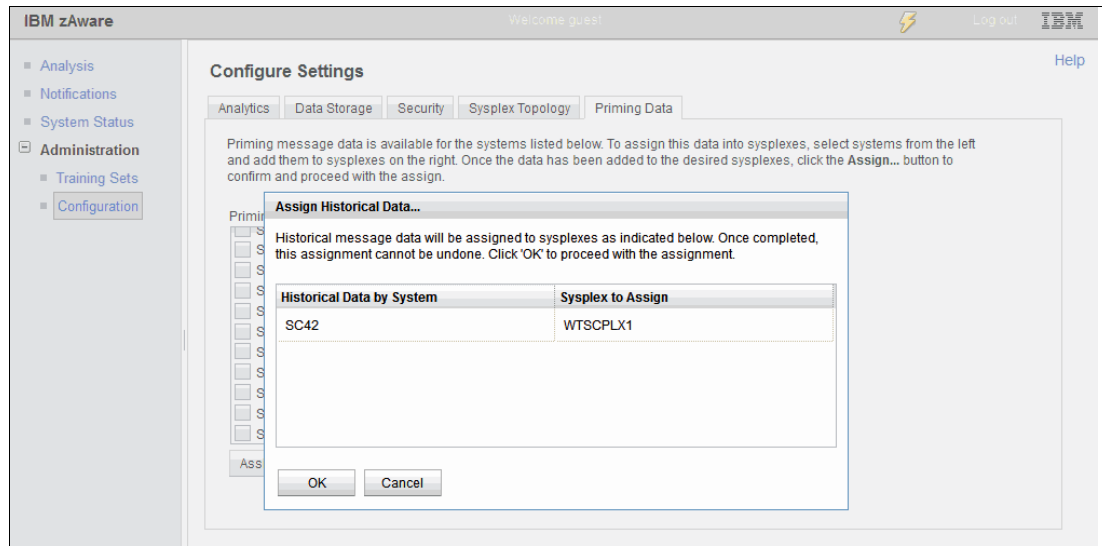


Figure 4-53 Assign systems confirmation window

Confirm that the system to sysplex mapping is correct and then press **OK**. The window shown in Figure 4-54 will display.

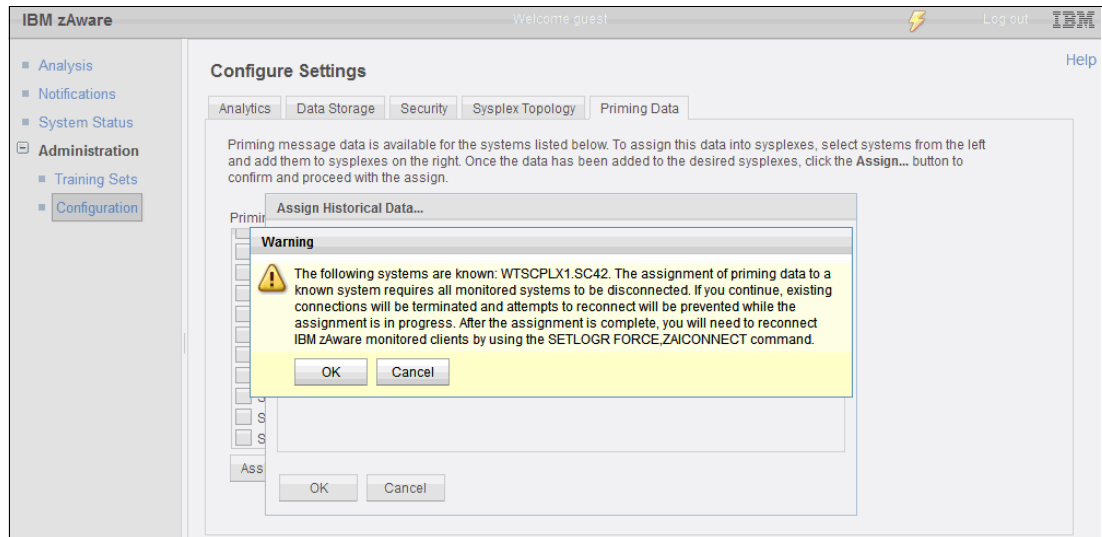


Figure 4-54 Assign systems warning message

5. Ensure that the system successfully reconnected with the IBM zAware server.

When you assign the bulk load data for a system to its sysplex, the analytics engine temporarily disconnected from all connected monitored clients. is recycled. When the assign processing completes, connections from the monitored clients can be restarted.

On the z/OS end, when System Logger sees the connection being terminated from the IBM zAware end, it will automatically attempt to reconnect. It will continue its reconnection attempts for up to 20 minutes.

If it is unable to successfully reconnect for some reason, issue the following commands:

DISPLAY LOGGER,STATUS,ZAI,VERIFY (Confirm the connection is established)

SETLOGR FORCE,ZAICONNECT,LSNAME=SYSPLEX.OPERLOG

6. Train the system.

Having performed the bulk load for the system and assigned it to the correct sysplex using the IBM zAware GUI Assign panel, you are ready to train the system.

Follow these steps:

- a. Logon to the IBM zAware application web page at:

https://ip_address/zAware/

Or use:

https://host_name/zAware/

- b. Click the **Training Sets** option within **Administration**.

- c. Locate the system you want to train and click the radio button to the left of the system name.

Training considerations: You can only train one system at a time, and the system to be trained must be connected to IBM zAware.

You can select multiple systems for training, but only one system will be trained at a time. Additional systems selected will be queued.

- d. Click the **Actions** button at the top of the list of systems.

- e. Click **Request Training** from the list of actions. You will receive a message about a model being queued for rebuild; click **OK**.

Figure 4-55 shows the result of requesting training on two systems. The first system has a status of *In Progress* and the second system shows *In Queue*. When training completes, the status will change to *Completed* or *Failed*. If the status is *Failed*, refer to 5.5.2, “Creating and updating the system model” on page 168, for information about this topic.

When a training is completed, analysis information about the monitored client will begin to be displayed on the **Analysis** page.

The screenshot displays the IBM zAware web interface. The top header shows 'zAware', 'Welcome noonan', 'Log out', and the IBM logo. A sidebar on the left contains navigation links: Analysis, Notifications, Administration (expanded), System Status, Training Sets (selected), and Configuration. The main content area is titled 'Training Sets' and shows 'Managed Systems'. Below this is a table with the following data:

System	Sysplex	Training Progress	Last Training Result	Last Training Result Time	Current Model Built
SC42	WTSCPLX1	In Progress	—	—	—
SC43	WTSCPLX1	In Queue(1)	—	—	July 20, 2012 5:17:28 PM Eastern Daylight Time

Below the table, there is a link for 'Current Training Status Details (Click on training statuses above to view details)' and a 'Refresh' button. The last refresh time is noted as 'Sun Jul 22 2012 13:24:20 GMT-0400 (Eastern Daylight Time)'.

Figure 4-55 Training status

Training considerations: The first training starts based on the first date that data is available for (loaded in the database) versus the current date (actually, yesterday). If this difference satisfies the training period (the default is 90 days), then IBM zAware will attempt to train automatically.

If you do not have enough data to satisfy the training period, you may choose to manually train with less data. After the first model is built successfully, “scheduled training” will be started based on the training interval (the default is 30 days), and is only driven for a connected monitored system.



Maintaining and managing IBM zAware

This chapter provides answers to many questions you might have about managing your IBM zAware environment.

The following topics are discussed:

- ▶ Managing IBM zAware components
- ▶ Starting and stopping the IBM zAware application
- ▶ Managing IBM zAware disks
- ▶ Managing connections from monitored clients
- ▶ Maintaining IBM zAware data
- ▶ Managing IBM zAware firmware
- ▶ Disaster recovery considerations
- ▶ Daily and weekly management tasks
- ▶ Checklist for adding a monitored client
- ▶ System Logger commands for IBM zAware support
- ▶ Problem determination for IBM zAware

5.1 Managing IBM zAware components

For those with lengthy experience in mainframe environments, the concept of having a function that does not require significant customization or day-to-day management might seem novel. It might recall the similar challenge of giving up control of your dispatching priorities to Workload Manager when WLM compatibility mode support was removed.

However, that is the intended mode of operation for IBM zAware. IBM zAware might be considered as an appliance, wherein you simply focus on how to use it and all the details about how it works are hidden from you. Across the IT industry, this concept of “black box” appliances, in which only their externally visible behavior is considered and not their implementation or inner workings, is becoming more and more common.

To give you an idea of the differences between z/OS and IBM zAware, in terms of day-to-day management tasks, Table 5-1 provides several comparisons.

Table 5-1 Comparison of z/OS and IBM zAware management actions

Task/Function	z/OS	IBM zAware
Syslog	Must be archived and managed.	Log is not accessible to you. Archiving and retention is handled transparently.
EREP	Should be analyzed and archived.	N/A
SMF	Must be offloaded, analyzed, and retained for differing periods of time.	N/A
RMF	Performance must be monitored and managed.	Ensure that configuration matches IBM guidelines. No performance tools to be monitored.
Fixes and preventative service	You have to order and apply preventative service and monitor for and apply HIPER service.	Service is managed by IBM hardware engineer, like all other hardware service.
Database backups	You have to schedule and manage backups and drive recovery if necessary.	IBM zAware creates its own database backups as necessary and performs recoveries if necessary.

Compared to managing a z/OS environment, managing an IBM zAware environment is by design much simpler. Getting used to this “hands-off” paradigm can sometimes be a challenge for traditional mainframe staff.

Another concept to keep in mind is that IBM zAware is not a production application in the traditional sense. For example, it does not contain operational data, so you might even decide not to back it up. Users who are logged on to it are not able to update any data, so bringing the LPAR down while users are logged on does not raise any loss of data or loss of update issues. And because it can be brought down and back quickly, taking an outage to implement a change does not have the same level of impact that recycling a z/OS image would have. These are all concepts that might take some time to get comfortable with.

Having said that, there are things in an IBM zAware environment that you need to manage. And there are processes that are conceptually similar to z/OS, but the mechanics of how they are carried out are different. The objective of this chapter is to provide you with all the information you need to effectively manage your IBM zAware environment.

5.2 Starting and stopping the IBM zAware application

IBM zAware does not have a traditional operating system console. Instead, the only interfaces you have for managing it are the HMC and the IBM zAware GUI. Because IBM zAware is contained in a special purpose LPAR, starting it consists of activating the LPAR from the HMC. After the LPAR has been activated, the LPAR status on the HMC will change to *Operating*. However, it takes between three and ten minutes to initialize all the components inside the partition. At that point, you will be able to logon to the IBM zAware GUI, and any monitored clients will be able to re-establish their connections to the IBM zAware LPAR.

Similar to how the IBM zAware LPAR is started, you also use the HMC to stop it. The shutdown process is simple. It is *not* necessary to manually stop the connections from the monitored clients. The benefit of not stopping the connections from the monitored clients is that the connected System Logger will interpret the interrupt as a temporary event and will continue to accept messages, storing it in its buffers while it attempts to reestablish the connection to IBM zAware.

When System Logger is able to reconnect, it will send all the messages that were queued. The Analysis view will be updated with the messages that were issued during the period when the IBM zAware application was inactive.

If you quiesce the connections before you recycle the IBM zAware LPAR, System Logger will stop buffering messages, and the connections to IBM zAware would *not* be automatically re-established when it comes back up.

There is no mechanism for determining whether any users are currently logged on to IBM zAware. However, even if there are users logged on, they cannot be in the middle of any updates, and stopping and starting IBM zAware does not take long, so they will quickly be able to log back on again.

When you perform a Deactivate from the HMC, this results in a request being sent to the IBM zAware application to begin stopping all of its components. This will close the connections to all monitored clients, quiesce any database requests, and then shut down. This can take a few minutes, depending on the activity within IBM zAware at the time you issue the Deactivate request. When the deactivation completes, you will see a status of Success on the HMC.

Note that System Logger will only try to restart the connection for about 20 minutes. If it is not able to reconnect within that time, it will quiesce the connection and discard any messages that were queued in its buffers. Therefore, if you need to recycle the IBM zAware LPAR, try to ensure that it is stopped and restarted as quickly as possible.

Performing a planned shutdown of IBM zAware: It is common to restart z/OS LPARs by simply performing an Activate, without first performing a Deactivate. That is because you will have performed a planned shutdown of z/OS prior to performing the Activate.

However, the only way to perform a planned shutdown of IBM zAware is to perform a Deactivate from the HMC. If you perform an Activate without a preceding Deactivate, you introduce the risk of corrupting the IBM zAware database.

5.2.1 Starting and stopping the IBM zAware Analytics Engine

Among other things, the Analytics Engine manages the connections from the monitored clients. If you have a reason to temporarily close the connections from all monitored clients, you can stop the Analytics Engine from the System Status window as shown in Figure 5-1.

During the time the Analytics Engine is stopped, monitored clients cannot connect to the IBM zAware application, and no real-time data will be processed. When the Analytics Engine status is stopped, the connections for all monitored clients change to INACTIVE in the IBM zAware System Status window. On the z/OS end, the connection status changes to YES - CONNECTING.

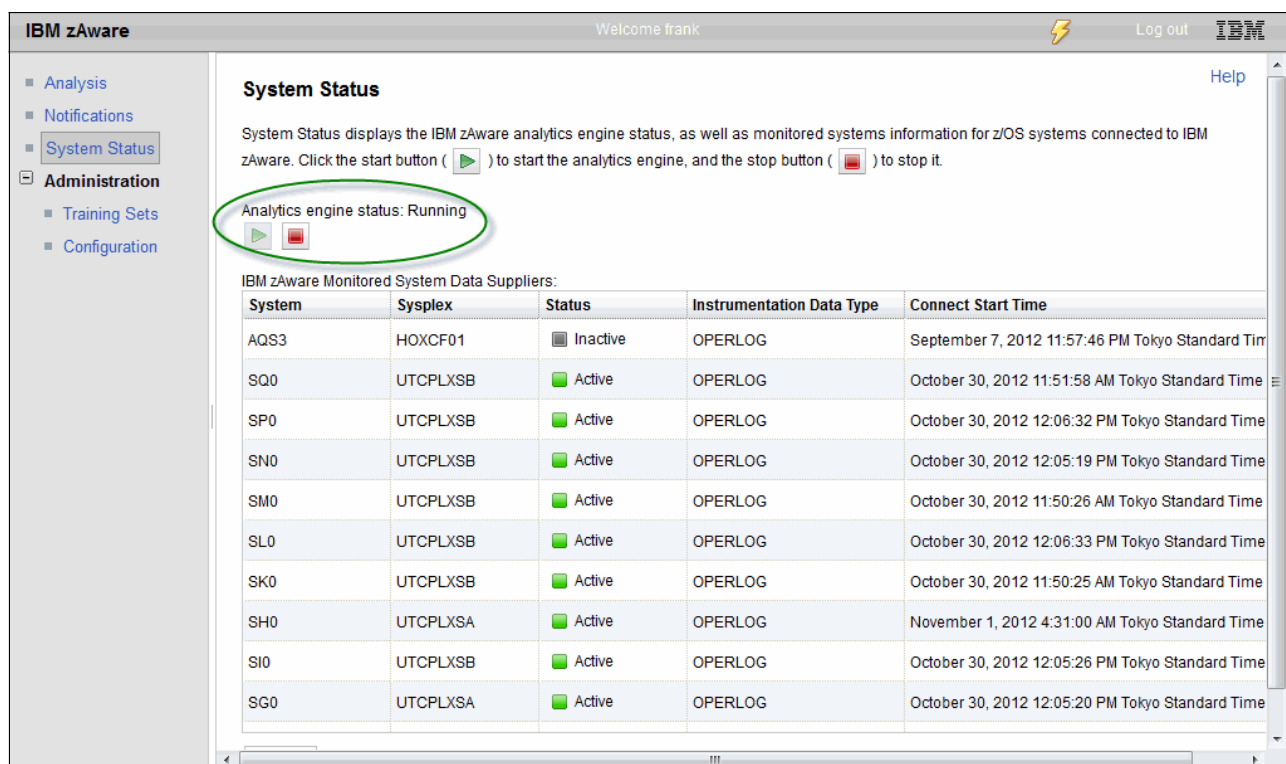


Figure 5-1 Starting and stopping Analytics Engine

This has a number of benefits compared to stopping the connections from the z/OS end:

- All connections can be stopped from a single place, rather than having to issue a command on every monitored client.
- Because the monitored clients see this as a temporary error, they will continue to accept and buffer new message for later transmission to IBM zAware.

- The connections can be re-enabled from a single place. Note that you cannot restart the connections from the IBM zAware. However, System Logger will automatically try to restart the connection for up to 20 minutes. If you restart the Analytics Engine within 20 minutes, the connections should all be restarted with no further manual intervention required.

Other than this situation of closing the connections from all monitored clients, we do not see any other case where you might need to stop the Analytics Engine. If you are recycling the IBM zAware LPAR, then performing the Deactivate from the HMC will stop the Analytics Engine and all other IBM zAware components. Therefore there is no need to manually stop it as part of shutting down the IBM zAware LPAR.

5.2.2 IBM zAware LPAR automation considerations

Most z/OS installations have developed automation routines to manage the shutdown and startup of their z/OS systems. Part of the reason for that is that the startup and shutdown typically involves tens or hundreds of started tasks and trying to control all of them manually is not a realistic option.

However, IBM zAware does not involve any of this complexity, so there is no need for automation to be involved in the startup or shutdown of an IBM zAware LPAR. IBM zAware does provide an API, and that API can be used for monitoring the health of the LPAR to some extent. But the API does not provide any functions to start or stop the IBM zAware LPAR, so there are no considerations for involving your z/OS automation tools in stopping or starting IBM zAware.

5.2.3 IBM zAware support for dynamic configuration changes

z/OS provides extensive support for dynamic configuration changes. For example, you can add and remove devices, configure channels offline and online, and add more general purpose or specialty engines, all without an IPL. The primary reason for this great flexibility is because of the stringent requirements for the highest levels of availability.

IBM zAware is not expected to have the same level of availability requirements that z/OS does. And the number of people using an IBM zAware LPAR at a given time is likely to be a small fraction of the number that are typically using a z/OS system.

For this reason, and also to keep IBM zAware as simple to operate as possible, most configuration changes (adding new network adapters, taking an adapter offline for service, adding engines, and so on) require that you deactivate the IBM zAware LPAR, make the required changes to the LPAR profile, and then reactivate the LPAR again. Because it only takes a few minutes to recycle the IBM zAware LPAR, this should not be a significant inconvenience. See “Adding disks to the IBM zAware LPAR” on page 161 for information about IBM zAware support for adding disks using Dynamic I/O Reconfiguration support.

5.3 Managing IBM zAware disks

The IBM zAware databases are kept on CKD DASD that you assign to the IBM zAware LPAR. This section discusses manual actions related to the IBM zAware disks that you might need to perform.

5.3.1 User ID authority requirements

All IBM zAware GUI user IDs have one of two levels of authority (known as “roles” in IBM zAware terminology):

- ▶ USER
- ▶ ADMIN

All disk management actions must be performed by a user ID defined with ADMIN authority. Attempts to access the disk administration functions from a user ID with insufficient authority will result in errors being returned to the user.

5.3.2 Adding disks to the IBM zAware configuration

All IBM zAware disk management functions (except for backing up the disks) are performed from the IBM zAware GUI. To access these functions, click **Administration** → **Configuration** → **Data Storage**. This will bring you to the panel shown in Figure 5-2.

The screenshot shows the IBM zAware GUI interface. On the left is a navigation menu with 'Administration' expanded, showing 'System Status', 'Training Sets', and 'Configuration'. The main area is titled 'Configure Settings' and has tabs for 'Analytics', 'Data Storage', 'Security', 'Sysplex Topology', and 'Priming Data'. The 'Data Storage' tab is active, showing storage statistics: Total capacity (GB): 22.15, Total storage used (GB): 2.32, and Total storage used (%): 10.48. Below this is a table titled 'Data Storage Devices' with columns for Device, Status, Device Type, and Capacity (GB). The table lists 15 devices, with the first three in 'In Use' and the rest 'Available'. A 'Refresh' button and 'Last Refresh' timestamp are at the bottom of the table.

Device	Status	Device Type	Capacity (GB)
9407	In Use	3390/0c	7.38
9409	In Use	3390/0c	7.38
9408	In Use	3390/0c	7.38
9b49	Available	3390/0a	—
9b48	Available	3390/0a	—
9b41	Available	3390/0a	—
9b40	Available	3390/0a	—
9b43	Available	3390/0a	—
9b42	Available	3390/0a	—
9b45	Available	3390/0a	—
9b44	Available	3390/0a	—
9b47	Available	3390/0a	—

Figure 5-2 Data Storage Devices list

The first list contains all devices that accessible to the IBM zAware LPAR. The list shows the device number, status, and device type of each device. The columns can be sorted by clicking their headings so you can group the devices by their status or device type, or list them by device number.

Use care when adding devices to avoid overwriting a device: If you did not specify an Explicit Device Candidate List in HCD, the list of devices is likely to include those that are in use by other LPARs.

Exercise caution when adding devices, because if you accidentally add a device that is in use by another LPAR, IBM zAware will *overwrite* the disk contents to create its file system.

Note that this list is only for information and status purposes. If you want to make a change to the configuration, you must click **Add and Remove Devices**. This results in the Add and Remove Devices page (shown in Figure 5-3) being displayed.

