Detecting and Diagnosing Problems when z/OS “Thinks” it is Running Okay

z/OS Soft Failure Detection, Avoidance, Diagnosis

Bob Abrams
IBM Corporation
Poughkeepsie, NY

June 10, 2013

Many thanks to Karla Arndt
Trademarks

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

Not all common law marks used by IBM are listed on this page. Failure of a mark to appear does not mean that IBM does not use the mark nor does it mean that the product is not actively marketed or is not significant within its relevant market.

Those trademarks followed by ® are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

*, AS/400®, e business(logo)®, DBE, ESCO, eServer, FICON, IBM®, IBM (logo)®, iSeries®, MVS, OS/390®, pSeries®, RS/6000®, S/30, VM/ESA®, VSE/ESA, WebSphere®, xSeries®, z/OS®, zSeries®, z/VM®, System i, System i5, System p, System p5, System x, System z, System z9®, BladeCenter®

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

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

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

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the 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.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.
Agenda:
Detecting and Diagnosing Problems when z/OS “Thinks” it is Running Okay

- Soft Failure detection & avoidance
  - Provided at multiple levels of the stack
  - Types of problems handled by each type of soft failure detection

- Soft Failure Detect/Diagnose/Avoid Capabilities in z/OS
  - **Detection:** z/OS Components
  - **Avoidance:** Health Checks
  - **Detection & diagnosis:** PFA, Runtime Diagnostics, zAware
  - **Business Application view:** Systems Management products

- Lessons learned on reducing impact of soft failures

*All elements work together for an integrated IBM solution approach*
Types of Failures on System z and z/OS

**Masked Failure**

- Software/Hardware detects failure
- Software/Hardware corrects failure
- No impact to business
- Example: Hardware power supply failure: switch to alternate, IBM alerted, concurrent replacement

**Hard Failure**

- Software/Hardware detects failure
- Automations and operations restart the failing component
- Minimal impact to business
- Example: Application terminates but is restarted by ARM

**Soft Failure**

- User detects failure, impact to business
- Difficult to determine recovery actions
- Example: component is failing, holds resources (locks, enqueues) required by other components, causes sysplex wide stall, leads to sysplex wide IPL
The Issue with Soft Failures

“Your systems don’t break. They just stop working and we don’t know why.”

“Sick, but not dead” or Soft failures

### Symptoms

- 80% of the business impact, but only 20% of the problems
- Long duration
- Infrequent
- Unique
- In software or hardware
- Cause creeping failures and sympathy sickness
- Hard to determine how to isolate
- Hard to determine how to recover
- Hard for software to detect internally
- Probabilistic, not deterministic

### Manifested as

- Stalled or hung processes
  - Single system or sysplex members
  - Sympathy sickness
- Resource contention
- Storage growth
- CF, CDS growth
- I/O issues (paths, response times)
- Repetitive errors
- Queue growth
- Configuration issue
  - Single point of failure, threshold, cache structure size, not enabling new features

© 2012, 2013 IBM Corporation
Customer Pain Points:

- Fault occurs long before anyone notices
- Difficult to identify where the problem is coming from
  → Leads to long decision time before recovery actions
Early Detection: Outage Detection and Avoidance

Soft Failure detection in z/OS components
- Detect soft failures as close to the source as possible
  Thresholds set by installation

IBM Health Checker for z/OS
- Keep subtle configuration errors from resulting in Soft Failures
  Emit Alerts

Predictive Failure Analysis
- Machine Learning - Convert diagnostic data to knowledge in real time
  Convert soft failures to correctable incidents
Earlier Problem Diagnosis: Reduce Decision Time

Runtime Diagnostics:
* Machine-speed understanding
  * Enables faster, identifies the culprit, correct recovery actions

zAware
* Anomaly detection based on pattern recognition techniques that examine the health of the system
z/OS Problem Avoidance, Detection, Diagnosis and Recovery