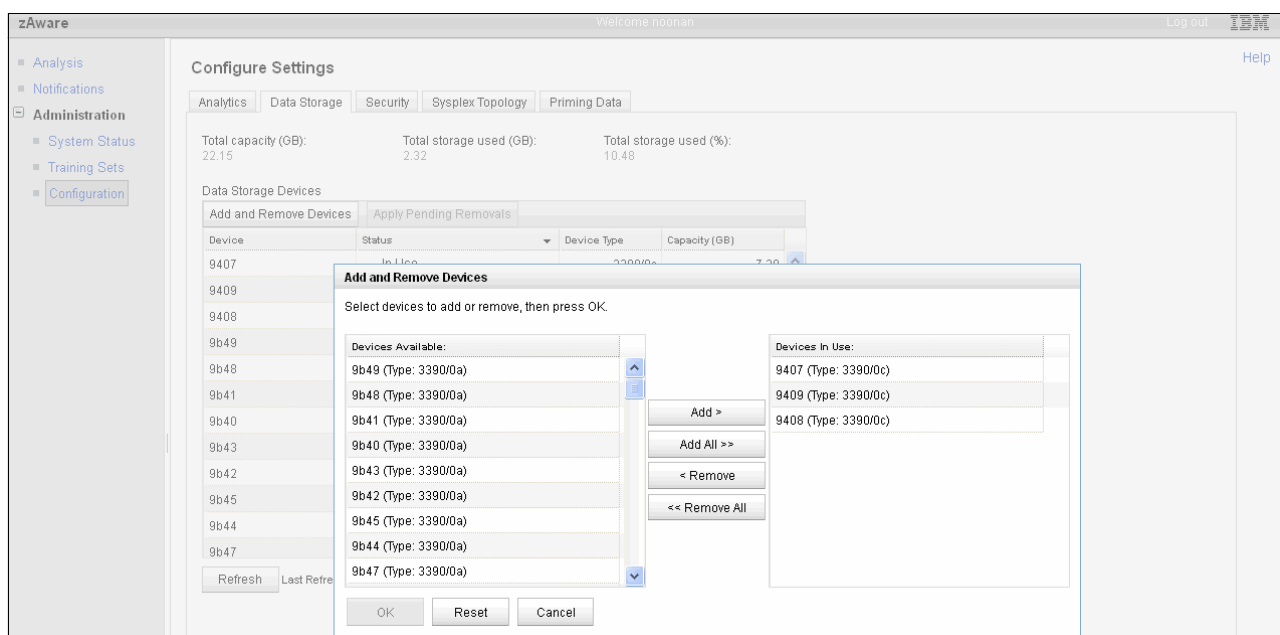


Figure 5-3 Add and Remove Devices page

This is the window to use to add or remove devices from IBM zAware. The left pane contains all the devices that are not currently in use by this LPAR. The devices in the right pane are those that are currently in use by this LPAR.

Follow these steps to add disks to the IBM zAware configuration (this will automatically add the IBM zAware file system to the device and remove any other data that might have been present on the device):

1. In the left pane, click each device you want to add to the IBM zAware file system.

Each device will become highlighted.

2. Click the **Add** button.

This will move those devices to the right pane. At this point nothing has been done to the devices.

3. Click **OK**.

Be aware: This step will erase any data on the device.

4. You will now be taken back to the previous window. If you sort the “Status” column, the devices that were just added will show as Being Added and then In Use when they have been initialized and the file system has been allocated and formatted.

After you add a device, the storage summary information near the top of the page will be updated. The Total storage used (%) value will decrease, because you have added more space. The Total capacity (GB) and Total storage used (GB) values will increase.

Volume serial number is changed when a disk is added: As part of the formatting that takes place when you add a disk to the IBM zAware configuration, the volume serial number will be changed to 0Xdddd, where dddd is the device number of that volume.

5.3.3 Removing disks from the IBM zAware configuration

The same panels that you use to add disks to IBM zAware are used to remove disks from it, as shown in Figure 5-4.

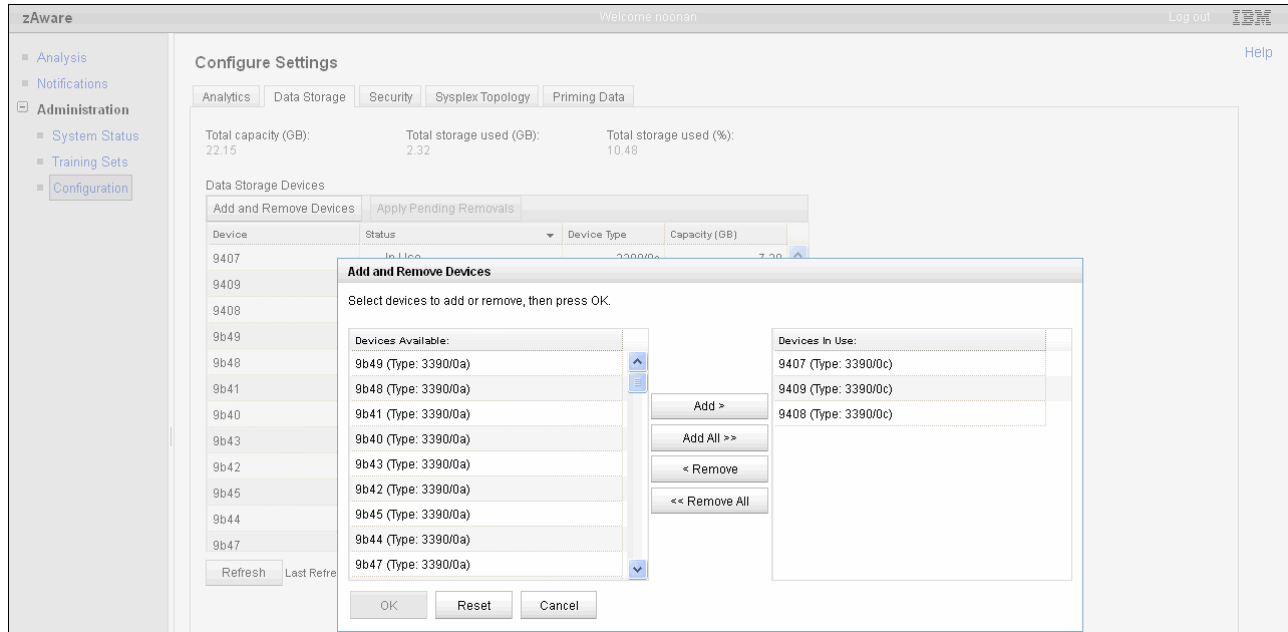


Figure 5-4 Add and Remove Devices window

Again, the devices that IBM zAware is not currently using are listed in the left pane, and the devices that are currently in use are contained in the right pane. When you indicate that a disk is to be removed from the IBM zAware configuration, any data on that disk is automatically moved by IBM zAware to another device within the file system. There is *no* data loss. To remove a disk, perform the following steps:

1. Click each device you want to remove from the IBM zAware file system.

Each device will become highlighted.

2. Click the **Remove** button.

This will remove the device from the right box titled “Devices In Use.” At this point nothing has been done to the device.

3. Click **OK**.

You will be taken back to the first window. The devices that you removed will now show a status of Pending Removal. The storage summary fields will temporarily be blanked out and the Add and Remove Devices and Apply Pending Removals buttons will both be grayed out. IBM zAware now identifies any used file space that must be moved from the device to the remaining devices. Depending on the amount of data to be moved, this might take a few minutes.

When that processing completes, the **Apply Pending Removals** button will become live.

4. Click **Apply Pending Removals**.

A warning pop-up message will appear, describing the process that is about to take place, and pointing out that it might take some time to complete. You must click **OK** for processing to proceed.

When the processing completes, the device status will change to `Available`, meaning that it is no longer part of the IBM zAware configuration.

Adding disks to the IBM zAware LPAR

IBM zAware is not able to initiate a dynamic I/O reconfiguration on the CPC that it is running on. However, if a z/OS or z/VM LPAR performs such a configuration, any disk devices that are added to the IBM zAware LPAR will automatically appear in the `Devices Available` list on the `Data Storage` tab on the IBM zAware GUI.

If you are running IBM zAware on a CPC that does not have any z/OS or z/VM LPARs, then you must perform a power-on reset (POR) of the CPC to pick up a new I/O configuration. Following the POR, the IBM zAware LPAR will be able to see the new full set of disk devices that are defined to it.

5.3.4 Backing up the IBM zAware file system

The IBM zAware file system contains information from the monitored clients. It is constantly being updated as new data arrives from the monitored clients. And it contains the information that is required for it to successfully analyze the message traffic on each system. Specifically, the file system contains the following information:

- ▶ The model database, containing information about every monitored client
- ▶ The instrumentation data database, containing the information that will be used to update the model database
- ▶ Information about the sysplex topology
- ▶ The XML files, containing the Analysis results from the past

The IBM zAware file system is not a published interface, and you generally do not require any knowledge or understanding of it. However, there is one aspect of how it works that is important for you to know. The bulk of the IBM zAware information is kept in the file system on the CKD disks. However, IBM zAware also keeps some information in the Support Element of the CPC that it is operating on. And there is information in the Support Element that must match information in the file system on the CKD disks.

One part of that information is the device number of the volume that each part of the file system resides on. If there is a mismatch between the information in the Support Element and the file system on disk, or if either of those is missing, IBM zAware will not use the file system on the affected volumes. This imposes several restrictions on you:

- ▶ If you back up an IBM zAware disk, and then need to use that backup, it must be restored to a device with the same device number as the original volume.
- ▶ If you try to move IBM zAware to another CPC, the information that resided in the Support Element on the original CPC will not reside in the Support Element on the new CPC, meaning that it will not be able to use the CKD disks.
- ▶ If you mirror the CKD disks to another disk subsystem, IBM zAware will not be able to use the mirrored file system because the device number of the mirror will be different from the device number of the primary device.

This does not mean, however, that you cannot back up IBM zAware file system. It simply means that you need to keep these restrictions in mind when you are considering your backup options.

You have the following choices for handling a failure that impacts your IBM zAware file system:

- ▶ Do not take any backups of the file system. If you lose the file system, you would use archived message data from the connected systems to rebuild the database.
- ▶ Create full volume backups of the IBM zAware volumes from z/OS using a product such as DFSMSdss.
- ▶ Create backups of the IBM zAware volumes using a product such as IBM FlashCopy®.

Because IBM zAware does not provide a facility to perform logical file system backups, any backup that you perform will have to be a physical backup, at the volume level. In the following sections we briefly discuss the considerations for each of these options.

Do not take any backups

If you choose not to back up the volumes, and a hardware problem that affects one or more volumes containing the IBM zAware file system partition occurs, you will effectively lose access to all the information in the file system.

In this case, you would have to remove all the devices from the IBM zAware configuration, and then add them back again, to reinitialize the database. You would then connect the monitored clients and perform a bulk load for each system. Next, you would go through the Assign process. Finally, you would train them all to create the new models. This effectively initializes the IBM zAware application, which will be able to continue with its analysis. However, in the Analysis view you would only see analysis results appearing after the new model was created. Remember that IBM zAware does not create Analysis results for data that is submitted using the bulk load process.

The benefit of this approach is that it is simple. There is no ongoing cost or overhead in creating backups that you might never need.

The disadvantages of this approach are:

- ▶ If you have many monitored clients, having to run a bulk load on all of them is a laborious and time-consuming process.
- ▶ You will lose all the Analysis results.
- ▶ It is likely to be some time before you have IBM zAware running again.

Create full volume backups using DFSMSdss

The next option is to create full volume backups using a product like DFSMSdss. Given that the IBM zAware file system is probably spread across multiple volumes, you need to back up every volume at the same point in time. One way (although not a desirable way) to do this would be to deactivate the IBM zAware LPAR while the backup is running to ensure that no updates are taking place during the backup.

A better option would be to use the DFSMSdss FlashCopy Consistency Group feature. This allows you to take a consistent point-in-time backup across multiple volumes without having to quiesce activity to those volumes. Depending on your DASD subsystem, the use of FlashCopy places restrictions on the location of the source and target volumes.

Whichever option you select, it is vital that the source volumes are consistent for the backup. If you just perform a normal disk copy while the volume is being updated, it is likely that the resulting output volume will not be usable if you ever need to use it.

The benefits of this option (if you use the FlashCopy Consistency Group feature) are:

- ▶ The backup will be consistent, meaning that there should be no database integrity issues if you need to use it.
- ▶ It does not require manual intervention to stop IBM zAware from updating its disks.

The disadvantages is that the IBM zAware volumes must be online to the z/OS system.

If you do not want to dedicate another set of disks to act as the FlashCopy target, you can take full volume dumps to tape. However, the only way to achieve consistency across all IBM zAware volumes in that case is to deactivate the LPAR while the backups are running.

Create a copy using FlashCopy Consistency Groups

This option is similar to the DSS option, with the exception that the IBM zAware volumes do not need to be online to any z/OS system. Note, however, that they *do* need to be defined in the I/O configuration for the z/OS system that you will run the backup job on. Figure 5-5 shows an example configuration, where the source volumes (device numbers 9487-9489) are in the configuration of both the IBM zAware LPAR and the z/OS LPAR. The target volumes (device numbers 948A-948C), however, are only in the z/OS configuration.

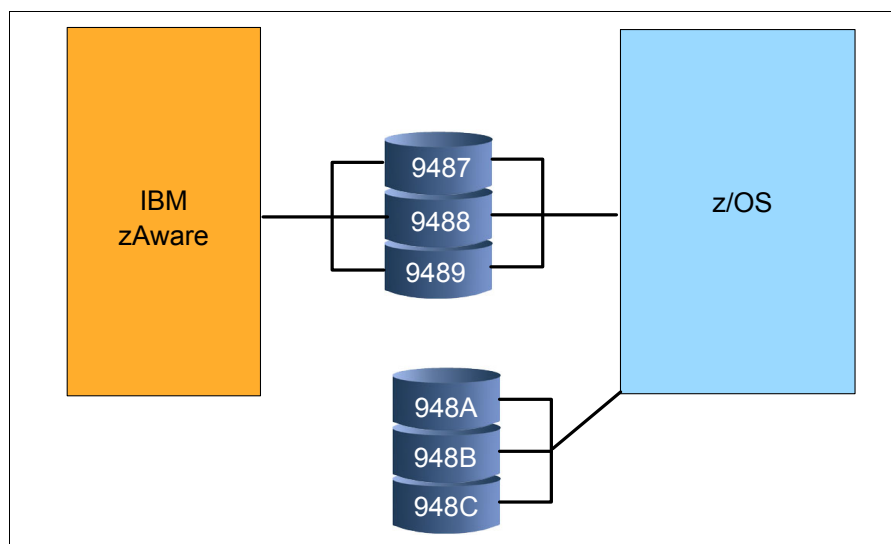


Figure 5-5 IBM zAware devices defined to z/OS for backup

A sample set of JCL to create a consistent backup of three volumes is shown in Example 5-1.

Example 5-1 Sample job to take point in time FlashCopy of IBM zAware disks

```
//KYNFFC JOB (0,0),'FC IBM ZAWARE',CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*****
/* ESTABLISH FLASHCOPY RELATIONSHIP *
/* SDEVN - SOURCE FLASHCOPY VOLUME *
/* TDEVN - TARGET FLASHCOPY VOLUME *
//*****
//STEP1 EXEC PGM=IKJEFT01,REGION=256K
//SYSTSPRT DD SYSOUT=*
//SYSUADS DD DSN=SYS1.UADS,DISP=SHR
```

```
//SYSLBC DD DSN=SYS1.BROADCAST,DISP=SHR
//SYSTSIN DD *
FCESTABL SDEVN(X'9487') TDEVN(X'948A') ACTION(FREEZE)
FCESTABL SDEVN(X'9488') TDEVN(X'948B') ACTION(FREEZE)
FCESTABL SDEVN(X'9489') TDEVN(X'948C') ACTION(FREEZE)
//*
//*****
//* WITHDRAW FLASHCOPY RELATIONSHIP *
//* DEVN - ANY FLASHCOPY VOLUME *
//*
//*****
//STEP2 EXEC PGM=IKJEFT01,REGION=256K
//SYSTSPRT DD SYSOUT=*
//SYSUADS DD DSN=SYS1.UADS,DISP=SHR
//SYSLBC DD DSN=SYS1.BROADCAST,DISP=SHR
//SYSTSIN DD *
FCWITHDR DEVN(X'9487') ACTION(THAW)
/*
```

Restoring the IBM zAware file system from a backup

If you have made a full volume backup or a FlashCopy of the IBM zAware devices, and there was a problem with one IBM zAware device, then every device must be restored. This is because the file system spans across the multiple devices, and restoring an individual device will not restore the database.

During any restore process, the IBM zAware partition must be deactivated.

When restoring the volumes, ensure that each backup is restored to the same volume (same device number) that it resided on at the time of the backup. If you restore to different device numbers, the information in the file system will not match the device number that the file system resides on, and IBM zAware will not use that file system.

When restored, the information in the database will be current at the time of the backup. This means the days between the backup and the restore date will have no analysis results. This will show in the Analysis Results panel as bars with no anomaly score and no message IDs.

Remember that you always have the fallback of initializing a new IBM zAware database and restoring the contents using a bulk load, as described in “Do not take any backups” on page 162. This is not ideal, but it is a fallback if all else fails. However, for this to be an option, you need to ensure that archived message data is kept for at least 90 days on all monitored clients.

5.3.5 Sharing disks between IBM zAware LPARs

It is not possible to share IBM zAware disks between multiple IBM zAware LPARs.

Consider that you are currently running IBM zAware in LPAR A10. The information stored in the Support Element is specific to that IBM zAware LPAR. This means that if you initialize IBM zAware in a different LPAR, for example A20, that LPAR will not have any information about the disk volumes. If you try to add the disks that were being used by the A10 LPAR to the IBM zAware in LPAR A20, it will consider those to be new disks and will initialize them.

5.3.6 Backup IBM zAware LPARs

We cannot think of a reason why you would want to move IBM zAware from one LPAR to another LPAR on the same CPC. But if you do make such a move, be aware that you will effectively be starting with an empty set of databases. This is because IBM zAware in the target LPAR will initialize any disks that you add to it, as described in 5.3.5, “Sharing disks between IBM zAware LPARs” on page 164.

Also, you cannot use any backups to recreate the volumes because the information in the backups will be specific to the LPAR that created that file system.

Disaster recovery considerations for IBM zAware are discussed in 5.7, “Disaster recovery considerations” on page 178.

5.4 Managing connections from monitored clients

There must be a connection from each monitored client to the IBM zAware LPAR. In a sysplex, even though all the systems might be connected to the SYSPLX.OPERLOG log stream, each system that is a monitored client of IBM zAware still needs its own connection to IBM zAware. For an example, consider the diagrams in Figure 5-6.

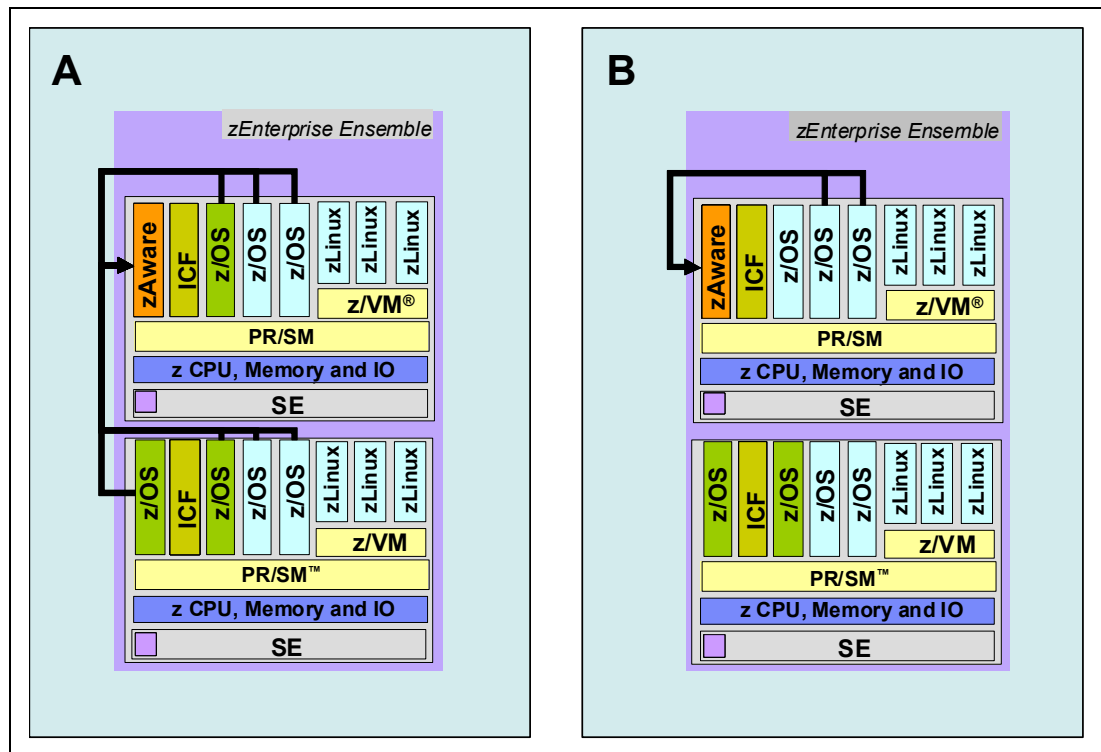


Figure 5-6 Connections to IBM zAware Server

Diagram A in Figure 5-6 shows two sysplexes in which all systems are monitored clients of IBM zAware and each system has a connection to IBM zAware. Diagram B shows that you can connect a subset of systems in a sysplex as monitored clients. This sysplex contains three test systems and two production systems, but only the production systems are to be monitored clients of IBM zAware. Therefore, only the two production systems in that sysplex have a connection to IBM zAware.

During normal processing, there should be no need to stop the connection between a monitored client and IBM zAware. If the connection goes down for some reason, System Logger will try for up to 20 minutes to reconnect. If the reason that the connection went down was that the IBM zAware LPAR reIPLed itself (perhaps after some service was applied), or because you deactivated and reactivated the LPAR, then it should be back before System Logger gives up trying to connect. During this 20-minute window, Logger will continue to accept new messages and save them in its buffer, so no message data will be lost.

If the 20 minutes expires and Logger is still unable to restart the connection, or if some other event causes it to give up, Logger will issue an IXG386I message with one of the following texts:

- ▶ STATUS: IP ADDRESS RETRIEVE FAILED
- ▶ STATUS: SOCKET CREATE FAILED
- ▶ STATUS: SOCKET CONNECT FAILED
- ▶ STATUS: SOCKET VALIDATION FAILED

It is advisable to implement some automation that would monitor for these messages and, at a minimum, update the status of the IBM zAware resource to reflect the fact that at least one system has lost connectivity.

Another potential reason for the connection between a monitored client and IBM zAware becoming disconnected is that the z/OS system might be IPLed. When z/OS is shut down, it will close its connection to IBM zAware, and any messages that are still in its buffer at that point will be purged.