IBM Tivoli Omegam on XE for z/OS
IBM Tivoli Systems Automation
IBM Tivoli Netcool/OMNibus
IBM zAware
IBM z/OSMF
Consoles and OPERLOG
Runtime Diagnostics
IBM Health Checker for z/OS
Predictive Failure Analysis

z/OS components
(System resources & structures … Single system, Sysplex)

Problem Avoidance
Problem Detection
Problem Diagnosis
Problem Recovery
Auto IPL BCPii & SSDPP
Message Flooding SMF Record Flooding

© 2012,2013 IBM Corporation
Detection of Soft Failures by z/OS Components

- z/OS attempts to detect soft failures as close to the source as possible
  - Uses the least amount of resources
  - Requires the smallest amount of the stack to do detection

- Detection of a soft failure requires ability to identify when something is wrong
  - Thresholds set by the installation

- *Whenever possible, components try to avoid soft failures*

- Examples …
## Component Examples: Detection, Identification of Soft Failures

<table>
<thead>
<tr>
<th>Component</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRS</td>
<td>Enhanced contention analysis for ENQ, Latch</td>
</tr>
<tr>
<td></td>
<td>GRS Latch Identify string</td>
</tr>
<tr>
<td></td>
<td>WLM management of blocking units</td>
</tr>
<tr>
<td>UNIX System Services</td>
<td>Latch identity exploitation</td>
</tr>
<tr>
<td></td>
<td>XCF communication improvements (R13)</td>
</tr>
<tr>
<td></td>
<td>System limits</td>
</tr>
<tr>
<td></td>
<td>D OMVS, WAITERS to diagnose file system latch contention</td>
</tr>
<tr>
<td>JES2</td>
<td>JES2 Monitor</td>
</tr>
<tr>
<td>IOS</td>
<td>Missing Interrupt Handler</td>
</tr>
<tr>
<td></td>
<td>Identify systems sharing a reserve</td>
</tr>
<tr>
<td></td>
<td>Captured UCB protection</td>
</tr>
<tr>
<td></td>
<td>I/O timing facility</td>
</tr>
<tr>
<td></td>
<td>Detect &amp; remove “flapping links”</td>
</tr>
<tr>
<td></td>
<td>Dynamic Channel Path Management</td>
</tr>
<tr>
<td>DFSMS</td>
<td>CAS contention detection</td>
</tr>
<tr>
<td></td>
<td>VSAM RLS index traps</td>
</tr>
<tr>
<td></td>
<td>Media Manager</td>
</tr>
</tbody>
</table>
## Component Examples: Detection of soft failures...

<table>
<thead>
<tr>
<th>Component</th>
<th>Features</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCF / XES</td>
<td>Stalled member support</td>
<td>Identify unresponsive system, restore to normal operation OR remove it to avoid sympathy sickness</td>
</tr>
<tr>
<td></td>
<td>Exploitation of BCPII to determine dead system more quickly</td>
<td>Avoid waiting the Failure Detection Interval (FDI) if the system is truly dead ... detect &amp; reset failed system, eliminate data corruption, avoid sympathy sickness.</td>
</tr>
<tr>
<td></td>
<td>Sysplex Failure Management, scenarios</td>
<td>• Not updating status, Not sending signals (ISOLATETIME(0): Fencing initiated n seconds after FDI exceeded)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• System updating status, not sending signals (Loss of connectivity: CONNFAIL(YES): remove systems with low weights)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• System Not Updating Status, But IS Sending Signals (SSUMLIMIT(900)) ... length of time system can remain not updating heartbeat (semi-sick), but sending signals)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Sysplex Member Stalled (MEMSTALLTIME ... break out of of an XCF signaling jam by removing the largest build-up)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Structure Hang conditions ... Take action when connector does not respond, avoiding user hangs (CFSTRHANGTIME) (R12)</td>
</tr>
<tr>
<td></td>
<td>Critical Member support; GRS exploitation (R12)</td>
<td>If a critical member is “impaired” for long enough, XCF will eventually terminate the member; GRS: remove system</td>
</tr>
</tbody>
</table>

Details in backup section
IBM Health Checker for z/OS
Soft Failure Avoidance

- Health checker’s role is to keep subtle configuration errors from resulting in Soft Failures
  - Performance
  - System effects
  - Check configuration for best practices
  - Single points of failure for log structures, data sets, CDS
  - Storage utilization, running out of resources
  - How many ASIDs do I have left? LXs? When will I run out?
  - Whether DAE is inactive
  - VSAM RLS latch contention, CF Cache size, CDS SPOF, etc.
  - System Logger structure usage
  - I/O timing, protection
  - ...

- Also used to emit PFA alerts
  - Warnings of detected soft failures

- 187 z/OS Health Checks in z/OS R13 (plus ISVs)
  - IBM Health Checker for z/OS User’s Guide
    http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/e0z2l161/CCONTENTS?SHELF=all13be9&DN=SA22-7994-13&DT=20120814100820
  - List of Health Checks:
    http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/e0z2l161/3.1?SHELF=all13be9&DT=20120814100820
## Health Checker: Soft Failure avoidance examples

<table>
<thead>
<tr>
<th>Component</th>
<th>Health Check</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCF</td>
<td>XCF_CDS_SPOF</td>
<td>Evaluate primary &amp; secondary CDS configuration to determine if Sysprog inadvertently created a single point of failure</td>
</tr>
<tr>
<td></td>
<td>XCF_SFM_SUM_ACTION</td>
<td>Check ISOLATETIME value, to allow SFM to fence and partition a system without operator intervention and without undue delay.</td>
</tr>
<tr>
<td></td>
<td>XCF_SFM_SUMLIMIT</td>
<td>Checks status update missing (SUMLIMIT) value</td>
</tr>
<tr>
<td></td>
<td>XCF_SFM_ACTIVE</td>
<td>Verifies SFM active, policy values</td>
</tr>
<tr>
<td></td>
<td>XCF_SFM_CFSTRHANGTIME</td>
<td>Verifies CFSTRUCTURE hang time</td>
</tr>
<tr>
<td></td>
<td>XCF_SFM_CONNFAIL</td>
<td>Threshold for loss of connectivity</td>
</tr>
<tr>
<td>RACF</td>
<td>RACF_GRS_RNL</td>
<td>Evaluates whether the RACF ENQ names are in a GRSRNL list: system exclusion resource name list (SERNL) or the system inclusion resource name list (SIRNL)</td>
</tr>
<tr>
<td>UNIX System Services</td>
<td>USS_PARMLIB</td>
<td>Validate current system against parmlib IPL’d with</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remind you to update parmlib (due to dynamic changes)</td>
</tr>
<tr>
<td></td>
<td>USS_CLIENT_MOUNTS</td>
<td>With Sysplex, some file systems accessed locally, some of function shipped to the File system owner. Some are accessed locally, but are configured to function ship</td>
</tr>
</tbody>
</table>

*Details in backup section*
IBM zAware

Console and OPERLOG

IBM Health Checker for z/OS

Predictive Failure Analysis

Runtime Diagnostics

z/OS Components
System resource and control blocks
Single system or Sysplex-scope

Early Detection ➔ Outage Avoidance, PD

Problem Diagnosis

Problem Avoidance

Problem Detection
Predictive Failure Analysis

- **Causes of soft failures**
  - *Damaged systems*
    - Recurring or recursive errors anywhere in the software stack
  - Serialization
    - Priority inversion, classic deadlocks, owner gone
  - *Resource exhaustion*
    - Physical and software resources
  - Indeterminate or unexpected states

- **PFA uses**
  1. Historical data from each LPAR
  2. Machine learning and mathematical modeling
     - To *detect and alert* you to *abnormal behavior* and its potential causes
     - To *enable you* to convert soft failures to a *correctable incident*
Soft Failures: Hypothetical IT Example

1. A transaction --
   - that has worked for a long time starts to fail, or
   - occasionally (yet, rarely) fails
   - Example – “Reset Password and send link to registered email account”