After the IPL, the IBM zAware server is defined in the IXGCNFxx member and the required network connectivity is in place, so System Logger will attempt to restart the connection. In fact, it will even try to restart the connection if that connection was quiesced prior to the IPL. Depending on how long it takes to start OMVS and TCP/IP, you might see some IXC3xx messages, indicating that Logger is trying to start the connection to IBM zAware. It will continue to attempt to start the connection for 20 minutes or until it is successful, whichever occurs first. During this time, Logger will be accepting messages and storing them in its buffer, waiting for the connection to IBM zAware to come active. If the buffer fills up during this period, new messages are not added to the buffer.

Although this explanation covers most of the situations where the connection might be lost, it is possible that other events can cause the link to go down. To detect this situation, there are a number of things you can do:

- ▶ Check the IBM zAware GUI System Status window. Investigate systems showing a status of Inactive that you expect to be active.
- ▶ Check the IBM zAware GUI Analysis view. You would normally expect to see at least a small light blue bar for each system. If any system has an anomaly score of 0.0 and a Unique Msg IDs count of 0, that is an indication of a possible connection problem.
- ▶ Issue a **D LOGGER,C,LSN=SYSPLEX.OPERLOG,D** command. The output will contain the ZAI CLIENT: field, which can have a value of:
 - YES - QUIESCED
 - YES - CONNECTING
 - YES - CONNECTED

If any of these tests indicate that the connection is unavailable, but both z/OS and the IBM zAware LPARs are available, try restarting the connection using the **SETLOGR FORCE,ZAICONNECT,LSN=SYSPLEX.OPERLOG** command. If this is not successful, refer to 5.11, “Problem determination for IBM zAware” on page 183 for information about

troubleshooting connection problems. Appendix C, “Using automation to monitor IBM zAware connections” on page 219 contains a REXX exec that can be used by your z/OS system automation product to query the status of the IBM zAware connections.

Access to the SERVAUTH profile: If you have AT-TLS activated for Secure Sockets, you will need to give System Logger access to the SERVAUTH profile `EZB.INITSTACK.sysname.tcpname` in RACF. This will let Logger connect to the IP stack and open a socket connection with the IBM zAware server before AT-TLS becomes active. If you do not add access to this profile, the attempt to start the connection might fail during an IPL.

You might also consider using the IBM zAware Application Programming Interface (API) to check that IBM zAware is receiving and analyzing information for each interval. This offers the benefit of not requiring manual intervention unless a problem is detected. It also is likely to detect a potential problem in a more timely manner than manual checking. For information about using the API, refer to 6.3, “IBM Tivoli NetView for z/OS” on page 196.

If you have many monitored clients, monitoring the status of the connections from so many systems might be a challenge. The easiest way to view the connection status of all systems is to use the IBM zAware System Status window. Additionally, if you use your automation to consolidate information from across the enterprise, you might consider getting every system to forward the status of its IBM zAware connections to a focal point. That focal point would provide an at-a-glance status of the health of all IBM zAware connections.

Note that if you stop the Logger connection to the IBM zAware server using the **SETLOGR FORCE,ZAIQUIESCE,LSN=SYSPLEX.OPERLOG** command, any messages received for transmission to IBM zAware will not be sent, and any messages that are already in the buffer will be purged.

5.5 Maintaining IBM zAware data

Most of the management of the information in the IBM zAware database and file system is handled automatically by IBM zAware, based on the customization that you provide.

This section discusses how you get data into IBM zAware, what influence you have over database updates, and the considerations for how long to retain the data in IBM zAware.

5.5.1 Bulk Data Load Utility

The Bulk Data Load Utility reads data sets containing historical message data that has been extracted from syslog or OPERLOG. Each message that is read is then written to a log stream. That log stream has been defined with the ZAI parameters, indicating that all blocks written to it are to be forwarded to IBM zAware.

z/OS installations typically store their system log in one of these places:

- ▶ In the OPERLOG log stream. The messages are kept in the log stream until they reach their retention period as specified in the log stream attributes.
- ▶ In a set of sequential data sets that are extracted from OPERLOG using the IEAMDBLG program.
- ▶ In the JES spool as SYSLOG files.
- ▶ In a set of sequential data sets that are created from the syslog spool files.

Regardless of where you keep your message data, you need to prepare files that will be input to the Bulk Data Load Utility. The utility can handle both 2-digit year and 4-digit year format¹, and files that contain carriage control characters or do not contain carriage control.

JES3 consideration: At the time of writing, the Bulk Data Load Utility does not support JES3 DLOG data sets.

If you have JES3 and want to prime the IBM zAware database with historical message data, you must use OPERLOG and extract the input files from OPERLOG.

The Bulk Data Load Utility consists of a batch job (provided in member AIZBLK in SYS1.SAMPLIB) that performs the following tasks:

- ▶ Defines a model log stream.
- ▶ Runs a REXX exec that reads the messages from the input data set and writes them to a temporary log stream. The REXX exec is provided in member AIZBLKE in SYS1.SAMPLIB.
- ▶ Deletes the temporary log stream and the model log stream.

Copy the job JCL and the REXX exec to your own libraries in preparation for any customization. The JCL clearly documents the changes you need to make to the JCL.

The job can be run multiple times for each system or sysplex. Any duplicate message information will simply overlay the existing entries in the database, so you do not need to be concerned if you accidentally send the same data more than once.

5.5.2 Creating and updating the system model

Having bulk loaded your data into IBM zAware and assigned the data to the correct sysplex, the next step is to create the model database with the message information for your system. This is known as “training” the system. The training process is used both for loading the initial set of information about a system (known as priming the database) and for then updating the model with information from the messages that are received in real time.

It is vital to remember the objective of IBM zAware: to make you aware of differences between the current message behavior and what the system normally looks like. But anyone who has experience managing z/OS systems knows that there are many “normals”. For example, weekdays look different than weekends. Prime shift is different than night shift. Month-end might look different than the rest of the month.

Therefore, to build a model of what the system normally looks like, IBM zAware needs as much historical information (within reason) as possible. If it does not find sufficient message diversity and sufficient patterns (for example, IPLs, subsystems starting and stopping, the network coming up and down, and so forth), it will refuse to create a model of that system.

The alternative is that it will have an inaccurate picture of the normal activity on a system, and will flag normal messages as anomalies. If that were to happen people would learn to ignore the analysis provided by IBM zAware, and thereby potentially miss critical messages.

¹ The year format (2-digit or 4-digit) is controlled by the HARDCOPY HCFORMAT keyword in the CONSOLxx member of Parmlib.

Initiating training

To initiate a manual training for a system, go to the Training Sets panel, as shown in Figure 5-7.

Training Sets

The Monitored Systems table provides training statuses and results for IBM zAware monitored systems. The Actions menu provides functions for managing model dates, requesting or canceling training, and managing ignored messages. Training details for a given system can be accessed by clicking on links in the Training Progress and Last Training Result columns.

Monitored Systems

System	Sysplex	Training Progress	Last Training Result	Last Training Result Time	Current Model Built
<input type="radio"/> AQS3	HOXCF01	—	✓ Complete	July 26, 2012 12:29:57 AM Tokyo Standard Time	July 2 Stand
<input type="radio"/> SC0	UTCPLXSA	—	✓ Complete	October 25, 2012 9:00:40 AM Tokyo Standard Time	Octob Tokyo
<input type="radio"/> SF0	UTCPLXSA	—	✓ Complete	October 21, 2012 9:55:23 AM Tokyo Standard Time	Octob Tokyo
<input type="radio"/> SG0	UTCPLXSA	—	✓ Complete	October 25, 2012 9:00:18 AM Tokyo Standard Time	Octob Tokyo
<input type="radio"/> SH0	UTCPLXSA	—	✓ Complete	November 1, 2012 9:01:09 AM Tokyo Standard Time	Nover Tokyo
<input type="radio"/> SI0	UTCPLXSB	—	✓ Complete	October 12, 2012 5:14:37 AM Tokyo Standard Time	Octob Tokyo
<input type="radio"/> SJ0	UTCPLXSB	—	⊗ Never Connected	—	
<input type="radio"/> SK0	UTCPLXSB	—	✓ Complete	October 11, 2012 9:02:00 AM Tokyo Standard Time	Octob Tokyo

► Current Training Status Details (Click on training statuses above to view details)

Refresh Last Refresh: Fri Nov 02 2012 13:34:45 GMT+0900 (Tokyo Standard Time)

Figure 5-7 Training Sets panel

There will be a line for every system that has assigned data. To initiate training for a system, select the radio button beside the system and click the **Actions** drop-down. Within the list, select **Request Training**. The **Training Progress** field for that system will change to In Progress. To obtain an update, click **Refresh**. If the training attempt is successful, the **Last Training Result** field will change to Complete and the data and time in the **Current Model Built** field will reflect the current time and date (in local time).

If the training attempt fails, there are a number of things you must consider:

- Did the bulk load job complete successfully? If not, address the problem, rerun the job, perform the Assign again, and then try training the system again.
- Did the input file to the bulk load contain sufficient message data? If the file contained less than 90 days' worth of data, try increasing the number of days in the input file and run the bulk load, Assign, and training again.
- If the training still does not work, check the training period (see "Controlling training options" on page 170). Try increasing the value, disconnecting and reconnecting that system, and training the system again. Note that this option affects every system that is to be trained, so set it back to the previous value after this training attempt.
- If the training attempt still fails, go to the Notifications panel, scroll to the end, and find the AIFTnnnn messages relating to the training attempt. You might see a message similar to that shown in Figure 5-8 on page 170. Refer to *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623, for help in understanding the reason for the failure.

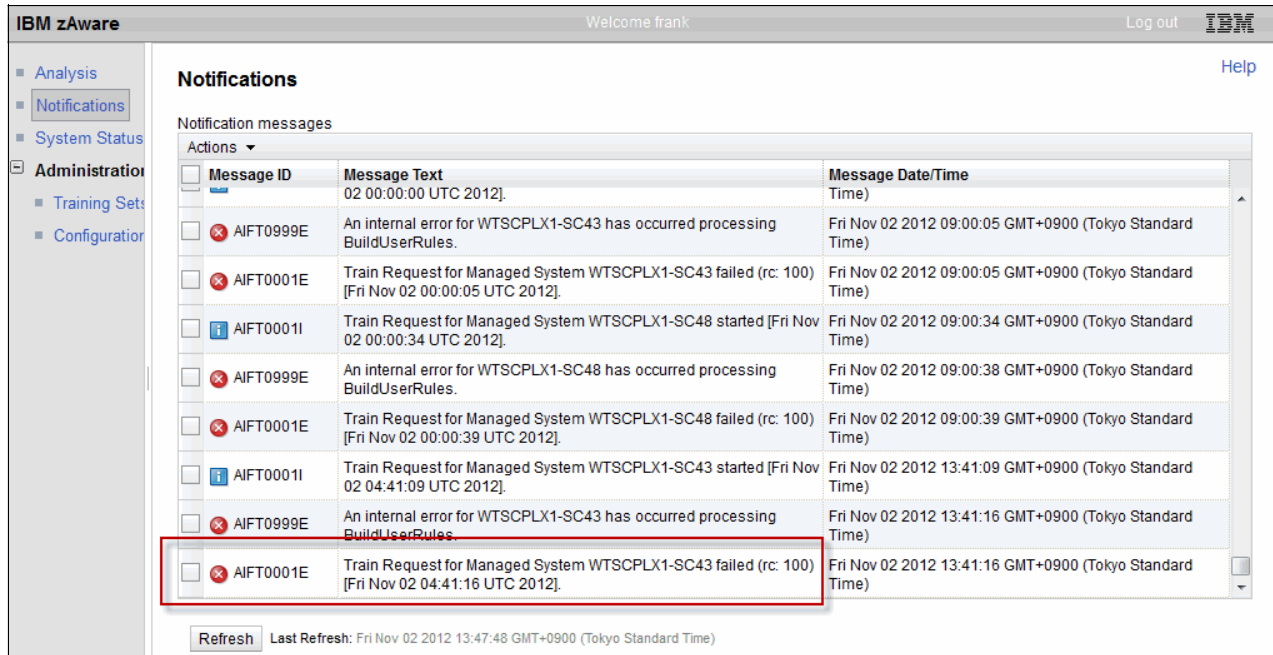


Figure 5-8 Notifications window showing training attempt

- At this point, if the system still does not train successfully, one thing that you must consider is whether this system is a good candidate for monitoring with IBM zAware.

If IBM zAware is unable to find sufficient message diversity and repeating message patterns, the value of the analysis of the message activity on that system will be compromised.

The Message Analysis Program documented in Appendix A, “Syslog Message Analysis Program” on page 207, can be used to identify the number of unique messages in the input file and the average and peak message rates.

As discussed in 3.2, “Selecting which systems to monitor with IBM zAware” on page 96, you could “cheat” and run a program that will generate repeating patterns of message IDs. That might be successful in getting the training to complete successfully. However, you need to consider whether this will just result in inaccurate analysis results from IBM zAware for that system, because the underlying system message activity is still not well suited to IBM zAware.

- If you believe that the input file you provided meets the criteria for successfully creating a model, take a nondisruptive dump of the IBM zAware LPAR as soon as the training attempt fails and send the dump to IBM. The process for doing this is described in 5.11.4, “Sending diagnostic information to IBM” on page 185.

Controlling training options

As discussed previously, IBM zAware has three logical databases:

- Instrumentation data database

This database contains information about the data that was loaded using a bulk load, or transmitted in realtime. The information in this database is the input when you train a system.

Be aware that the full message text for each message is not stored in the database. An analysis is performed on the message data as it arrives, and the required information is extracted and saved in the instrumentation data database.

- **Model database**

This database contains the models of normal message behavior for each monitored client.

- **The analysis results**

The XML files that contain the information that is presented on the Analysis View window are kept in the IBM zAware file system.

You can change parameters for the amount of data that will be used to train a system. The parameters apply to all the systems defined to the IBM zAware application.

The retention period for the information in each of these three databases is controlled from the **Administration** → **Configuration** → **Analytics** page, as shown in Figure 5-9.

The screenshot shows the IBM zAware web interface. The top bar includes the IBM logo, a 'Log out' link, and a 'Welcome frank' message. A left sidebar contains a navigation menu with items: Analysis, Notifications, System Status, Administration (expanded), Training Set, and Configuration. The main content area is titled 'Configure Settings' and has tabs for Analytics, Data Storage, Security, Sysplex Topology, and Priming Data. The 'Analytics' tab is active, displaying five configuration fields, each with a numeric input and a 'days' label: 'Instrumentation data retention time (training period - 730 days):' set to 365; 'Training models retention time (0 - 730 days):' set to 365; 'Analysis results retention time (30 - 3650 days):' set to 365; 'Training period (1 - 365 days):' set to 90; and 'Training Interval (7 - 365 days):' set to 30. At the bottom of the form are 'Apply' and 'Reset' buttons.

Figure 5-9 Configure Settings - Analytics page

In addition to the retention period for the three databases, there are also options to let you control the training period and the training interval. The considerations for these fields are explained here:

Instrumentation data retention time

Set this value to reflect the number of days that you believe are required to build an accurate model of the normal behavior for each system.

Note: The values on this window apply to all monitored systems. If some systems require higher values than other systems, use those higher values.

Training models retention time

This value determines the time period that is used when analyzing how anomalous a message is. If your systems' message behavior is exactly the same every month, it is probably sufficient to keep a few months' worth of data. If periods like year-end or quarter-end are different, set this to be a little over a year.

Analysis results retention time

This controls how far back in time you can go on the Analysis view and see the results for each interval. You need to decide how likely it is that you will want to go back and

look at the message analysis. For example, you might want to look at the analysis for this time last year. Or, possibly, back to the last time you upgraded the operating system or a major subsystem.

Training period

This controls how many days of data, going backwards starting from yesterday, are used when a training activity is performed. Note that there is a relationship between this value and the Instrumentation data retention time value. There is no point in setting the Training period field to a value larger than the Instrumentation data retention time.

The Training period value should be at least as large as the Training interval value.

Changing the Training period value: If you change the Training period value, all systems must disconnect and reconnect to IBM zAware to pick up the new values.

Training interval

This interval determines how often IBM zAware will perform an automatic training for each system.

Give some consideration to the Training interval value. That is, if you decide to exclude message data for “bad” days from the training process, you might want to set the training interval to a larger value so that you do not have to define the excluded days to IBM zAware so frequently (this is discussed in “When to exclude a day from training” on page 173).

Alternatively, consider the case where you install a new product on the day after the last training, and that product produces a new set of messages. IBM zAware will mark those messages as anomalous until you next update the model database. You can initiate a manual training for that system, but if you rely solely on the automatic training to update the model, then you will have a period of time when the results from the IBM zAware analysis might be misleading.

Purge consideration: If you decrease any of the retention period values, the data will not actually be purged from the database until the daily database purge process runs.

Determining the Training period

The Training period you use might need to be changed, depending on the age of the data you use in the bulk load process. The last day of a Training period is always yesterday, and this affects how much of the bulk load data is eligible for inclusion in the model creation.

For example, if the day that you initiate the Training is May 15, and the bulk load data contained 90 days’ worth of data and you want to include all available instrumentation data in the Training, then the Training period value might need to change. The default Training period is 90 days. Because the training period ends with the day before training is requested and looks back the amount of days defined in the Training period, change the Training period

value to 104 (and then disconnect and reconnect that monitored client) before initiating the training. This is shown in Figure 5-10.

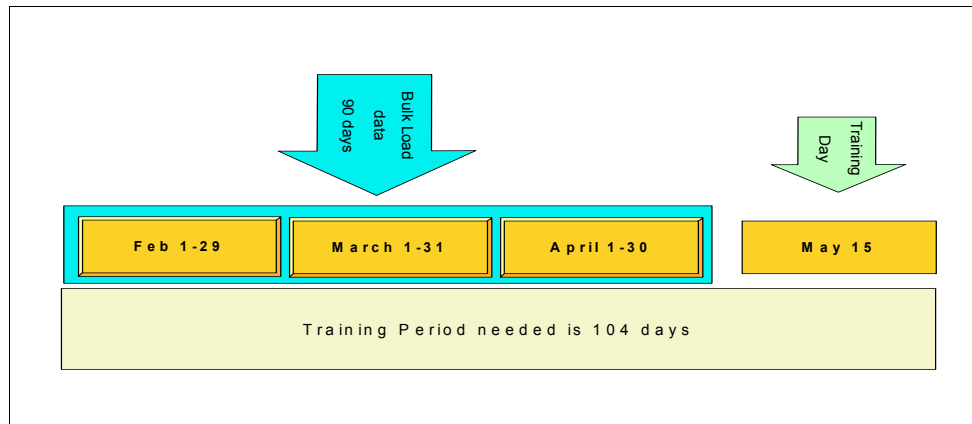


Figure 5-10 Determining the Training period

To include all the data from a bulk load, you must know the first day of the data and subtract that from the current day to get the Training period. Remember that this Training period affects all systems that will be trained from that point.

When to exclude a day from training

IBM zAware learns what a system normally looks like from the message data that you feed to it. If an event occurs that IBM zAware has not seen before, or has not seen frequently, that message will be assigned a high anomaly score. But if a message occurs many times, IBM zAware will think that message is normal for this system.

To avoid having that occur, IBM zAware provides the ability to identify abnormal days for each system, so that message data from those days will be excluded when IBM zAware updates its model of that system.

If you decide to use this capability, navigate to the Training Sets panel. Select the system that you want to exclude some days for and click the **Actions** drop-down. In the drop-down, select **Manage Model Dates**. This presents you with the window shown in Figure 5-11.

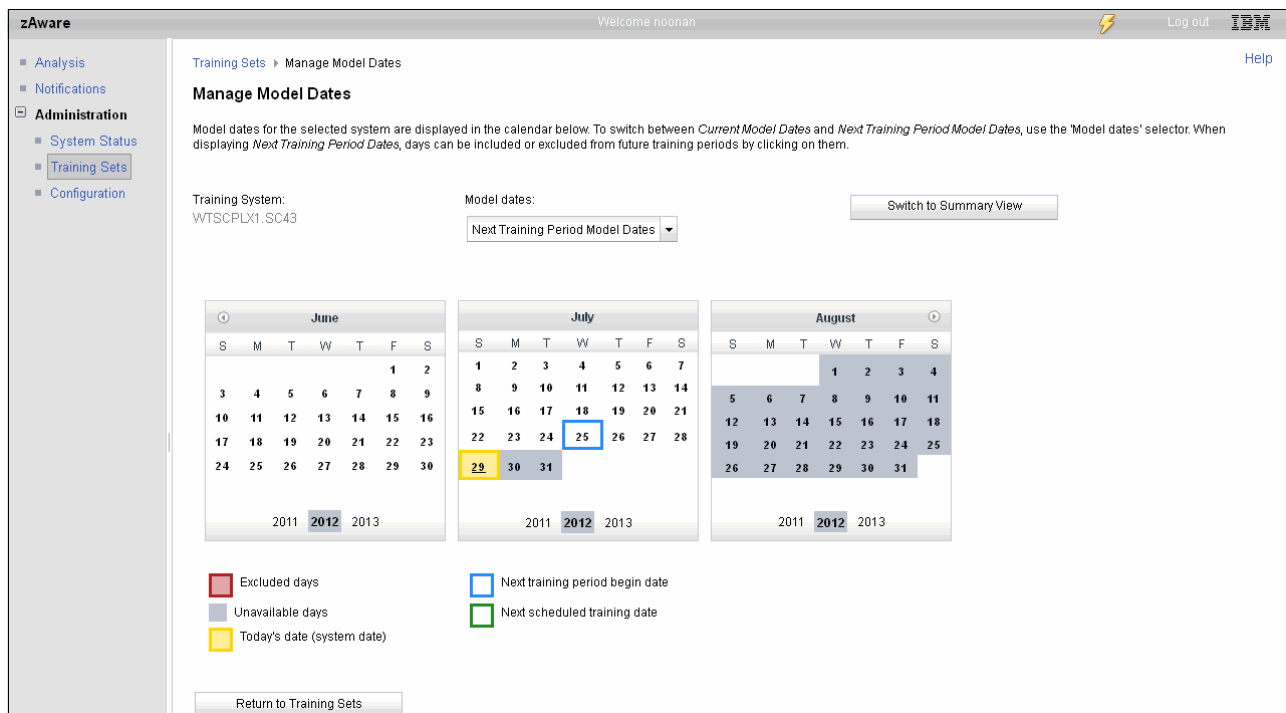


Figure 5-11 Excluding dates from training

On this panel, select the anomalous days that should be excluded from the next training attempt for this system.