2. The transaction starts failing more regularly

3. Recovery is successful –
   - Such that the overall, applications continue to work
   - Generates burst of WTO’s, SMF records and LOGREC entries

4. BUT, THEN! Multiple, failing transactions occur together on a heavily loaded system
   - Recovery occurs
   - Slows down transaction processor
   - Random timeouts of other transactions occur
   - System becomes “sick, but not dead”

Time period when everything running OK. PFA sees problem internally. Problem seen externally

This is a hypothetical problem which is a combination of multiple actual problems
PFA → Resource Exhaustion

- Projects *current trend into the future*
- One metric: Common storage exhaustion

![Common Storage Usage Check](chart)

- Capacity = 708
- Current usage = 644
- Prediction = 759

Exception issued when current trend was at 91% of capacity.
PFA ➔ Damaged Systems

- Does *trend analysis* and models behavior to create *expected* value or rate

- Determines workload change vs. abnormal condition
  - Clusters current and past trends
  - Uses multiple models over time ranges when needed
    - 1 hour, 24 hours, 7 days
  - Uses CPU normalization and calculations when needed
    - Rates = Metric / CPU

- Five metrics
  - LOGREC arrival rate, Message arrival rate, SMF arrival rate, Enqueue request rate, JES spool usage

Abnormal Behavior
What happens when PFA detects a problem?

- **Health check exception** written to console
  - New exceptions suppressed until new model is available

- **Prediction report** available in SDSF (s.ck)
  - “Top address spaces” = potential villains
  - Address spaces causing exception
  - Current and predicted values provided
  - Reports also available when no problem occurs

- **Modeling automatically runs** more frequently

- **Logs and data files** for service written to EXC_{timestamp} directories

- **Best practices and more information** in z/OS Problem Management
## Example report: Message Arrival Rate

### 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:

<table>
<thead>
<tr>
<th>Job Name</th>
<th>ASID</th>
<th>Message Arrival Rate</th>
<th>Predicted Message Arrival Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRACKED1</td>
<td>001D</td>
<td>58.00</td>
<td>23.88</td>
</tr>
<tr>
<td>TRACKED2</td>
<td>0028</td>
<td>11.00</td>
<td>0.34</td>
</tr>
<tr>
<td>TRACKED3</td>
<td>0029</td>
<td>11.00</td>
<td>12.43</td>
</tr>
</tbody>
</table>

...
Example report: Common Storage Usage

- **Top predicted users**
  - Tries to pinpoint potential villains
  - Those whose usage has *increased* the most in the last hour

- **Other information**
  - Expansion information
  - IBM Health Checker for z/OS message in its entirety

### Common Storage Usage Prediction Report

<table>
<thead>
<tr>
<th>Storage Location</th>
<th>Current Usage in Kilobytes</th>
<th>Prediction in Kilobytes</th>
<th>Capacity When Predicted in Kilobytes</th>
<th>Percentage of Current to Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>*CSA</td>
<td>2796</td>
<td>3152</td>
<td>2956</td>
<td>95%</td>
</tr>
<tr>
<td>SQA</td>
<td>455</td>
<td>455</td>
<td>2460</td>
<td>18%</td>
</tr>
<tr>
<td>CSA+SQA</td>
<td>3251</td>
<td>3771</td>
<td>5116</td>
<td>64%</td>
</tr>
<tr>
<td>ECSA</td>
<td>114922</td>
<td>637703</td>
<td>512700</td>
<td>22%</td>
</tr>
<tr>
<td>ESQA</td>
<td>8414</td>
<td>9319</td>
<td>13184</td>
<td>64%</td>
</tr>
<tr>
<td>ECSA+ESQA</td>
<td>123336</td>
<td>646007</td>
<td>525884</td>
<td>23%</td>
</tr>
</tbody>
</table>

* = Storage locations that caused the exception.

Storage requested from SQA expanded into CSA and is being included in CSA usage and predictions. Comparisons for SQA are not being performed.