One day for one system: The granularity for excluding message data is one day for one system. That is, if the whole sysplex was experiencing a problem on a given day, and you do not want those messages becoming part of the model, you need to go through this process for each system in the sysplex.

It is not possible to exclude a time range less than one calendar day. And there is no mechanism to exclude only certain messages.

5.5.3 Moving systems between sysplexes in IBM zAware

Although it is not something that would occur often, IBM zAware does support the ability to move a system from one sysplex to another. When you move a system, IBM zAware changes the topology of the sysplex information contained in its database.

To move a system from one sysplex to another:

1. Go to the **Administration** → **Configuration** → **Sysplex Topology** window.
2. Select the system that you want to move.
3. Click **Move Selected Systems**.

4. Select the sysplex that you want to move the system to. Note that the target sysplex must have already connected to the IBM zAware LPAR.
5. Click **OK**.

After the move is successful, when you view the Analysis panel and look at the sysplex you moved the system to, the system will now appear grouped under that sysplex.

5.5.4 Data management granularity

The retention period controls that IBM zAware provides apply to all monitored clients. When the retention period is reached, any data older than that for all clients will be purged.

There is no ability to have different retention periods for different monitored clients. There is also no ability to delete data for a monitored client before the retention period expires. So, for example, if you use a test system to perform initial testing of IBM zAware, and then decide to only connect production and QA systems to IBM zAware, you cannot delete the data for the test system until it times out based on the retention values.

Similarly, IBM zAware does not provide an ability to export the information for a monitored client. You can certainly move a monitored client from one IBM zAware to another. However in the new IBM zAware, you need to perform a bulk load and start fresh, just as you did when you connected that system to the first IBM zAware.

5.6 Managing IBM zAware firmware

It is to be expected that fixes and enhancements will be provided for IBM zAware over time. You can manage IBM zAware firmware similar to the way you manage your CFCC code.

Determining current IBM zAware firmware level

If you have a problem with IBM zAware, or simply want to know the IBM zAware firmware level that you are currently using, you can retrieve this information from the Hardware Management Console (HMC).

Select the System Information option under Change Management, as shown in Figure 5-12.

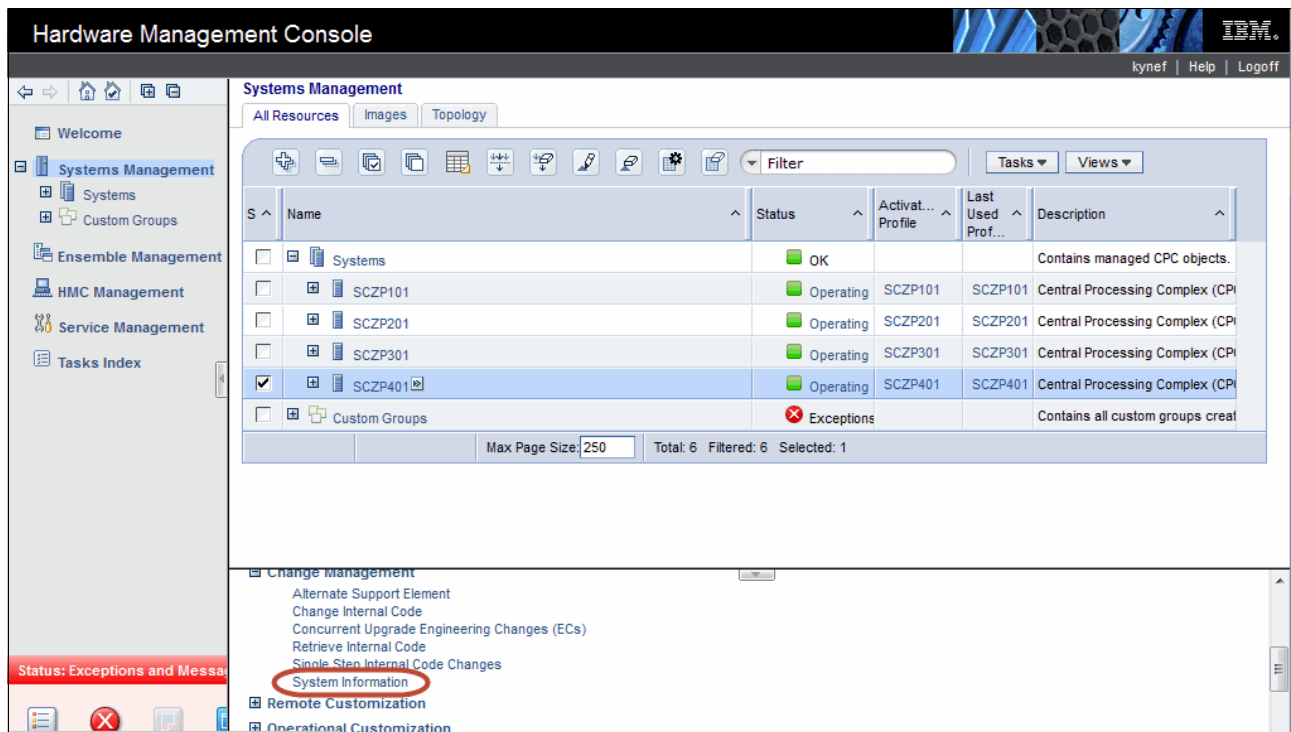


Figure 5-12 Selecting System Information option

If the zAware feature has been ordered and installed on the zEC12, the Microcode Control Level (MCL) for the zAware firmware can be displayed on the resulting window, as shown in Figure 5-13.

In this example, the activated MCL level of the IBM zAware code is shown to be H09126 - 012.

System Information - SCZP401

Machine Information

EC number:H09173

LIC control level:0001

Engineering Changes AROM

Type: 2827

Model number: H43

Serial number:00002000B8D7

Version: 2.12.0

Bundle level: 11z+

Internal Code Change Information

Select	EC Number	Retrieved Level	Installable Concurrent	Activated Level	Accepted Level	Description
<input type="radio"/>	H09173	089	089	089	041	SE Framework
<input type="radio"/>	H09125	003	003	003		zFlash
<input type="radio"/>	H09126	012	012	012	006	zAware
<input type="radio"/>	H09127	001	001	001		OSA Express4S 1000 base T
<input type="radio"/>	H09128	003	003	003	003	Firmware Partition Base
<input type="radio"/>	H09129	012	012	012	007	Express4S Crypto
<input type="radio"/>	H09130	001	001	001		OSA Express4S ICC
<input type="radio"/>	H09131	002	002	002		OSA Express4S Intra-Ensemble Management
<input type="radio"/>	H09132	003	003	003	001	OSA Express4S Intra-Ensemble Data
<input type="radio"/>	H09133	003	003	003	001	OSA Express4S OSD
<input type="radio"/>	H09134					OFCP Express8S LIC
<input type="radio"/>	H09135	003	003	003		Ficon Express8S LIC

EC Details...

Pending Actions

There may be some pending actions. Click "Query Additional Actions..." for more information.

Query Additional Actions...

OK

Help

Figure 5-13 Displaying IBM zAware firmware MCL level

Applying IBM zAware firmware updates

IBM zAware firmware updates (distributed by IBM as Microcode Control Levels or MCLs) can be retrieved onto the CPC Support Element non-disruptively. However, *activating* the updates will result in IBM zAware automatically reIPLing itself.

The process of activating new service on a CPC can vary from a few minutes to over an hour, and it is not possible to predict the exact point during that process when the IBM zAware MCLs will be activated. Your IBM Customer Engineers will tell you when they start the activate, and the point at which it has completed. The reIPL of IBM zAware, if it is needed, will occur during that interval.

When IBM zAware performs the IPL, the connections from all monitored systems will be reset. This results in messages similar to those shown in Example 5-2 on each monitored system.

Example 5-2 System Logger messages when IBM zAware reIPLs itself

```

IXG384I ZAI LOGSTREAM CLIENT ERROR OCCURRED 700
FOR LOGSTREAM SYSPLEX.OPERLOG
REASON: OMVS BPX-SERVICE ERROR - LOGGER WILL RETRY THE REQUEST,
DIAG=00000000

```

When the connection is reset in this manner, System Logger attempts to restart the connection for up to 20 minutes. If IBM zAware does not complete its reinitialization within this time, it will issue an IXG386I message and stop attempting to restart the connection.

Your z/OS automation product should be monitoring for that message and raise an alert that the operator should investigate the problem. Resolution might be as simple as issuing a

SETLOGR FORCE, ZAICONNECT, LSN=SYSPLEX.OPERLOG command. If that is not successful, use the System Logger IXG3xx messages to determine the cause of the problem.

Moving to a new IBM zAware release

It is important to remember that IBM zAware is unlikely to be considered a mission-critical application. We expect that most customers will not have a test IBM zAware LPAR. Furthermore, because IBM zAware automatically reIPLs whenever it detects that an MCL or upgrade has been applied, it is not possible to run more than one level of IBM zAware concurrently on a CPC. Refer to 3.2, “Selecting which systems to monitor with IBM zAware” on page 96 for more considerations about the need for test IBM zAware LPARs.

When IBM zAware MCLs or new releases are installed, perform the normal testing to ensure that it is performing as expected. If it is not, evaluate whether the problem is significant enough to warrant backing out the new code. Keep in mind, however, that backing out the IBM zAware changes is likely to result in also backing out service to other CPC components.

5.7 Disaster recovery considerations

In addition to the license for a normal production IBM zAware, you can also license IBM zAware on a CPC that will be used in case of a disaster.

IBM zAware can be beneficial in detecting unusual messages during a disaster recovery (DR) test. One of the top priorities in a DR test is to ensure that everything is working the same in the disaster recovery site as it normally works. By spotting anomalous messages, IBM zAware can be a powerful part of your DR test tool kit.

In the case of a DR test or a real disaster, you would set up your IBM zAware LPAR and its disks and then perform bulk loads from the monitored systems that are part of your DR plan. In fact, the steps to get IBM zAware up and running are identical to when you initially installed and implemented it in your production environment.

At the time of writing, you cannot use mirrored IBM zAware disks in your DR site. In 5.3.4, “Backing up the IBM zAware file system” on page 161, we discuss the relationship between IBM zAware information that is stored on the CPC Support Element and its file system. Although it is possible to mirror the IBM zAware disks, when you add them to the IBM zAware in the DR site, it will view them as new disks and format them, thereby removing the data on them.

5.8 Daily and weekly management tasks

As you become more familiar with IBM zAware, you will probably develop your own list of management activities. This section provides suggestions that you can use to get your list started. You can add to and fine-tune these lists based on your own experiences.

5.8.1 Daily management tasks

As stated in 5.1, “Managing IBM zAware components” on page 154, IBM zAware is designed to be largely self-maintaining. However, there are a small number of daily tasks that should be carried out in relation to IBM zAware:

- ▶ Using the Analysis view, check for any bars that contain an unusually high number of message IDs or that are dark blue, yellow, or orange. These are all indicators of anomalous message behavior and should be investigated.
- ▶ While in the Analysis view, check that there is a bar for every monitored client for every interval. If any clients have intervals with no bars, the likely cause is that the client was IPLed/ However, check to ensure that this *is* the cause of any missing analysis data.
- ▶ Use the System Status view to verify that all monitored clients have a Status of Active. Have automation in place on each monitored client to monitor the status of the connections. In addition, it is also prudent to take the time to ensure that the IBM zAware view of the status of each connection is consistent with the z/OS view.
- ▶ Check the Notifications window for messages. It is most likely that any messages will be related to training attempts. If any messages indicate a failed training attempt or other problem with IBM zAware, follow the guidance for the message as documented in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

After you have addressed the messages, delete them.

5.8.2 Weekly management tasks

There is also a small number of tasks to perform less frequently.

- ▶ Using the Data Storage tab in the Configure Settings window, check the Total Storage usage (%). If it exceeds a threshold (perhaps 80%), add more volumes to the IBM zAware configuration. And if you are backing up the IBM zAware volumes, remember to add the new volumes to your backup jobs.

If the IBM zAware disks become completely full, attempts to send more log data to that LPAR will fail. The monitored clients will present the following messages:

```
IXG372I ZAI LOGSTREAM CLIENT MANAGER ERROR
FOR LOGSTREAM SYSPLEX.OPERLOG
FUNCTION=BPX4AIO  ERRNO=0000008C  ERRNOJR=76697242
```

Note that this message is presented any time System Logger finds that IBM zAware will not accept message traffic, so this message does not necessarily mean that the IBM zAware disks are full. However, if the disks *are* full, you would see this message on all connected clients.

- ▶ If you are backing up the IBM zAware disks, check the backup job to ensure it is completing successfully.
- ▶ If you decide to follow a strategy of excluding abnormal days from training, use the Manage Model Dates function on the Training Sets window to provide IBM zAware with the list of dates that you want to exclude.

5.9 Checklist for adding a monitored client

As you become more familiar with IBM zAware, you will probably want to add more monitored clients. The process for this will vary from one enterprise to another, depending on your

specific requirements. However, the following checklist can be helpful as a base to get you started with your own checklist:

- ▶ Ensure that the system that you plan to add has at least 90 days of archived syslog or OPERLOG.
- ▶ Ensure that the system has all necessary IBM zAware-related service applied.
- ▶ If necessary, add the system IP address to the firewall that is protecting the IBM zAware LPAR.
- ▶ Check that the IBM zAware file system has sufficient spare space for the additional data.
- ▶ Verify that the IBM zAware LPAR has enough memory (see “Memory” on page 101 for more information about memory requirements for IBM zAware LPARs).
- ▶ Update the z/OS system’s TCP/IP hosts file, if necessary.
- ▶ Ensure that the z/OS LPAR is able to see the IBM zAware IP address.
- ▶ Ensure that all required security accesses have been defined (see 3.4.4, “Security” on page 109 for more information).
- ▶ Modify the OPERLOG log stream attributes to add the ZAI keywords if this has not already been done.
- ▶ Update the IXGCNFxx member to point at the IBM zAware LPAR, and activate that member using the **SET IXGCNF=xx** command.

Remember to update the IEASYSxx member to explicitly point at the IXGCNFxx member.

- ▶ Extract 90 days’ worth of syslog or OPERLOG into the file that you will input to the Bulk Data Load Utility job.
- ▶ Issue the **D LOGGER,STATUS,ZAI,VERIFY** command to ensure that System Logger can connect to IBM zAware.
- ▶ Issue the **SETLOGR FORCE,ZAICONNECT,LSN=SYSPLEX.OPERLOG** command to start the connection between System Logger and IBM zAware.
- ▶ Submit the Bulk Data Load Utility job and ensure that it completes successfully.
- ▶ Use the **D LOGGER,C,LSN=SYSPLEX.OPERLOG,D** command to monitor the amount of data that is queued in the buffers, waiting to be sent to IBM zAware.
- ▶ When all data has been sent, logon to the IBM zAware GUI, navigate to the Priming Data tab, and assign the data that you just bulk loaded to the correct sysplex.
- ▶ After a few minutes, check the System Status window to verify that all systems have connected again after the Assign. In particular, ensure that the new system that you are adding has reconnected.
- ▶ Navigate to the Training Sets window and initiate training for your new system.
- ▶ Update the automation on the new monitored client to monitor the status of the connections to the IBM zAware LPAR.

5.10 System Logger commands for IBM zAware support

The PTF to enable the System Logger support for IBM zAware added a number of new commands to System Logger. These commands can be used to stop and start the connection to IBM zAware, and to check the status of the connection. To be able to quickly and successfully manage your IBM zAware configuration and address any problems, become familiar with these commands.

The first command is the most basic one. It updates the information that System Logger gets from the IXGCNF Parmlib member. If you update that member, or are adding the IBM zAware information for the first time, you can activate it with the following command:

```
SET IXGCNF=xx
```

Be aware that if System Logger currently has an active connection to an IBM zAware LPAR, or is trying to activate such a connection, and the new IXGCNF member points to a different host name or IP address for the IBM zAware server, the attempt to activate the new member will be rejected. If you want to change the IP address or host name value in the IXGCNFxx member, you must first quiesce the connection to IBM zAware.

You can display the current IXGCNF settings using the following command:

```
D LOGGER,IXGCNF
```

Note that this will return information about aspects of System Logger other than simply the IBM zAware information.

To obtain somewhat more information about the IBM zAware support in System Logger, issue the following command:

```
D LOGGER,STATUS,ZAI
```

This command returns some of the same information as the **D LOGGER,IXGCNF** command. However, it also provides information about whether any log streams have the ZAI attribute, and whether those log streams are connected to this system.

If this system is a monitored client, the output from that command should contain **ZAI LOGSTREAM CLIENTS: ACTIVE**. The output also provides information about buffer use if messages are queued waiting to be sent to IBM zAware. It is most likely that you will see non-zero buffer use numbers when you are running a bulk load job or if the connection to IBM zAware is temporarily unavailable.

To confirm that this system is able to successfully connect to the IBM zAware LPAR, use the following command:

```
D LOGGER,STATUS,ZAI,VERIFY
```

This will prompt Logger to initiate the connection to IBM zAware. If there is already an active connection, Logger will verify that the connection is operational. If Logger is unable to start the connection, refer to 5.11, “Problem determination for IBM zAware” on page 183 for help with determining the cause of the problem.

To verify that System Logger is currently connected to IBM zAware and confirm that messages are being successfully transmitted, use the following command:

```
D LOGGER,C,LSN=SYSPLEX.OPERLOG,D
```

The output from this command will include, among other things, the status of the ZAI client. For a monitored client, you normally expect this to show a status of **CONNECTED**. It also provides a count of the number of blocks that were sent to IBM zAware successfully, and the number of blocks that System Logger was unable to transmit.

Normally, System Logger will automatically start the connection to IBM zAware after an IPL, assuming that you provided an IXGCNFxx member with the required information. However, if you need to manually start the connection, you can do so using the following command:

```
SETLOGR FORCE,ZAICONNECT,LSN=SYSPLEX.OPERLOG
```

Note this command can be issued even if the connection to IBM zAware is already active. Additionally, this command does *not* cause System Logger to purge any messages from its buffers.

Finally, to stop the connection to IBM zAware for some reason, issue the following command:

```
SETLOGR FORCE,ZAIQUIESCE,LSN=SYSPLEX.OPERLOG
```

This command will cause the connection to go inactive. It will also purge any messages that are queued in the System Logger buffers, waiting to be sent to IBM zAware.

There is currently no command to cause System Logger to close the connection to IBM zAware, but to continue accepting messages into its buffers for eventual transmission to IBM zAware. However, shutting down the IBM zAware LPAR appears as a temporary connection error to System Logger, and it will continue to buffer messages to be sent over to IBM zAware. For this reason, there will probably be few cases where you would want to use this command. If you want to permanently stop a monitored client from sending data to IBM zAware, update the IXGCNFxx member used by that system to specify a SERVER value of NONE.

5.10.1 IBM zAware messages

The IBM zAware notifications window shown in Figure 5-14 provides information about various IBM zAware activities, particularly attempts to performing training for the monitored systems. The messages are documented in Appendix D of *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623. If you encounter a message that is not included in your level of that document, retrieve the most recent update from ResourceLink.

Message ID	Message Text	Message Date/Time
AIFT0001I	Train Request for Managed System WTSCPLX1-SC66 started [Sat Oct 20 19:37:39 UTC 2012].	Sun Oct 21 2012 04:37:39 GMT+0900 (Tokyo Standard Time)
AIFT0999E	An internal error for WTSCPLX1-SC66 has occurred processing BuildUserRules.	Sun Oct 21 2012 04:37:49 GMT+0900 (Tokyo Standard Time)
AIFT0001E	Train Request for Managed System WTSCPLX1-SC66 failed (rc: 100) [Sat Oct 20 19:37:49 UTC 2012].	Sun Oct 21 2012 04:37:49 GMT+0900 (Tokyo Standard Time)
AIFT0001I	Train Request for Managed System WTSCPLX1-SC42 started [Sat Oct 20 20:17:09 UTC 2012].	Sun Oct 21 2012 05:17:09 GMT+0900 (Tokyo Standard Time)
AIFT0999E	An internal error for WTSCPLX1-SC42 has occurred processing BuildUserRules.	Sun Oct 21 2012 05:17:15 GMT+0900 (Tokyo Standard Time)
AIFT0001E	Train Request for Managed System WTSCPLX1-SC42 failed (rc: 100) [Sat Oct 20 20:17:15 UTC 2012].	Sun Oct 21 2012 05:17:15 GMT+0900 (Tokyo Standard Time)
AIFT0001I	Train Request for Managed System UTCPLXSA-SC0 started [Sat Oct 20 22:47:41 UTC 2012].	Sun Oct 21 2012 07:47:41 GMT+0900 (Tokyo Standard Time)
AIFT0999E	An internal error for UTCPLXSA-SC0 has occurred processing BuildUserRules.	Sun Oct 21 2012 07:47:46 GMT+0900 (Tokyo Standard Time)
AIFT0001E	Train Request for Managed System UTCPLXSA-SC0 failed (rc: 100) [Sat Oct 20 22:47:46 UTC 2012].	Sun Oct 21 2012 07:47:46 GMT+0900 (Tokyo Standard Time)

Refresh Last Refresh: Thu Nov 01 2012 13:21:57 GMT+0900 (Tokyo Standard Time)

Figure 5-14 IBM zAware notifications window

You can see in the figure that some messages are informational (the ones ending in I), and some are error messages (the ones ending in E). For error messages that require action by IBM, IBM zAware attempts to gather the required documentation and automatically send it to IBM for analysis. The “response” section of the documentation for each message will indicate

whether a user action is required, or if IBM zAware will automatically open an incident and send supporting documentation to IBM.

You can use the Hardware Messages for the CPC that the IBM zAware LPAR resides on to determine whether any calls to IBM were made by the CPC, as shown in Figure 5-15.

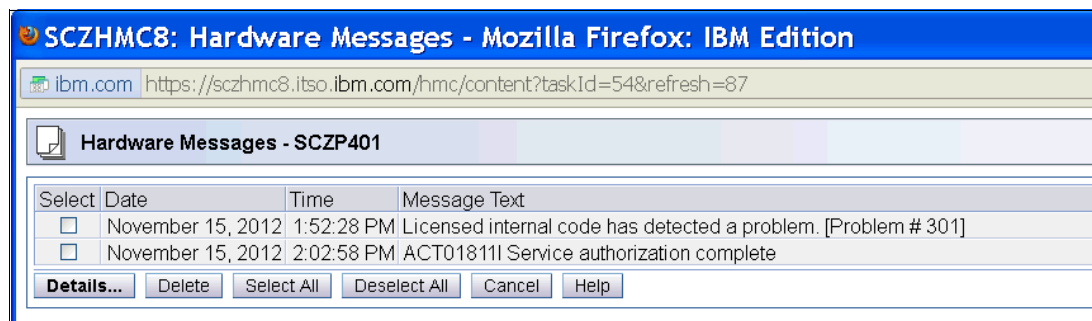


Figure 5-15 HMC Hardware Messages

5.11 Problem determination for IBM zAware

IBM zAware is designed to be a “closed application”. This means that IBM zAware does not have a console where you can issue commands or look at messages. For people who are used to working with z/OS, this is a different paradigm, and you might wonder how you can investigate perceived problems with IBM zAware or your configuration. There are facilities to help you perform some problem determination.

5.11.1 Problems during IBM zAware initial setup

The initial setup of IBM zAware is straightforward, with few things that can go wrong. In our case, the only challenges we encountered were related to connecting to the IBM zAware LPAR, either from a web browser or from our z/OS systems.

When you perform an Activate against the IBM zAware LPAR, the Activate will probably run for longer than you might be used to with z/OS. In our case, the activation took about 3 minutes. At that point, the LPAR status on the HMC changed to `Operating`. We took this to mean that IBM zAware was ready for use and immediately tried logging on.

However, after the LPAR completes initializing, the application server still has additional work to do before it is ready to accept browser sessions. Wait a few minutes after you see the `Operating` status before you attempt to start a session with IBM zAware.

If the IBM zAware initialization fails for some reason, the status will be reflected on the HMC. If you are unable to determine the reason for the failure, open an incident with IBM. See 5.11.4, “Sending diagnostic information to IBM” on page 185, for information about reporting IBM zAware problems to IBM.

5.11.2 Confirming connectivity to IBM zAware

A few minutes after the status of the IBM zAware LPAR changes to `Operating`, it should be possible to logon to the IBM zAware GUI. Point your browser at the URL that represents the IBM zAware LPAR, remembering to include “/zAware” after the IP address or domain name. Note that the “/zAware” is case sensitive.

If the browser is unable to connect to IBM zAware, try pinging the IBM zAware LPAR. If the ping is unsuccessful, issue a **TRACERT** command (if using Windows), or **tracert** (if using Linux) and bring the output to your network team.

To verify that you can connect to IBM zAware from System Logger, issue the **D LOGGER,STATUS,ZAI,VERIFY** command. If the command fails, issue a **PING** command from that z/OS system to ensure that it has network connectivity to the IBM zAware LPAR. If the PING is successful, check the return and reason codes from the IXG387I message.

Assuming that the required network connectivity is in place (and that there are no firewall issues), two possible reasons for System Logger being unable connect to IBM zAware are:

- ▶ No disks have yet been added to the IBM zAware LPAR. Because there is nowhere to store the messages that would be sent over from System Logger, IBM zAware will not allow the connection to be started. In this case, logon to the IBM zAware GUI and add some disks.
- ▶ The analytics engine on IBM zAware has been stopped. If the engine is stopped, System Logger will not be able to connect successfully and you will observe messages similar to those shown in Example 5-3.

Example 5-3 Attempting to start connection when analytics engine is stopped

```
IXG372I ZAI LOGSTREAM CLIENT MANAGER ERROR 396
FOR LOGSTREAM SYSPLEX.OPERLOG
FUNCTION=BPX1CON  ERRNO=00000468  ERRNOJR=76630291
```

5.11.3 Checking for problem notifications on the IBM zAware GUI

Certain events can cause a notification to be generated on the IBM zAware GUI. If there are notifications that have not been removed yet, there will be a small lightning symbol near the top right of the GUI and the **Notifications** window will show the messages, as shown in Figure 5-16.

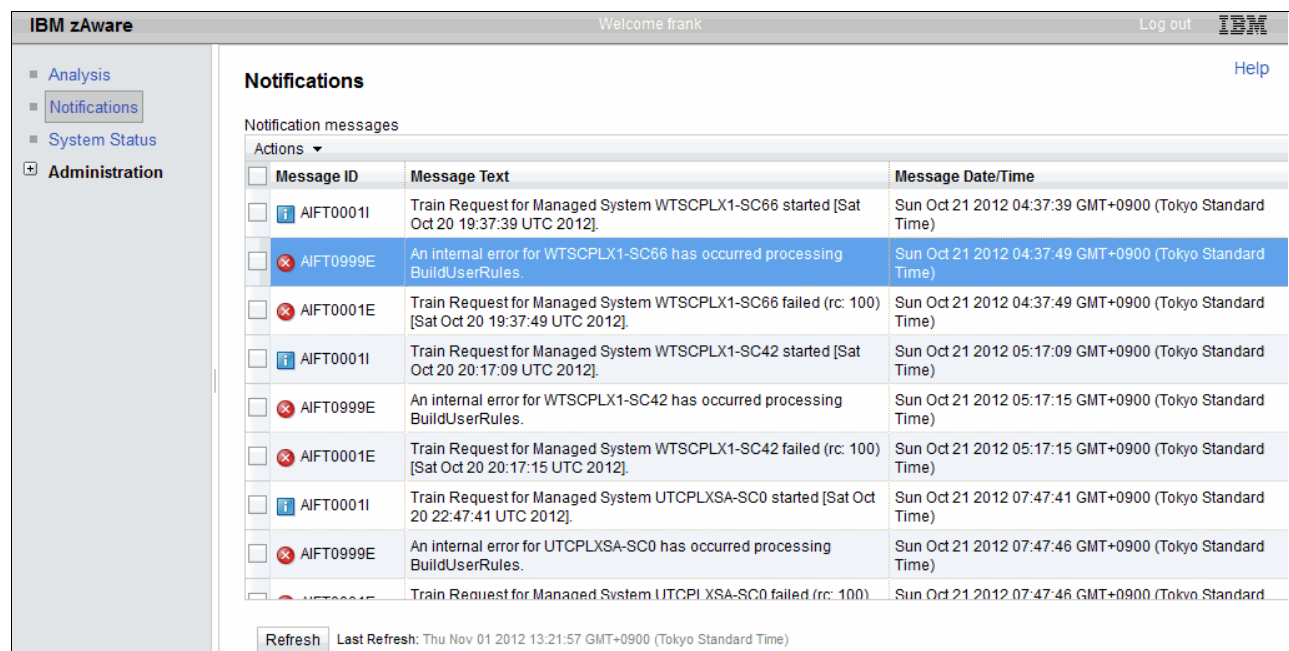


Figure 5-16 IBM zAware Notifications window

Refer to Appendix D of *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623, for information about the error and the recommended action. If the message is not yet documented in that book, open an incident with IBM.

5.11.4 Sending diagnostic information to IBM

IBM zAware is designed to automatically gather diagnostic information and Call Home to IBM when certain problems are encountered. Assuming that Call Home is enabled on your CPC, this should result in the IBM support center contacting you to obtain more information about the event. To determine whether a Call Home was generated for a given call, use the HMC to check whether there are hardware messages outstanding against the CPC that IBM zAware runs on. The hardware message will identify the IBM zAware LPAR name and the date and time that the problem was encountered and reported to IBM.

If the problem did not result in a Call Home, or the problem is such that IBM zAware is not aware of the problem, take a dump of the IBM zAware LPAR and submit it to IBM. Follow these steps:

- Take an LPAR dump of the IBM zAware LPAR.

Logon to the Support Element of the CPC where IBM zAware is running. Expand the list of partitions and select the IBM zAware LPAR. Then expand the **Service** option in the bottom half pane of the window, as shown in Figure 5-17.

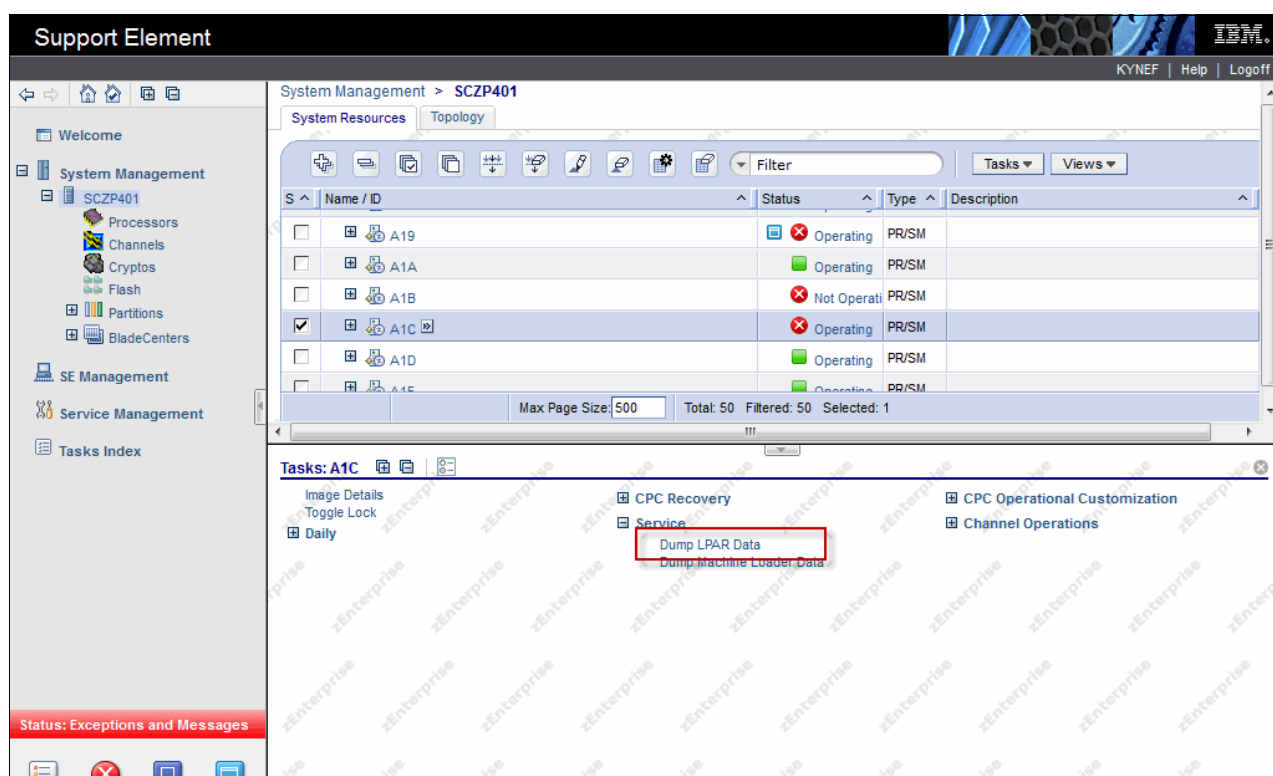


Figure 5-17 Navigating to LPAR dump option

Click **Dump LPAR Data**. You will be presented with the window shown in Figure 5-18.

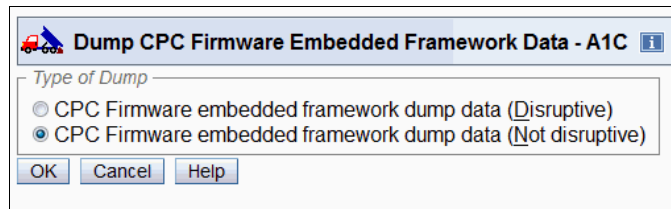


Figure 5-18 LPAR Dump options

Ensure that the **Not disruptive** option is selected and click **OK**. The dump might take some time. When the dump completes, click **OK**.

- Deselect the IBM zAware LPAR and select the CPC instead. The bottom pane will now change and look similar to that shown in Figure 5-19.

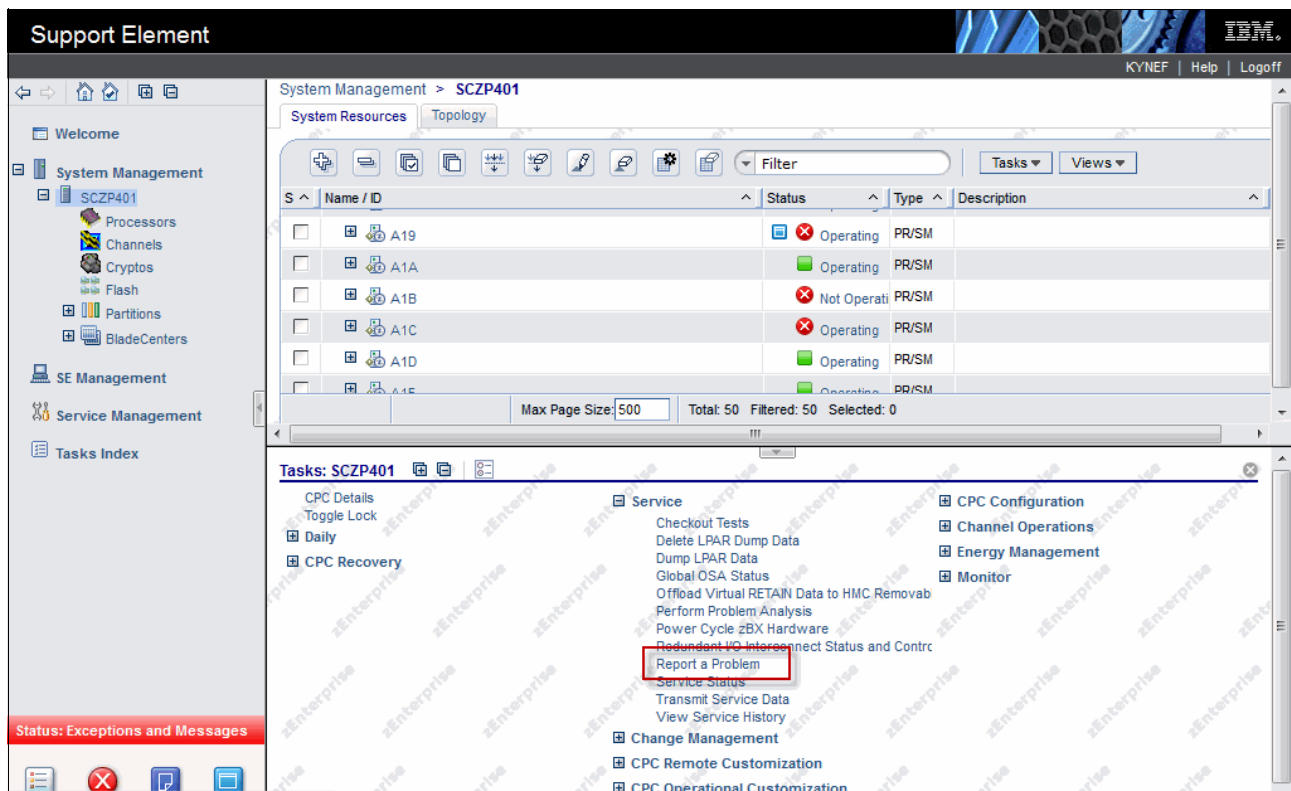


Figure 5-19 Opening a PMV

- Click **Report a Problem** in the Service drop-down. This will display the selection shown in Figure 5-20 on page 187.

Report a Problem - SCZP401

To report a problem, select a problem type then enter the problem description.

Problem Type

- ☐ Power
- ☐ CPC
- ☐ LAN
- ☐ Software
- ☐ I/O
- ☐ zBX
- ☒ **Type V Viewable PMH(PMV)**
- ☐ Other
- ☐ Test automatic problem reporting

Problem Description

This is just a test

Request Service **Cancel** **Help**

Figure 5-20 Reporting a PMV

- Select the **Type V Viewable PMH(PMV)** option. Enter descriptive text in the Problem Description box. Then click **Request Service**.

You will be presented with a window requesting contact information. Complete the information and click **Request Service**.

- Depending on the RSF profile settings for the system, you might be required to authorize the service request through the hardware messages window. If you are, go to the Hardware Messages icon. There should be an entry with a Message Text that says Problem reported by customer. Select that entry and click **Details**. View the details to ensure this is the PMV that you opened, and then click **Request Service** as shown in Figure 5-21.

Problem Analysis - SCZP401

System name: SCZP401
 Date: Nov 2, 2012
 Time: 5:40:13 AM

Problem Description

A TYPE V problem was reported using the Report a Problem task on the hardware system console. Authorization for this problem is required.

Corrective Actions

Your service call will be placed automatically using the remote support facility (RSF). When the service call completes, a Hardware Management Console message indicating the status of the call will be displayed.

Request Service... **No Service** **Display Sense Data** **Delete** **Cancel** **Help**

Figure 5-21 Authorizing service request

- You are presented with a window confirming your contact information. Click **Request Service**.
- Allow a little time for the PMV to be processed in IBM. Then go back to the Support Element window. In the Service drop-down, click **View Service History** as shown in Figure 5-22 on page 188.

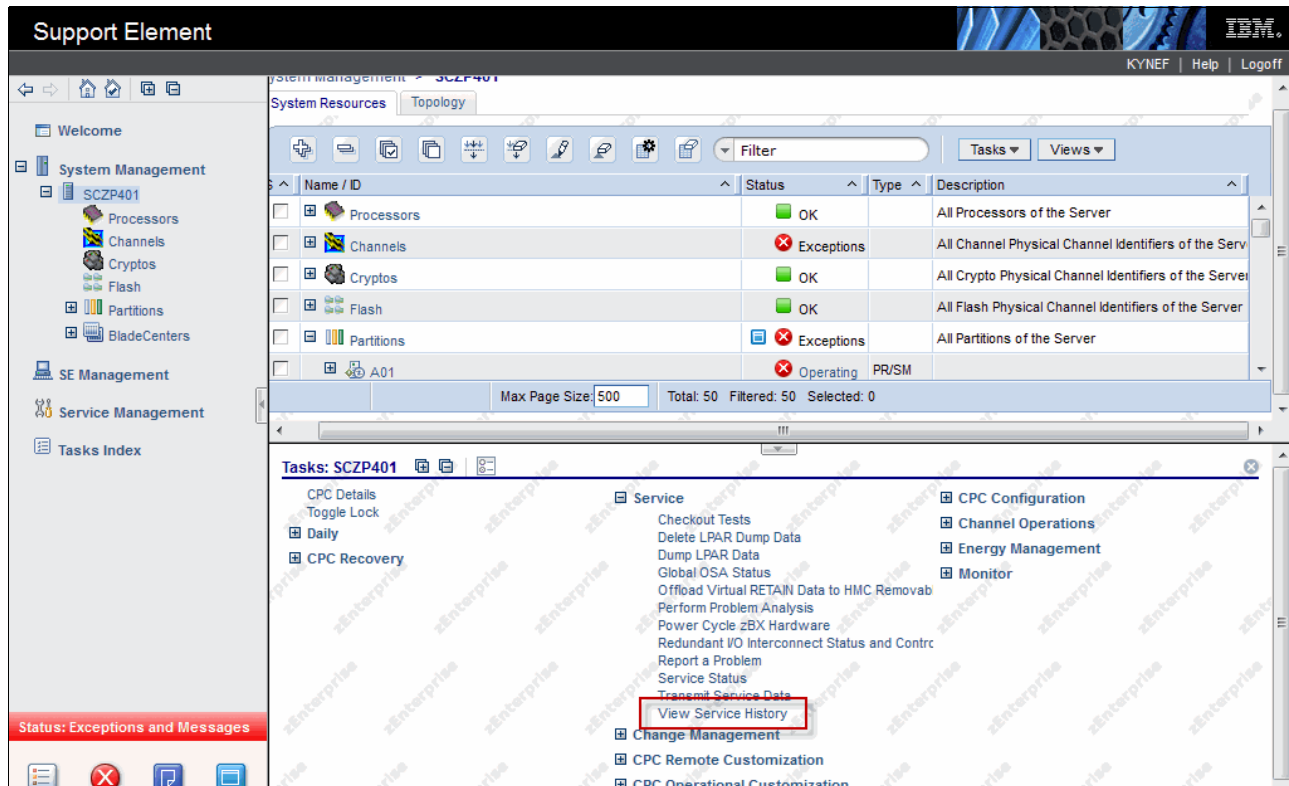


Figure 5-22 Retrieving the PMV number

- ▶ You will be presented with a window showing the service history for the CPC. Your call should be at the top of the list. Select your call and click the **View** drop-down. Then click **Problem Summary**. You will be presented with a window similar to that shown in Figure 5-23. Record the PMH number and click **OK**.

Service History - SCZP401

System name: SCZP401
Machine type: 2827
Machine model: H43
Machine serial number: 00002000B8D7
Problem management hardware (PMH) number: 79940
Problem number: 279
Problem type: V
Problem data: No parts identified

Date	Time	Problem State
Nov 2, 2012	5:40:15 AM	Problem detected
Nov 2, 2012	5:40:15 AM	Customer notified

OK
Help

Figure 5-23 Viewing the Problem Summary

- ▶ This will bring you back to the Service History window. Close that window.
- ▶ In the Support Element window, select **Transmit Service Data** under the Service list. You will be presented with a window similar to that shown in Figure 5-24 on page 189.

Transmit Service Data to IBM - SCZP401

Select the data you want and the destination for the data. Enter the related problem management hardware number if known.

Service Data Destination

☒ IBM Service Support System

Service Data Selections

- ☐ Support element trace
- ☐ System availability data
- ☐ Support element log
- ☐ Support element log - truncated
- ☐ Support element backup log
- ☐ Problem determination data
- ☐ Change internal code trace
- ☐ Installation completion report
- ☐ Task recording data
- ☐ Print screen files
- ☐ Component logs
- ☐ Audit log
- ☐ Check stop dump data
- ☐ Concurrent Engineering Changes Upgrade Data
- ☐ System activity and activation profiles
- ☐ LICCC data files
- ☐ LPAR zone group files
- ☐ Logical partition dump data
- ☐ CPC dump data
- ☐ Coupling facility logical partition dump data
- ☐ Coupling facility diagnostic dump data
- ☐ CPC hardware configuration data
- ☐ Alternate SE problem data
- ☐ Temporary upgrade billing data
- ☐ Power logs
- ☐ Hydra service data
- ☐ Service trace data
- ☐ zBx switch data
- ☐ zVM Management Guest dump data
- ☒ CPC firmware embedded framework dump data

Problem Management Hardware Number

PMH number (optional)

IOCDs Files

Number of IOCDs files selected:

Virtual RETAIN Files

Virtual retain files for problem number:

Number of files selected:

zBX Resources

Number of zBX resources selected:

Performance Management Diagnostic Data

☐ Performance management data

Collection:

Figure 5-24 Transmitting service data to IBM

- ▶ Select the **CPC firmware embedded framework dump data** option. The right side of the window will change, and you will be presented with a place to enter the PMH number. Click **Send**.
- ▶ You will be presented with a window confirming that the information will be sent to IBM. Click **OK**.
- ▶ Remember to log off the Support Element when you are finished.



Integrating IBM zAware with other IBM products

This chapter discusses the options for integrating IBM zAware with other IBM products. To obtain the maximum value from IBM zAware, it is important to integrate it into your systems management environment. IBM zAware provides an application programming interface (API) that you can use to provide that integration. This chapter provides an example of the use of that API.

6.1 Introduction

The IBM zAware application provides a graphical user interface that is accessed using a web browser. This is useful for enabling you to periodically look at and review its results. However, you also need a way to monitor IBM zAware without having to constantly watch yet another console. Therefore, you will probably want to integrate IBM zAware with your other systems management products.

By linking the information from IBM zAware with other products, the data can be made more useful and queried close to real time, for analysis with your automation programs.

This chapter describes two types of integration that currently are possible with the IBM zAware application. The first type of integration is a direct URL connection from the z/OSMF user interface to IBM zAware. This lets you quickly open the IBM zAware graphical interface while you are reviewing the incident logs in z/OSMF, for example.

The second type of integration is with IBM Tivoli NetView for z/OS. NetView provides samples that can be used to interface between user-written programs and the IBM zAware API. This integration gives you the ability to programmatically retrieve the same information that is presented in the Analysis view in the IBM zAware graphical interface.

The information retrieved into NetView can be used in various ways. You can use the information to verify that the IBM zAware application is still recording information from your system. You can also pass the information to your automation to have it raise an alert if the anomaly score for an interval exceeds a certain threshold. And you can use the CANZLOG consolidated log feature of NetView to perform problem determination to determine the cause of the anomalies that IBM zAware highlighted.

It is expected that eventually other IBM products will be extended to integrate information from IBM zAware.

6.2 z/OSMF

In z/OSMF, you can link to other web pages from the z/OSMF navigation tree. This allows you to easily launch IBM zAware from within z/OSMF, and keeps your desktop organized so that problem determination tools are grouped together.

Figure 6-1 shows the z/OSMF home window after you have logged onto z/OSMF using a z/OSMF administrator user ID.

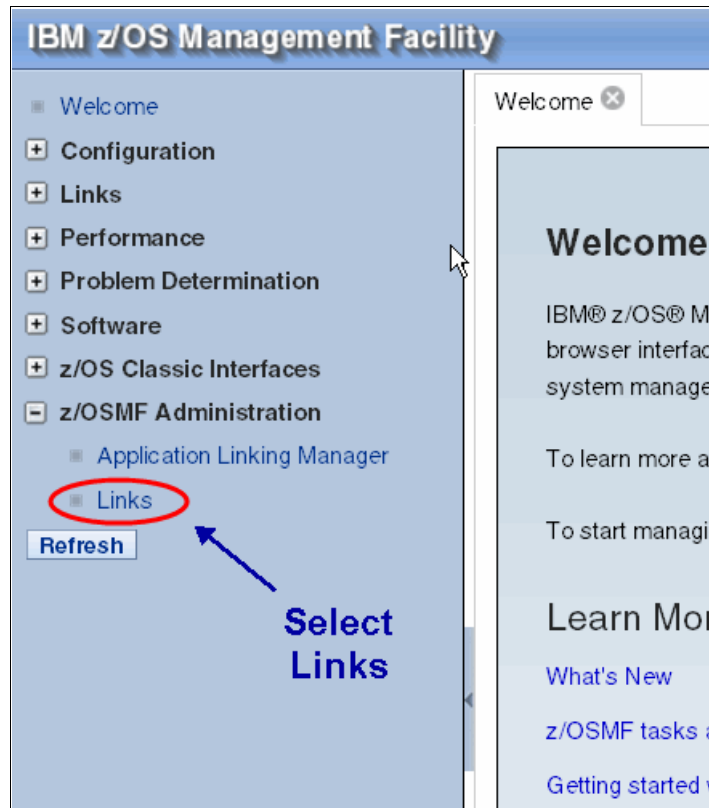


Figure 6-1 z/OSMF home window

1. To add the IBM zAware GUI as a link to z/OSMF, open **z/OSMF Administration** in the navigation tree and select **Links** as shown in Figure 6-1 on page 193.

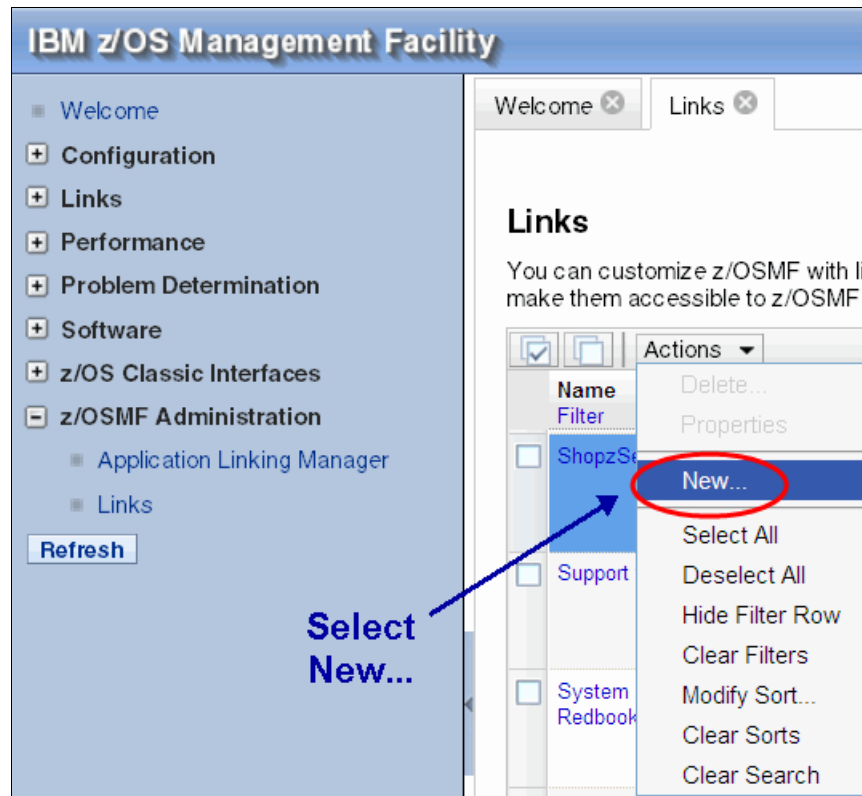


Figure 6-2 Select New from z/OSMF Links

2. When **Links** opens, the list of existing links appears. Click **Actions** → **New** as shown in Figure 6-2 on page 194.

Links ▸ New Link

New Link

Use this page to define a new link for z/OSMF.

* Name (maximum 30 characters):
ITSO IBM zAware 1

* SAF Resource Name Suffix (maximum 220 characters):
ZMMBASE.ZOSMF.LINK. zAware1

* URL (maximum 4000 characters):
https://IBM.zAware.IP.address/zAware/

* Category
Problem Determination

Open link in:

☒ New browser tab or window
☐ New z/OSMF tab

Authorizations

	User State	Description
<input checked="" type="checkbox"/>	SAF Authorized User	Access to z/OSMF tasks and links is contr
<input checked="" type="checkbox"/>	z/OSMF Authenticated Guest	User is logged into z/OSMF, but is not auth
<input type="checkbox"/>	z/OSMF Guest	User is not logged into z/OSMF.

OK Cancel

Figure 6-3 Create new link for IBM zAware

3. Enter the required data as shown in Figure 6-3:
 - a. Enter a name, the SAF resource name suffix, and the URL for the IBM zAware GUI.
 - b. Select the category based on where you want IBM zAware to be located in your z/OSMF navigation tree.
 - c. Select your choice as to where the window opens.
 - d. Select the authority required to use the link.
 - e. Click **OK**.

In our example, we choose to have IBM zAware appear in the **Problem Determination** category.

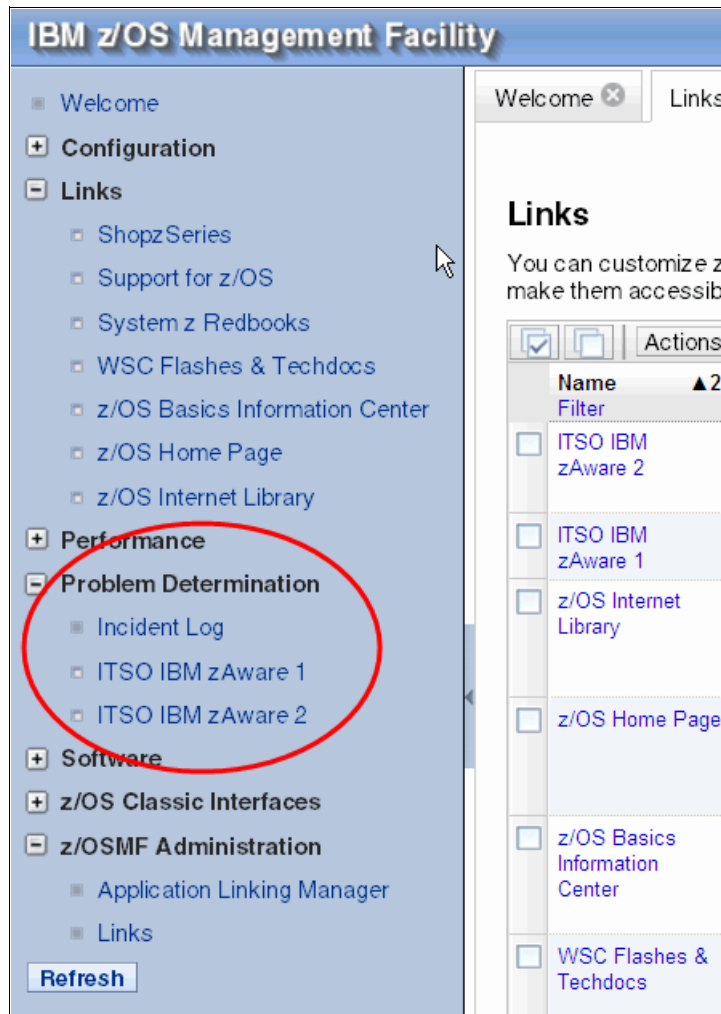


Figure 6-4 IBM zAware links added to z/OSMF

Figure 6-4 shows the results, with the links to the IBM zAware partitions contained in the Problem Determination category of z/OSMF.

Use these links whenever you want to access IBM zAware while signed in to z/OSMF.

6.3 IBM Tivoli NetView for z/OS

In relation to IBM zAware, NetView plays two roles. One role is that it provides an interface between a program running on z/OS and the IBM zAware API. The other role is that it provides an infrastructure (perhaps in tandem with your automation product) to do something with the information that is returned. First we discuss the interface role.

The Analysis results contained in the IBM zAware file system can be retrieved using an API provided with IBM zAware (the API is discussed in more detail in 6.4, “IBM zAware Application Programming Interface” on page 205). The API can deliver two levels of information back to your programs:

- ▶ A detailed list of message IDs and their attributes for a specific interval
- ▶ A summary of the anomaly score and message ID count for each analysis interval

Message ID count: When retrieving the summary information for a system, be aware that the message ID count is a count of the *unique* message IDs that occurred in that period. It is not a count of the total number of messages that were issued in the period.

The data returned from the API is formatted as XML records.

NetView provides a set of sample programs to make it easier for your programs to communicate with the IBM zAware API. The sample programs, together with information about their installation and use, can be downloaded from the NetView Tivoli System z Monitoring and Application Management wiki, available on the web at:

<https://www.ibm.com/developerworks/mydeveloperworks/wikis/home/wiki/Tivoli%20System%20z%20Monitoring%20and%20Application%20Management/page/Integration%20Scenarios%20for%20Tivoli%20NetView%20for%20zOS?lang=en>

The sample programs and their function are listed and briefly described in Table 6-1.

Table 6-1 Sample programs to retrieve IBM zAware XML data

Program	Description
ZAITIMER	The ZAITIMER program is designed to be called from a timer every 10 minutes. It will build the parameters for the ZAIPROC exec to retrieve the details of the previous 10-minute interval and then call ZAIPROC with those parameters. It retrieves the results from the previous 10-minute interval because those will be complete. The current interval results are still being calculated and might change. Detailed information about how to use the ZAITIMER program is provided in “Using ZAITIMER” on page 199.
ZAIPROC	The ZAIPROC program accepts parameters that control the type of information to be returned. ZAIPROC calls the ZAIGET program to retrieve information from the IBM zAware API. The output will be in single-line and multi-line messages, each of which is described in the comments of the program. Detailed information about how to use the ZAIPROC program is provided in “Using ZAIPROC” on page 200.
ZAIGET	The ZAIGET program performs the HTTPS connection to the IBM zAware application, and performs the API actions to retrieve the data. When the data is returned, the output is placed in a NetView safe ^a named ZAIGET, which the ZAIPROC program uses. Information about the use of ZAIGET is included in the download package.

- a. For more information about safes refer to the section about “PIPE SAFE” in the NetView “Programming: Pipes” manual, available online at:
<http://pic.dhe.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=%2Fcom.ibm.itnetvi.ewforzos.doc%2Fdqs12mst192.htm>

The ZAIPROC program creates single-line and multi-line messages that are exposed to the NetView Automation table, and that will also be available for browsing in the NetView CANZLOG. The messages contain the information retrieved from the IBM zAware application. You can use your automation table to process these messages. The various NetView functions are shown in Figure 6-5 on page 198.

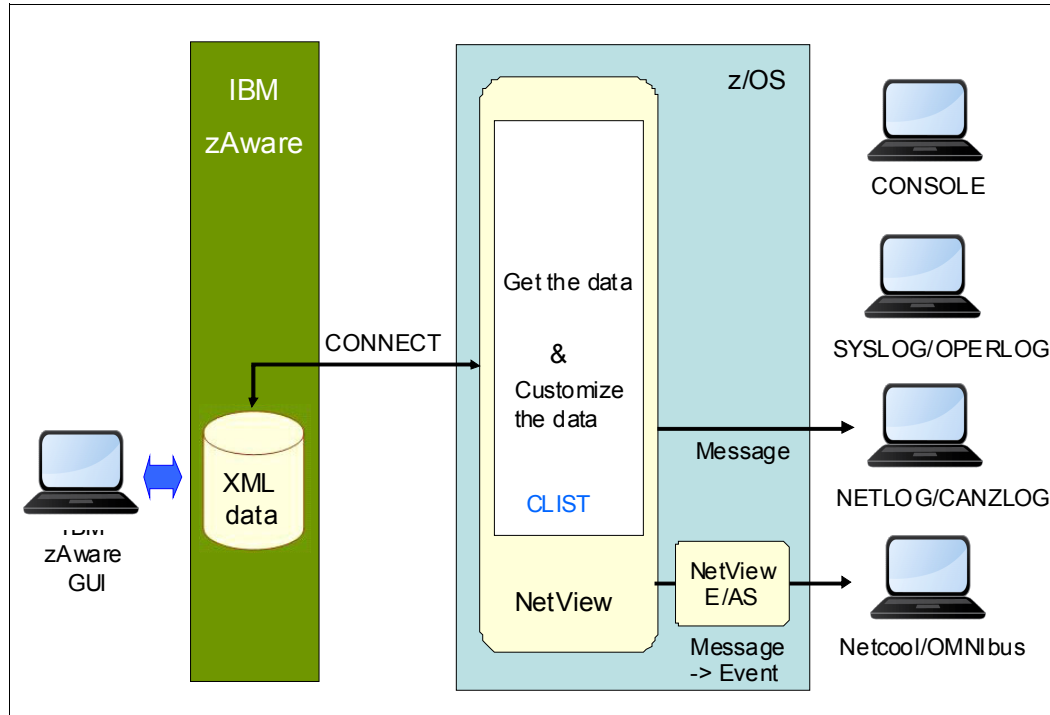


Figure 6-5 Retrieving data from IBM zAware API

6.3.1 Using the ZAI sample programs

The concept of the ZAITIMER and ZAIPROC sample programs is that ZAITIMER will be called on a regular basis (we suggest using a 10-minute interval to be in synch with IBM zAware). ZAITIMER calls ZAIPROC, passing it parameters to identify the interval that is to be retrieved. ZAIPROC then uses the ZAIGET sample program to send the request to the IBM zAware API.

The ZAIPROC program can provide results in one of the following two forms, depending on the setting of the REQUEST parameter when ZAIPROC is called:

- ▶ If REQUEST(INTERVAL) is specified, a detailed list of message IDs with an anomaly score greater than zero (0) that occurred in a particular interval is returned.
 - The default anomaly scores that will be returned are those above 96.5. You can override this using the ASCORE parameter on the ZAITIMER and ZAIPROC programs.
 - If there was data returned for an interval, but the anomaly score was less than the ASCORE value (default 96.5¹) the results will not display. Message ZAI0004I will be issued instead, stating that the result was filtered due to a low anomaly score. Setting the ASCORE value to zero (0) will cause all the records for that interval to be displayed.
- ▶ If REQUEST(LPAR) is specified, a summary list of anomaly scores and message ID counts for each interval in a day is returned.

When ZAIPROC is passed the parameters for a specific time interval, the results are returned in a multi-line message for automation in the NetView message automation table.

¹ When you analyze the information returned from IBM zAware, the anomaly score is used to determine how important the results of a particular interval are. The important anomaly scores are generally those above 96.5. They are important because they have the greatest deviation from the model of past behavior.

Example 6-1 shows sample output from ZAIPROC when the parameters specify to retrieve the previous 10-minute period from the IBM zAware application. The ZAI0001I message is used to return the requested information for any INTERVAL requests. The message includes the sysplex and system that the information relates to, and the time period that was retrieved. The times are always shown in UTC time. The message then shows the interval anomaly score for that period and each distinct message ID and its anomaly score. You can use the ZAI0001I message to drive your automation.

Example 6-1 ZAI0001I INTERVAL results message

ZAI0001I Interval Results.

System : UTCPLXSB-SP0
Interval: 2012-08-04T12:40:00.000Z
Anomaly : 57.4

Anomaly	Message	Count	Cluster	Contribution	Rarity
1.000000	CSQX520E	2	NEW	2.226	101.0
0.940000	CSQX470E	4	UNCLUSTERED	2.822	90.0
0.871000	IXG284I	1	UNCLUSTERED	2.051	20.0
0.865000	IEF285I	1	UNCLUSTERED	2.012	2.0
0.839000	IEC070I	30	UNCLUSTERED	1.832	20.0
0.801000	DFS2864I	8	UNCLUSTERED	1.618	5.0
0.670000	IGD104I	1	UNCLUSTERED	1.11	18.0
0.645000	IEF237I	2	UNCLUSTERED	1.038	2.0
0.618000	DFS994I	2	UNCLUSTERED	0.964	21.0
0.202000	CSQX004I	1	UNCLUSTERED	0.226	26.0

Alternatively, when ZAIPROC is called with REQUEST(LPAR), the information is returned in message ZAI0005I. The message lists the sysplex and system and the intervals for the date given to the program.

If the date given is the same as the current UTC date, the ZAIPROC program will filter the results to return the last six intervals, starting from the current UTC time and going back six intervals. If the date is not the same as the current UTC date, ZAIPROC will not filter the result and every interval score for that date, 144 entries, will be displayed.

Example 6-2 shows an LPAR request for the current UTC date. This returns the anomaly score and message ID count for the last six intervals.

Example 6-2 ZAI0005I LPAR results message

ZAI0005I Interval Results.

System : UTCPLXSB-SP0
Interval: 2012-08-04T00:00:00.000Z

Anomaly	MessageIDs
37.70000	21 Oldest interval
51.00000	39
14.20000	20
57.40000	43
57.40000	62
51.00000	30 Newest interval

Using ZAITIMER

The main function of the ZAITIMER program is to calculate the previous time period using UTC time, and supply that and other derived parameters to the ZAIPROC program. It uses

the previous interval rather than the current one because the results are complete, and therefore have a final anomaly score for that interval.

You typically code the call to ZAITIMER on a timer in NetView² and run it every 10 minutes. It does not need any parameters. However, you can specify any of the parameters listed in Table 6-2. These parameters are then passed to the ZAIPROC program, which is called from the ZAITIMER program.

Table 6-2 ZAITIMER parameters

Parameter	Value
PLEX(.....)	This contains the sysplex name to retrieve information about.
SYS(...)	This contains the system name to retrieve information about.
SERVER(...)	This is the IP address of the IBM zAware server.
ASCORE(...)	This is the interval anomaly score filter value. This will change the default filter, so you can retrieve intervals that have anomaly scores lower than the default but higher than the value specified here. If you specify a value of zero (0), the results for the interval will be displayed regardless of the interval anomaly score.

The commands shown in Example 6-3 include both methods of executing ZATIMER. Firstly without parameters, meaning that the program will determine the PLEX and SYS parameters itself. And secondly with parameters, where the parameters are supplied for the sysplex, system, and anomaly score parameters.

Example 6-3 ZATIMER execution examples

```
ZAITIMER

ZAITIMER PLEX(UTCPLXSB) SYS(SPO) ASCORE(50)
```

Using ZAIPROC

The ZAIPROC program has two main purposes:

- ▶ To retrieve the requested information from IBM zAware using the sample ZAIGET program
- ▶ To parse the XML data that is retrieved from IBM zAware and send the results to the NetView netlog for viewing and processing through the NetView automation table

ZAIPROC issues ZAIInnna messages to present the information received from the API.

ZAIPROC can be called from other programs, or directly from the command line.

ZAIPROC has certain default parameters that are required for it to execute successfully. These are retrieved from NetView common global variables. The common global variables can be set in the CNMSTYLE processing³ when NetView starts, or dynamically with the NetView **RESTYLE COMMON** command.

² Refer to "Timer Commands" in the "NetView Program Automation Facilities" chapter of the NetView "Automation Guide", online at:
<http://pic.dhe.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=%2Fcom.ibm.itnetviewforzos.doc%2Fdqa12mst203.htm>

The required global variables are documented in Table 6-3 and in the program. The common global values can be overridden by supplying them as parameters to the program when it is executed.

Table 6-3 ZAIPROC global variables in CNMSTYLE

Common global variable name	Value description
zaiproc.zAwareServer	[Mandatory] The IP address or host name of the IBM zAware application.
zaiproc.zAwareUserid	<p>[Mandatory] The user ID that will logon to the IBM zAware application to execute the API.</p> <ul style="list-style-type: none"> ► It can be overridden by a parameter passed to the program. ► The password for the user ID will be retrieved from the NetView password file using GETPW, or you can specify the password as a parameter on the program call. ► The user ID need only have USER authority in the IBM zAware application.
zaiproc.zAwareAscore	[Optional] The value of the Anomaly score for which results that are greater than this value will be returned. If the value is zero (0), then all results are returned.

The ZAIPROC program has a number of parameters that are used to build the API request used by ZAIGET. Each parameter is explained in Table 6-4.

Important: When calling ZAIPROC, the command needs to be prefixed with the **NETVASIS** keyword. This is because the password character case needs to be retained.

Table 6-4 ZAIPROC parameters

Parameter	Value description
REQUEST(...)	[Mandatory] INTERVAL or LPAR
DATE(...)	[Mandatory when using REQUEST(LPAR)] Format UTC: mmddyyyy
TIME(...)	[Mandatory when using REQUEST(INTERVAL)] Format UTC: yyymmddhhmm00
PLEX(...)	[Mandatory] Sysplex name
SYS(...)	[Mandatory] System name
SERVER(...)	[Optional] IP address or host name of the IBM zAware application.

³ Refer to “Updating the CNMSTYLE Member” in the “Getting Ready to Start NetView” chapter of the NetView “Installation: Getting Started” manual, available online at:
http://pic.dhe.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=%2Fcom.ibm.itnetviewforzos.doc%2Fing10mst102.htm&path%3D65_1_8_4

Parameter	Value description
ASCORE(...)	[Optional] Anomaly score filter applied to data returned on the REQUEST(INTERVAL) calls. <ul style="list-style-type: none"> ► Any anomaly score for an interval that is below this value will not be displayed. ► Setting the value to zero (0) will return all results with no filtering.
USER(...)	[Optional] The user ID that will logon to the IBM zAware application to retrieve the XML data.
PASS(...)	[Optional] The password for the user ID. Case sensitive. <ul style="list-style-type: none"> ► When not specified, the password for the user ID will be retrieved from the NetView password file using GETPW. ► When specified, use the NETVASIS command before ZAIPROC.

Next we show a few examples of calling the ZAIPROC program with different parameter combinations.

First, Example 6-4 uses a password on the parameters. Because the password is case sensitive, the **NETVASIS** command has been added to the front of the **ZAIPROC** command. This makes NetView keep the case the command was entered in. Otherwise, NetView will convert the command and parameters to uppercase.

Example 6-4 Using ZAIPROC to retrieve a specific interval result from IBM zAware

```
netvasis zaiproc request(interval) time(20120804090000) plex(utcplxsbs) sys(sp0)
pass(rabbit)
```

ZAI0001I Interval Results.

```
System :          UTCPLXSB-SP0
Interval:      2012-08-04T09:00:00.000Z
Anomaly :          37.7
```

Anomaly	Message	Count	Cluster	Contribution	Rarity
1.000000	DSNB260I	1	NEW	0.226	101.0
0.940000	CSQX470E	4	UNCLUSTERED	2.822	90.0
0.871000	IXG284I	1	UNCLUSTERED	2.051	20.0
0.865000	IEF285I	1	UNCLUSTERED	2.012	2.0
0.839000	IEC070I	30	UNCLUSTERED	1.832	20.0
0.819000	DFS2864I	6	UNCLUSTERED	1.714	5.0
0.670000	IGD104I	1	UNCLUSTERED	1.11	18.0
0.645000	IEF237I	2	UNCLUSTERED	1.038	2.0

Here, Example 6-5 shows a request for the LPAR information for the current day. This means that the current UTC time of the system is retrieved, the number of 10-minute periods since the start of the day is calculated, then the Anomaly scores for the last six intervals are returned.

Example 6-5 Using ZAIPROC to retrieve LPAR summary of last six intervals from IBM zAware

```
zaiproc request(lpar) date(08042012) plex(utcplxsbs) sys(sp0)
```

ZAI0005I Interval Results.

System : UTCPLXSB-SP0
Interval: 2012-08-04T00:00:00.000Z

Anomaly	MessageIDs	
20.10000	18	Oldest interval
51.00000	39	
37.70000	27	
73.00000	75	
31.50000	22	
31.50000	22	Newest interval

Define NetView password file

To store a password in NetView for the user ID that logs onto the IBM zAware application, you need to use the NetView (AON) password data set.

The sample JCL for defining this data set can be found in the NetView CNMSAMP data set, member CNMSJ004. The member with the IDCAMS statements to define the password data set is EZLSI01.

After you have defined the data set, you can allocate it to the NetView CNMSJ009 procedure using DD EZLPSWD.

To define the password for the user ID that logs onto the IBM zAware application, you use the GETPW command. You also need to use the **NETVASIS** command in front of GETPW, to ensure the case of the password is retained. When you use **NETVASIS**, make sure the parameter keywords are in uppercase, because GETPW expects them to be that way.

Imagine you have a user ID of A1234B defined in the IBM zAware application, with a password of 12@aBcE!, which is mixed case. And the NetView domain that executes the ZAIPROC program, which uses GETPW to retrieve the password, is CNM01.

The commands in Example 6-6 show how to define the password in the password file using GETPW.

Example 6-6 Defining a password with GETPW

```
netvasis getpw a1234b cnm01,INIT=12@aBcE!
```

Retrieving XML data from IBM zAware with ZAIGET

AT-TLS function of TCP/IP must be activated: Because the IBM zAware API uses an HTTPS connection, the AT-TLS function of TCP/IP needs to be activated on z/OS. Refer to Appendix B, “Activating TCP/IP AT-TLS” on page 213 for a sample configuration.

The ZAIGET program performs all the TCP/IP communications with the IBM zAware application. It accepts parameters passed on the command line.

The ZAIGET program needs to be called from another program when you want to view the data retrieved by it. The data is returned into a SAFE, and the SAFE only lives as long as the REXX execution environment that called ZAIGET exists.

In this book, we use ZAIPROC as the front-end to ZAIGET. It is ZAIPROC that does the parsing of the data returned from the API calls that ZAIGET performs.

ZAIGET expects a task global variable, ZAWARE.URL, to be set with the HTTP details of the API request, and expects the user ID and password to be passed as parameters to the program. The details of its parameters are contained in the program comments.

ZAIGET performs the following functions:

1. It connects to the IBM zAware application and logs on the user ID.
2. It send the API parameters to the server.
3. It captures the returned data.
4. It outputs the data into a named SAFE, called ZAIGET.

6.3.2 Using the information retrieved from the IBM zAware API

A typical z/OS system has many monitors including MVS, IBM CICS, IBM DB2, and MQ performance monitors. It also has health check monitors, network performance monitors, network management consoles, workload scheduler consoles, automation consoles, and possibly many others. The last thing you need is yet another console to have to monitor.

Alternatively, the information that is provided by IBM zAware might be critical in helping you avoid an outage. How do you balance the need to limit the number of consoles that operators must monitor with the need to be aware of anomalies that IBM zAware might detect?

By using the sample programs documented in this book, you can integrate the information from IBM zAware into your operations management environment. The ZAIInnna messages can be processed by your automation product, and alerts can be raised based on thresholds that you specify.

For example, you might set up ZAITIMER to use a relatively low interval Anomaly score so that information about most anomalous messages in the interval is returned to NetView. This means that the information is available in CANZLOG for quick viewing if you need it.

Alternatively, you can set up your automation to only raise an alert if the interval Anomaly score is greater than a higher value such as 96.5, for example. This way, you have all the information you are likely to want in CANZLOG, but you do not have automation constantly raising alerts that operators will soon learn to ignore.

Another important use of NetView and the IBM zAware API is to regularly check the health of the IBM zAware LPAR. You probably will not have someone constantly monitoring the IBM zAware GUI, but you want to be aware if some problem causes the IBM zAware application or one of its components to stop working. You will not want to turn to IBM zAware to help debug a potential problem situation, only to find that it has not been collecting data for the last three hours.

You can monitor the health of the IBM zAware application by using the REQUEST(LPAR) flavor of the ZAIPROC program to retrieve information for the whole day. In this case you will probably be more interested in the fact that you actually received a response, than in the information returned. Because the ZAITIMER exec is designed to call the REQUEST(INTERVAL) flavor of ZAIPROC, you need to provide some automation to build the appropriate parameters to call ZAIPROC, but that should be a minor effort.

As part of monitoring the health of the IBM zAware environment, you will check that the connections from System Logger to IBM zAware are active. However, that checking uses System Logger commands rather than the IBM zAware API. That monitoring is discussed in Chapter 5, "Maintaining and managing IBM zAware" on page 153.

If you have IBM Tivoli System Automation for z/OS, you can then perhaps create a logical resource to track the status of the IBM zAware application. For example, you might track whether it is running or not, or whether the connection from OPERLOG to IBM zAware is active. There are many ways the information can be transformed and used to extend the management of IBM zAware.

6.4 IBM zAware Application Programming Interface

The IBM zAware application provides an API that enables you to retrieve information about the Analysis view results. There are two types of results available, INTERVAL and LPAR, as described in Table 6-5. The details of the API are documented in *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623.

Table 6-5 API request types

Request type	Description
INTERVAL	Return a 10-minute analysis interval result as an XML package.
LPAR	Return all the summary results for every interval for a full day, all 144, 10-minute periods in an XML package.

The IBM zAware API requires the use of Secure Sockets (HTTPS). Therefore, as previously mentioned, if you are going to use the API from z/OS you will need to activate the TCP/IP AT-TLS function. Details about AT-TLS and how we activated it in the ITSO systems are contained in Appendix B, “Activating TCP/IP AT-TLS” on page 213.



A

Syslog Message Analysis Program

This appendix explains to use the Message Analysis Program to analyze your console message traffic in the context of a IBM zAware environment.

Message Analysis Program

To help clients better understand their console message traffic, and set up message suppression lists, Message Processing Facility (MPF) exits and Message Flood Automation (MFA), IBM provides a program called the Message Analysis Program (MAP). This program is available on the z/OS Tools and Toys web site at:

<http://www-03.ibm.com/systems/z/os/zos/features/unix/bpxa1ty2.html>

The program is provided on an as-is basis.

In addition to its use with automation, MPF, MFA, and so on, the program also provides information that might be valuable in an IBM zAware environment.

Capacity planning

The amount of CPU, disk, and network capacity that is required for IBM zAware is a factor of the following items:

- ▶ The peak rate at which messages are produced and sent to IBM zAware
- ▶ The number of unique message IDs
- ▶ How long the information is kept by IBM zAware

MAP provides you with the first two of these three items.

Running the MAP program

To run the MAP program, first download the zip file from the Tools and Toys web site. In the zip file, you will find a README file that contains instructions for uploading the other MAP files to your z/OS system. Note that some of the files must be uploaded in Binary format, and others in Text format.

The README file also includes instructions for link-editing the MAP object code.

When you have all the files uploaded to z/OS and the program has been link-edited, the next step is to create the input file or files for the program. MAP can process syslog files and files that have been created from OPERLOG using the IEAMDBLG program.

There might be a few changes you need to make to the control statements for the program:

- ▶ The default date format that MAP expects is 2-digit years (this is controlled by the HCFORMAT keyword in your CONSOLxx member). If you are using 4-digit years, you need to specify "YEAR4" in the MAP control statements.
- ▶ If the input file contains carriage control characters, but is not defined as VBA, you need to adjust for that by telling the program to offset all records by one byte. You do this by specifying OFFSET(1) in the MAP control statements.

A sample set of JCL to run the MAP program is shown in Example A-1.

Example A-1 Sample JCL to run MAP program

```
//MSGLG610 JOB (0,0),  
//          CLASS=A,MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=&SYSUID  
//* *****  
//*          ALL MESSAGES.
```



```

/* *****
//JOB LIB      DD DSN=KYNEF.MSGLG610.LOAD,DISP=SHR
//MSG RATE     EXEC PGM=MSGLG610,REGION=5000K
//OPT IN      DD *
TITLE: ZAWARE TEST ITS0
OFFSET(1)
YEAR4
REPORT(SUM,AMSG,CMSG,MFRQ,RTGR,IMRT)
RATMSGSGS(ALL)
MSGINTV(5)          -- 1 HOUR INTERVALS
* MSGSCAL(RELATIVE)
MSGSCAL(LINEAR)
MSGGRANGE(32000)    -- USE WITH MSGSCAL(LINEAR) ONLY
MSGLNECT(85)        -- LINES BETWEEN PAGE-SKIPS (DEFAULT 60)
/*-----*
/* 'DUMPTBL(MSG,RATE)' SHOULD BE SPECIFIED AS AN OPTION IF *
/* STATISTICAL DATA IS BEING COLLECTED FOR RETURN TO IBM. *
/*-----*
//DATA      DD DSN=KYNEF.WTSCPLX1.OPERLOGS,DISP=SHR
//TTLLIB    DD DSN=KYNEF.MSGLG610.TEXT,DISP=SHR
//SYSUDUMP  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
/*-----*
/* IF USED, THE FOLLOWING DD'S SHOULD HAVE:  OUTPUT=(*.STDOUT) *
/*-----*
/*LABEL     DD SYSOUT=*
//IPLST     DD SYSOUT=*
//PRNTOUT   DD SYSOUT=*
//COMPRATE  DD SYSOUT=*
//COMPMSG   DD SYSOUT=*
//UNKNOWN   DD DUMMY
//PREVIEW   DD DUMMY
//IMSGRATE  DD DUMMY
//BURST     DD DUMMY
/*-----*
/* THE FOLLOWING DD'S ARE USED TO DUMP INTERNAL TABLES FOR REUSE. *
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
/* -----> IF USED, 'DUMPRATE' -MUST- HAVE A DISP OF 'MOD' <----- *
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
/* THE 'DUMPMMSG' DD AND 'DUMPRATE' DD SHOULD BE SPECIFIED IF *
/* STATISTICAL DATA IS BEING COLLECTED FOR RETURN TO IBM. *
/*-----*
//DUMPCNT   DD DUMMY
//DUMPMMSG  DD DUMMY
//DUMPCMD   DD DUMMY
//DUMPRATE  DD DUMMY
//OUT       DD DUMMY

```

The //DATA DD card must be updated to point at your input file. Also, the JOBLIB and TTLLIB DD cards must be updated to point at wherever you stored the MAP program and the message definitions that are provided with MAP.

The program can provide many different reports (they are all described in the GUIDE160.HTML file in the zip file). But here we simply focus on the ones that are of interest in the context of this book.

When you run this job, the output will contain a number of sections. The ones that we are interested in are **** Message IDs in Alphabetic Order ****. The last five lines of that report will look something like the text in Example A-2. Specifically, you are interested in the number of unique message IDs.

Example A-2 Identifying number of unique message IDs

```
100010 Total message lines
      863 Unique message IDs
17079 Total messages without IDs processed
      83 Unique messages without IDs identified
3.351 Entries searched on average
```

The job output contains a graph titled “message rate / second for: All message traffic incl. cmd responses all msgs selected.” There are four pairs of Message Rate graphs. The first graph of each pair shows message rates as a percentage of total message traffic. The second graph of each pair shows message rates as a percentage of the total number of intervals. A sample is shown in Figure A-1.

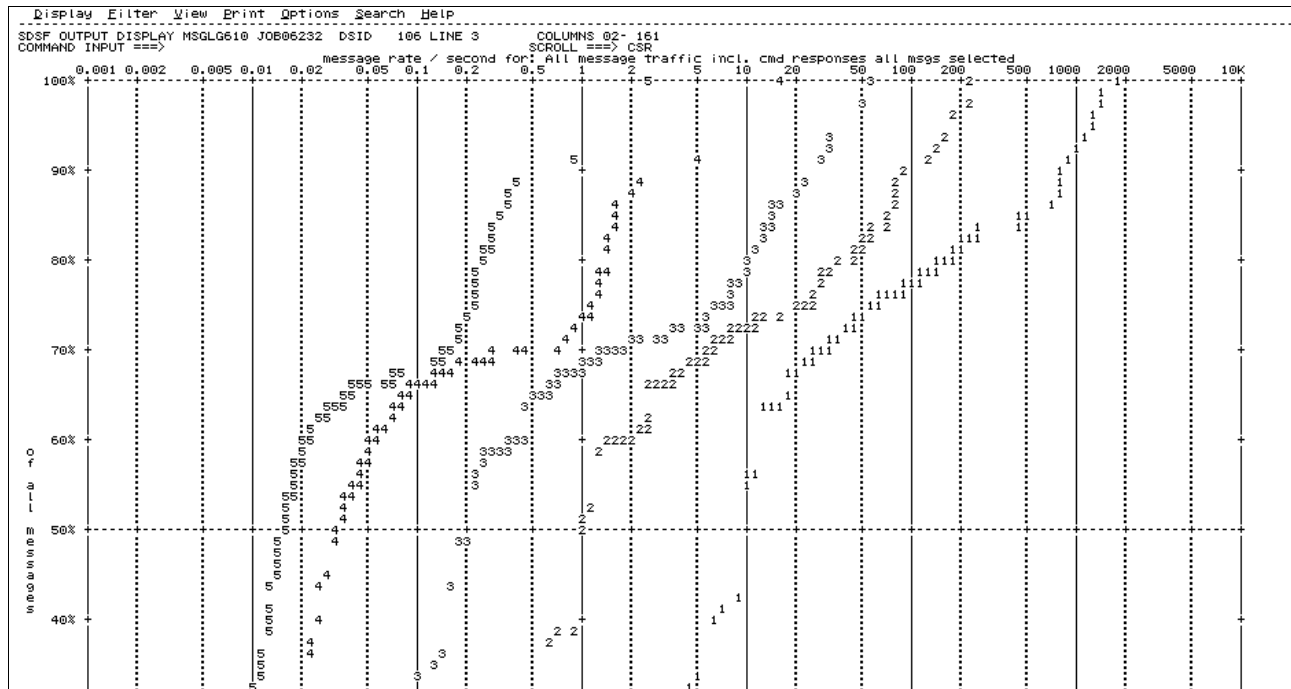


Figure A-1 Identifying peak message rates

In the chart, note the following points:

- ▶ The 5s indicate the message rate per hour.
- ▶ The 4s indicate the message rate per 10-minute interval.
- ▶ The 3s indicate the message rate per minute.
- ▶ The 2s indicate the message rate per 10-second interval.
- ▶ The 1s indicate the message rate per second.

For example, in the chart in Figure A-1, you can see that the 1-second interval line intersects the 80 percent horizontal line at the 200-message per second vertical line. This means that 80 percent of the messages found on the log occur in busy 1-second intervals having an instantaneous message rate of 200 messages per second or less and 20 percent of the messages found on the log occur in busy 1-second intervals having an instantaneous

message rate of 200 messages per second or more. In this example, the 10-second interval line intersects the 100 percent line at about 250 messages per second.

Assuming that each message is about 130 bytes and that you want to cater for the busiest 10-second interval, in this example, you might want to provide a bandwidth of at least 32.5 KB per second for this LPAR.

Determining ability to build a model

As described in 3.2, “Selecting which systems to monitor with IBM zAware” on page 96, to be able to create a representative model for a system, IBM zAware needs a minimum number of unique message IDs. In addition, each of those messages must occur a minimum number of times. When considering which systems are good candidates for monitoring with IBM zAware, ensure that the message traffic from those systems meets the modeling criteria.

You can use the Message Analysis Program to identify the number of unique message IDs from each system *and* the number of occurrences of each message.

As shown in Example A-2 on page 210, MAP displays the number of unique messages IDs in the input file that you provide to it. The number of occurrences of each message is provided in the MAP report entitled “** Message IDs by Frequency”. The report is sorted by number of occurrences, with the most frequently occurring messages at the top.

Example A-3 Determining number of occurrences of each message

** Message IDs by Frequency					
Total		ID cnt	ML cnt	ML min	ML max
20350 U/S	<IEF196I	20350			
5120 U/S	DFS3258A	5120			
4297 U/S	+CSQX004I	2149	2148	1	1
4286 U/S	HZS0002E	1196	3090	1	4
4179 U/S	HZS0001I	1603	2576	1	2
3580 U/S	<IXC467I	1072	2508	1	3
3023 U/S	<IOS2001I	763	2260	2	3
2557 U/S	<IXL014I	812	1745	2	4
2521 U/S	???0029	2521			
2243 U/S	<IEE252I	2243			

To determine whether your input file meets the model criteria, get the number of unique message IDs, then count the number of messages that only occurred once or twice and subtract that from the number of unique message IDs. If the resulting value is too small, you can try increasing the number of days in the input file and see if that helps you meet the criteria.

A useful example of a system to do this for is a system that acts solely as a network owner. Those systems will typically not have a large number of unique message IDs, but they might still be systems that you want IBM zAware to monitor.



Activating TCP/IP AT-TLS

This appendix contains examples that show how we configured AT-TLS on the ITSO test system to enable us to use IBM Tivoli NetView for z/OS to access IBM zAware information using the IBM zAware API. AT-TLS is needed to provide the Secure Sockets connection for the HTTPS request API provided with the IBM zAware application.

Impact of AT-TLS on TCP/IP startup

The IBM zAware API allows you to retrieve either the current day's interval results, in summary, or each interval's result in detail.

Keep the following considerations in mind when you decide to activate the AT-TLS function of TCP/IP:

- ▶ Server certificates from the IBM zAware application will need to be imported into your security product's stored key facilities. In RACF we used the SITE virtual keyring. This allows any user ID with access to RACF profile IRR.DIGTCERT.LISTRING to use the IBM zAware certificate to connect to the IBM zAware API.
- ▶ When AT-TLS is activated, it will not allow connections to the TCP/IP stack until the policy for AT-TLS has been loaded. This can impact your applications in the following ways:
 - It can cause message EZZ4248E TCPIP WAITING FOR PAGENT TTLS POLICY to be issued. Refer to “Common AT-TLS startup errors” in *z/OS Communications Server Diagnosis Guide*, GC31-8782, for more information about this topic.
 - The normal TCP/IP initialization message is no longer the one that you should use in your automation product to indicate TCP/IP is available. If you do not change the message that your automation uses to flag that TCP/IP initialization has completed, your applications that are dependent on TCP/IP will start too early, resulting in socket connection failures.
 - When AT-TLS is activated, the TCP/IP initialization complete message changes to EZZ4250I.
 - You also need to stop messages EZB6473I and EZAIN11I from being used as initialization complete messages in your automation. Under IBM Tivoli System Automation for z/OS, add a message trap in INGMMSGU1 as shown in Example B-1:

Example B-1 NetView automation table statements

```
IF (MSGID= 'EZB6473I' | MSGID='EZAIN11I')  
  THEN CONTINUE(N);
```

- ▶ The Policy Agent (PAGENT) started task for TCP/IP needs to be started because it is responsible for loading the AT-TLS policy.
- ▶ The started task user ID associated with the Policy Agent address space in your security tool needs access to profile EZB.INITSTACK.*sysname.tcpname* in the SERVAUTH class. This allows the Policy Agent to connect to the TCP stack, before it has loaded the AT-TLS policy.
- ▶ If you want any of your applications to access the stack before the AT-TLS policy has been loaded, then you also need to grant them READ access to the EZB.INITSTACK.*sysname.tcpname* in the SERVAUTH class.

AT-TLS policy for IBM zAware certificate

When we created the policy for AT-TLS on the ITSO test system, we used a SITE certificate to store the IBM zAware applications certificate.

To use the SITE certificate of RACF in the AT-TLS policy, we used statements as shown in Example B-2 on page 215.

Example B-2 AT-TLS policy statement for SITE certificate use.

TTLSEKeyringParms	keyR~TRAINER
{	
Keyring	*SITE*/*
}	

Adding the IBM zAware certificate to the RACF SITE keyring

In our ITSO system, we used RACF to control access to the IBM zAware application certificate.

The IBM zAware certificate was retrieved from the server by using the export certificate feature of Mozilla Firefox. This function is accessed through the **tools** → **options** → **Advanced icon** → **Encryption tab** → **View Certificates** button. Locate the IBM zAware LPAR's host name or IP address, select the certificate, and use the export button to export the certificate.

Because the IBM zAware default certificate is self-signed (implying no certificate chain), we exported it as a DER file. We then uploaded the certificate to a sequential data set on z/OS as a binary file.

After the file became available on z/OS, we imported that certificate into RACF to store in its SITE keyring using the commands shown in Example B-3.

Example B-3 RACF commands to store server certificate.

```
RACDERT ADD('CERT.ZAWARE.DER.BIN') SITE WITH LABEL('zAware')
```

Creating the client certificate for the system

We also created a client certificate for the IBM zAware connection. Although this step might not be necessary, we show the commands needed to accomplish this in Example B-4.

Example B-4 Create client certificate in RACF.

```
RACDERT SITE GENCERT WITH LABEL('IBMZAWARECLIENT')  
SUBJECTDSN(CN('IBM_zAware_client')) KEYUSAGE(HANDSHAKE) NOTAFTER(DATE(2017-01-01))
```

AT-TLS policy file details

Example B-5 contains the policy file we used in our test environment to allow a connection from z/OS to the IBM zAware API using secure sockets.

Example B-5 AT-TLS policy file.

TTLSERule	Default_zAware-Requester~1
{	
LocalAddr	ALL
RemoteAddrGroupRef	zAware_Servers
LocalPortRangeRef	portR1

RemotePortRangeRef	portR2
Direction	Outbound
Priority	255
TTLSTGroupActionRef	gAct1~zAware-Requester
TTLSEnvironmentActionRef	eAct1~zAware-Requester
TTLSTConnectionActionRef	cAct1~zAware-Requester
}	
TTLSTGroupAction	gAct1~zAware-Requester
{	
TTLS-enabled	On
}	
TTLSEnvironmentAction	eAct1~zAware-Requester
{	
HandshakeRole	Client
EnvironmentUserInstance	0
TTLSKeyringParmsRef	keyR~TRAINERA
}	
TTLSTConnectionAction	cAct1~zAware-Requester
{	
HandshakeRole	Client
TTLSCipherParmsRef	cipher1~Default_Ciphers
TTLSConnectionAdvancedParmsRef	cAdv1~zAware-Requester
CtracedClearText	Off
Trace	0
}	
TTLSTConnectionAdvancedParms	cAdv1~zAware-Requester
{	
CertificateLabel	zawareclient
SecondaryMap	Off
}	
TTLSKeyringParms	keyR~TRAINERA
{	
Keyring	*SITE*/*
}	
TTLSCipherParms	cipher1~Default_Ciphers
{	
V3CipherSuites	TLS_RSA_WITH_AES_256_CBC_SHA
V3CipherSuites	TLS_DHE_RSA_WITH_AES_256_CBC_SHA
V3CipherSuites	TLS_DH_RSA_WITH_AES_256_CBC_SHA
V3CipherSuites	TLS_DHE_DSS_WITH_AES_256_CBC_SHA
V3CipherSuites	TLS_DH_DSS_WITH_AES_256_CBC_SHA
V3CipherSuites	TLS_RSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites	TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites	TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites	TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
V3CipherSuites	TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA
V3CipherSuites	TLS_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites	TLS_DHE_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites	TLS_DH_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites	TLS_DHE_DSS_WITH_AES_128_CBC_SHA
V3CipherSuites	TLS_DH_DSS_WITH_AES_128_CBC_SHA
}	
IpAddrGroup	zAware_Servers
{	


```

    IpAddr
    {
        Addr 9.nn.n.nnn
    }
    IpAddr
    {
        Addr 9.nn.n.nnn
    }
}
PortRange                                portR1
{
    Port                                1024-65535
}
PortRange                                portR2
{
    Port                                443
}

```



Using automation to monitor IBM zAware connections

This appendix provides a sample REXX exec that can be used by your z/OS system automation product to query the status of the System Logger connections to the IBM zAware application.

Querying the status of connections to IBM zAware

System Logger issues a variety of IXG3xx messages when it is starting or stopping its connections to the IBM zAware application. It is advisable to add those messages to your automation.

Additionally, you might also want to perform periodic checks of the connections to ensure that they are working as expected.

To verify that a connection is active, you can create a special monitor routine that performs the following actions:

1. Issue the following command:
`D LOGGER,C,LSN=operlogLogStream,D`
2. Capture the response message IXG601I, which is a MLWTO.
3. Locate the following string:
`ZAI CLIENT:`
4. Scan that line to see if it contains the status `CONNECTED` or `QUIESCED`.
 - a. If it finds `CONNECTED`, end the monitor with an ok return code.
 - b. If it finds anything else, for example, `QUIESCED`, change the state of the resource to `FAILED`.

If using NetView, this routine can be performed with the sample REXX program shown in Figure C-1.

```

/*-----REXX-----*/
parse source . . ident . .
say ident||': Started....'

'PIPE CC D LOGGER,c,lsn=sysplex.operlog,d',
'| SEP',
'| LOCATE /ZAI CLIENT:/',
'| EDIT WORD 5.3 N',
'| VAR CMDOUT'

'PIPE CC D LOGGER,STATUS,ZAI',
'| SEP',
'| LOCATE /SERVER/',
'| EDIT WORD 2.4 N',
'| VAR ZAIOUT'

if cmdout = 'YES - CONNECTED' then
do
/* Add command to update the status to available. */
say ident||': Connection to zAware is connected. Parms: 'zaiout
rc = 0
end
else
do
/* Add command to update status to failed. */
say ident||': Connection to zAware is quiesced. Parms: 'zaiout
rc = 8
end
exit rc

```

Figure C-1 NetView REXX to check LOGGER connection to IBM zAware application



D

Problem determination sample

This appendix provides an example of the steps to follow when you need to determine why a z/OS system will not connect to the IBM zAware application.

Diagnosing System Logger connection errors

This sample provides an example of a problem whereby System Logger was unable to connect to the IBM zAware application. The sample illustrates the key messages to look for, and explains which documents provide the information you need to address this problem.

In our sample, when attempting to connect to IBM zAware, System Logger failed and displayed the message shown in Example D-1.

Example D-1 System Logger error message

```
IXG372I ZAI LOGSTREAM CLIENT MANAGER ERROR 758
FOR LOGSTREAM SYSPLEX.OPERLOG
FUNCTION=BPX1CON  ERRNO=00000467  ERRNOJR=76630291
```

The error indicates that the function experiencing the failure was BPX1CON (FUNCTION=BPX1CON). It also provides two sets of error codes, namely ERRNO=00000468 ERRNOJR=76630291.

To connect to IBM zAware, System Logger uses UNIX System Services Assembler callable services. These callable services are described in *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803.

The BPX1CON service is the equivalent of a socket connection call (trying to establish a connection between two sockets). In this case, the call would be between System Logger and IBM zAware.

The error and reason codes are described in *z/OS V1R13.0 UNIX System Services Messages and Codes*, SA22-7807. The return code ERRNO=00000468 (hex value) indicates ECONNREFUSED = The attempt to connect was rejected.

The reason code ERRNOJR=76630291 (hex value) is actually 7663 (reason code qualifier) and 0291 (reason code). Researching the reason code qualifier shows us that it is z/OS TCP/IP-related. The reason code indicates JrTcpError (TCP returned an error identified by the return code).

So, the error is revealed to be a socket connection error. We follow these steps to determine why the error occurred:

1. We need to know the endpoint connections. On the system that was experiencing the problem, IEASYS sets IXGCNF=00.
2. IXGCNF sets the SERVER value to 10.1.110.3. So we are trying to reach the IBM zAware server using this IP address. This means that we have to have some route to this IP address.
3. Trying to ping 10.1.110.3 from this system times out. This indicates a routing issue.

Now we have to discover why we cannot reach this address (routing issue). For a useful guide to resolving TCP/IP routing issues, see “Diagnosing network connectivity problems” in *z/OS Communications Server Diagnosis Guide*, GC31-8782.

4. Next we display the routes on the system by using the **D TCPIP, ,NETSTAT,ROUTE** command. We are looking for something with an 10.X.X.X IP address range. The output from the command is shown in Example D-2.

Example D-2 Output from D TCPIP,NETSTAT,ROUTE command

```
EZZ2500I NETSTAT CS V1R13 TCPIP 714
DESTINATION      GATEWAY      FLAGS      REFCNT      INTERFACE
DEFAULTNET       9.12.4.1     UGS        0000000000  OSA2100LNK
DEFAULTNET       9.12.4.1     UGS        0000000000  OSA2120LNK
9.12.4.0/22       0.0.0.0      US         0000000001  OSA2100LNK
9.12.4.0/22       0.0.0.0      US         0000000000  OSA2120LNK
9.12.4.59/32      0.0.0.0      UH         0000000000  OSA2100LNK
9.12.4.77/32      0.0.0.0      UH         0000000000  OSA2120LNK
9.12.9.162/32     0.0.0.0      H          0000000000  CTC4A21
127.0.0.1/32      0.0.0.0      UH         0000000001  LOOPBACK
192.168.50.4/32   0.0.0.0      H          0000000000  EZASAMEMVS
192.168.50.4/32   0.0.0.0      UH         0000000000  EZAXCF$A
192.168.50.6/32   0.0.0.0      UHS        0000000000  EZAXCF$A
201.2.10.12/32    0.0.0.0      UH         0000000000  VIPLC9020A0C
```

5. The output does not show any 10.x.x.x addresses. This indicates that we have no network interface with 10.x.x.x. However, that address might be reachable through our default gateway. So we try to ping 10.1.110.3 from somewhere else in the 9.x.x.x network (my workstation). The ping times out.

It appears that no router in the network knows how to get to the subnet where 10.1.110.3 resides.

6. However, we know that one of the other systems in the same sysplex is able to successfully connect to the IBM zAware LPAR using that address. So on that system we display its routing table using the **D TCPIP, ,NETSTAT,ROUTE** command. The output is shown in Example D-3.

Example D-3 Output from D TCPIP,NETSTAT,ROUTE command on the other system

```
EZZ2500I NETSTAT CS V1R13 TCPIP 021
DESTINATION      GATEWAY      FLAGS      REFCNT      INTERFACE
DEFAULTNET       9.12.4.1     UGS        0000000000  OSA2040LNK
DEFAULTNET       9.12.4.1     UGS        0000000000  OSA2060LNK
9.12.4.0/22       0.0.0.0      US         0000000000  OSA2040LNK
9.12.4.0/22       0.0.0.0      US         0000000000  OSA2060LNK
9.12.4.58/32      0.0.0.0      UH         0000000000  OSA2040LNK
9.12.4.110/32     0.0.0.0      UH         0000000000  OSA2060LNK
9.12.9.164/32     0.0.0.0      H          0000000000  CTC4A31
10.1.110.0/24     0.0.0.0      US         0000000001  IUTIQDF4LNK
10.1.110.2/32     0.0.0.0      UH         0000000000  IUTIQDF4LNK
127.0.0.1/32      0.0.0.0      UH         0000000018  LOOPBACK
192.168.50.0/24   0.0.0.0      US         0000000000  IQDIOLNKC0A83206
192.168.50.4/32   0.0.0.0      UHS        0000000000  EZAXCF$2
192.168.50.6/32   0.0.0.0      H          0000000000  EZASAMEMVS
192.168.50.6/32   0.0.0.0      UH         0000000000  IQDIOLNKC0A83206
192.168.50.6/32   0.0.0.0      UH         0000000000  EZAXCF$2
201.2.10.13/32    0.0.0.0      UH         0000000000  VIPLC9020A0D
201.2.10.200/32   0.0.0.0      UH         0000000000  VIPLC9020AC8
```

7. On this system, we see that there is a route to the 10.1.110.0/24 network and the interface is IUTIQDF4LNK. So we display this interface on using the **D TCPIP, ,NETSTAT,DEVLINKS,INTFN=IUTIQDF4LNK** command. The output is shown in Example D-4.

Example D-4 Output from D TCPIP,NETSTAT,DEVLINKS,INTFN=IUTIQDF4LNK command

```

EZZ2500I NETSTAT CS V1R13 TCPIP 039
DEVNAME: IUTIQDF4          DEVTYPE: MPCIPA
DEVSTATUS: READY          CFGROUTER: NON  ACTROUTER: NON
LNKNAME: IUTIQDF4LNK      LNKTYPE: IPAQIDIO  LNKSTATUS: READY
  IPBROADCASTCAPABILITY: NO
  ARPOFFLOAD: YES          ARPOFFLOADINFO: YES
  ACTMTU: 8192
  VLANID: NONE
  READSTORAGE: GLOBAL (2048K)
  SECCLASS: 255            MONSYSPLEX: NO
  IQDMULTIWRITE: DISABLED
ROUTING PARAMETERS:
  MTU SIZE: 8192           METRIC: 00
  DESTADDR: 0.0.0.0        SUBNETMASK: 255.255.255.0
MULTICAST SPECIFIC:
  MULTICAST CAPABILITY: YES
GROUP          REFCNT      SRCFLTMD
-----
224.0.0.1      0000000001  EXCLUDE
  SRCADDR: NONE
LINK STATISTICS:
  BYTESIN                      = 298990
  INBOUND PACKETS              = 3559
  INBOUND PACKETS IN ERROR     = 0
  INBOUND PACKETS DISCARDED    = 0
  INBOUND PACKETS WITH NO PROTOCOL = 0
  BYTESOUT                    = 2536839
  OUTBOUND PACKETS            = 4708
  OUTBOUND PACKETS IN ERROR   = 0
  OUTBOUND PACKETS DISCARDED  = 0

```

8. Knowing that system has a connection, we check the TCP PROFILE data set for that system. The TCPIP JCL points to SYS1.#@\$A.TCPPARMS(PROFILE).

Checking this member for the LNKNAME listed in the output (IUTIQDF4LNK), we find the following:

```

DEVICE IUTIQDF4 MPCIPA NOAUTORESTART
LINK IUTIQDF4LNK IPAQIDIO IUTIQDF4

```

9. We look up this statement in “DEVICE and LINK - MPCIPA HiperSockets devices statement” of the *z/OS Communications Server IP Configuration Reference*, SC31-8776. Here we find the following information about the statement:

```

DEVICE--device_name--MPCIPA
device_name

```

The name of the device must use the following convention:

Prefix is IUTIQD.

Suffix xx [hexadecimal value (00x - FFx) of the corresponding IQD CHPID]. This value cannot conflict with the IQD CHPID used for dynamic XCF.

10. We now know that F4 is the HIPERSOCKETS CHPID being used by the system that *is* able to connect. Using a **D M=CHP(F4)** command on the system that cannot connect, we see that F4 is not defined to that system, as shown in Example D-5.

Example D-5 Output from D M=CHP command

```
IEE174I 10.54.03 DISPLAY M 747
CHPID F4: TYPE=00, DESC=UNKNOWN, OFFLINE
DEVICE STATUS FOR CHANNEL PATH F4
CHP=F4 DOES NOT EXIST
***** SYMBOL EXPLANATIONS *****
+ ONLINE      @ PATH NOT VALIDATED  - OFFLINE      . DOES NOT EXIST
* PHYSICALLY ONLINE  $ PATH NOT OPERATIONAL
```

11. Checking the IBM zAware HMC firmware definition, we see that it has a channel definition for F4. But when checking the HMC, we realize that the system that is able to connect is on the same CPC as the IBM zAware partition, but that the other system is on a different CPC. Therefore, the other system could not use a Hipersocket connection (unless it was extended through the IEDN network).
12. To connect the system with the problem to an IBM zAware partition, we have to use an OSA connection from this LPAR to the IBM zAware OSA interface.

In this case, the problem turned out to be that both systems were using the same IXGCNFxx member, but the SERVER value in that member was not valid for the failing system.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *Exploiting the IBM Health Checker for z/OS Infrastructure*, REDP-4590
- ▶ *IBM z/OS V1R13 Communications Server TCP/IP Implementation: Volume 4 Security and Policy-Based Networking*, SG24-7999
- ▶ *Systems Programmer's Guide to: z/OS System Logger*, SG24-6898
- ▶ *S/390 Parallel Sysplex: Resource Sharing*, SG24-5666
- ▶ *z/OS Management Facility*, SG24-7851
- ▶ *z/OS Version 1 Release 12 Implementation*, SG24-7853
- ▶ *Building an Ensemble Using IBM zEnterprise Unified Resource Manager*, SG24-7921
- ▶ *z/OS Version 1 Release 13 Implementation*, SG24-7946

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following web site:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- ▶ *IBM Health Checker for z/OS V1R13*, SA22-7994
- ▶ *z/OS MVS Setting Up a Sysplex*, SA22-7625
- ▶ *z/OS MVS Systems Commands*, SA22-7627
- ▶ *z/OS UNIX System Services Programming: Assembler Callable Services Reference*, SA22-7803
- ▶ *z/OS V1R13.0 UNIX System Services Messages and Codes*, SA22-7807
- ▶ *IBM System z Advanced Workload Analysis Reporter (IBM zAware) Guide*, SC27-2623
- ▶ *z/OS Communications Server IP Configuration Guide*, SC31-8775
- ▶ *z/OS Communications Server IP Configuration Reference*, SC31-8776
- ▶ *z/OSMF Configuration Guide*, SA38-0652
- ▶ *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- ▶ *z/OS Communications Server Diagnosis Guide*, GC31-8782
- ▶ *z/OS Problem Management*, G325-2564

Online resources

These web sites are also relevant as further information sources:

- ▶ Message Analysis Program (search for MSGLG610)
<http://www-03.ibm.com/systems/z/os/zos/features/unix/bpxalty2.html>
- ▶ IBM Journal of Research and Development, *Reliability, Availability, and Serviceability of IBM Computer Systems: A Quarter Century of Progress*, Volume 25, No. 5, September 1981
<http://domino.research.ibm.com/tchjr/journalindex.nsf/9fe6a820aae67ad785256547004d8af0/6cb039c60ac4527a85256bfa0067f4d3!OpenDocument>

The following blogs by members of the IBM zAware development and test teams provide additional, current information that might be helpful:

- ▶ The journey of IBM zAware
https://www-304.ibm.com/connections/blogs/systemz/entry/zaware?lang=en_us
- ▶ IBM zAware installation and setup
https://www-304.ibm.com/connections/blogs/systemz/entry/zaware_installation?lang=en_us
- ▶ Top 10 Most Frequently Asked Questions About IBM zAware
https://www-304.ibm.com/connections/blogs/systemz/entry/zawarefaq?lang=en_us
- ▶ IBM zAware demo video
https://www-304.ibm.com/connections/blogs/systemz/entry/zawaredemo?lang=en_us

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



Extending IBM z/OS System Management Functions with IBM zAware

(0.2"spine)
0.17"<->0.473"
90<->249 pages



Extending IBM z/OS System Management Functions with IBM zAware



Redbooks®

Gain an understanding of the role of IBM zAware

Learn about IBM zAware planning and implementation

Benefit from IBM zAware capabilities

This IBM Redbooks publication explains the capabilities of the IBM System z Advanced Workload Analysis Reporter (IBM zAware), and shows how you can use it as an integral part of your existing System z management tools.

IBM zAware is an integrated, self-learning, analytics solution for IBM z/OS that helps identify unusual system behavior in near real time. It is designed to help IT personnel improve problem determination so they can restore service quickly and improve overall availability.

The book gives you a conceptual description of the IBM zAware appliance. It will help you to understand how it fits into the family of IBM mainframe system management tools that include Runtime Diagnostics, Predictive Failure Analysis (PFA), IBM Health Checker for z/OS, and z/OS Management Facility (z/OSMF).

You are provided with the information you need to get IBM zAware up and running so you can start to benefit from its capabilities immediately. You will learn how to manage an IBM zAware environment, and see how other products can use the IBM zAware Application Programming Interface to extract information from IBM zAware for their own use. The target audience includes system programmers, system operators, configuration planners, and system automation analysts.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-8070-00

ISBN 0738437727