### Address spaces with the highest increased usage:

<table>
<thead>
<tr>
<th>Job Name</th>
<th>Storage Location</th>
<th>Current Usage in Kilobytes</th>
<th>Predicted Usage in Kilobytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOB3</td>
<td>*CSA</td>
<td>1235</td>
<td>1523</td>
</tr>
<tr>
<td>JOB1</td>
<td>*CSA</td>
<td>752</td>
<td>935</td>
</tr>
<tr>
<td>JOB5</td>
<td>*CSA</td>
<td>354</td>
<td>420</td>
</tr>
<tr>
<td>JOB8</td>
<td>*CSA</td>
<td>152</td>
<td>267</td>
</tr>
<tr>
<td>JOB2</td>
<td>*CSA</td>
<td>75</td>
<td>80</td>
</tr>
<tr>
<td>JOB6</td>
<td>*CSA</td>
<td>66</td>
<td>78</td>
</tr>
<tr>
<td>JOB15</td>
<td>*CSA</td>
<td>53</td>
<td>55</td>
</tr>
<tr>
<td>JOB18</td>
<td>*CSA</td>
<td>42</td>
<td>63</td>
</tr>
<tr>
<td>JOB7</td>
<td>*CSA</td>
<td>36</td>
<td>35</td>
</tr>
<tr>
<td>JOB9</td>
<td>*CSA</td>
<td>31</td>
<td>34</td>
</tr>
</tbody>
</table>

* = Storage locations that caused the exception.
The PFA Checks and Enhancements

- **z/OS 1.10 SPE**
  - Common storage exhaustion check
    - CSA+SQA → below the line
    - ECSA+ESQA above the line
  - LOGREC arrival rate check
    - Groups arrivals by key
    - Four time ranges

- **z/OS 1.11**
  - Message Arrival Rate check
    - (WT0 + WT0R) / CPU = Rate
    - Tracked, persistent address spaces
    - Other persistent address spaces
    - Non-persistent address spaces
    - Total system
  - Frames and Slots usage check
    - Removed from all releases in 2012 with OA40065

- **z/OS 1.12**
  - SMF arrival rate check
    - SMF arrival count / CPU = Rate
    - Same categories as Message Arrival Rate
  - Common storage exhaustion = 6 locations
  - Dynamic modeling improvements
  - Supervised learning (exclude jobs)
  - Performance and serviceability

- **z/OS 1.13**
  - JES spool usage check
    - JES2 only
    - Tracks all persistent address spaces
  - Enqueue request rate check
    - Tracked, persistent address spaces
    - Total system
  - Integration with Runtime Diagnostics to detect “too low”
Dealing with Soft Failures when they occur

- Systems running, but seem impaired
  - Slow
  - Applications "not working"
  - Sysplex problems
  - UNIX problems
  - Irate users

**Action!**: Check out the systems for cause
- Outstanding contention?
- Critical errors detected?
- Important tasks looping?
- I/O bottlenecks?
- Network issues?

- Prep for "Bridge" call
- Is it fixed yet?

- Investigate the clues using monitors and commands

- Get diagnostic data such as dumps and command output

- Restart the system
  - Open a PMR
  - Fix your resume/CV
Runtime Diagnostics Usage and Benefits

- Performs diagnostics on a “sick, but not dead” system in a timely manner
  - Performs analysis similar to that of a very experienced system programmer
  - But, more comprehensive and faster – goal of 60 seconds or less
  - Looks for specific evidence of “soft failures” occurring RIGHT NOW!

- Runtime Diagnostics…
  - Is not automation or a monitor
  - Takes no corrective action, but recommends next steps
    - Which jobs to cancel
    - What to investigate further such as classes of resources or a single address space using a monitor like RMF or Tivoli Omegamon
  - Has no background processing and minimal dependencies on system services

- Use Runtime Diagnostics…
  - when the help desk or operations reports a problem on the system
  - to get ready for the “bridge call”
  - when PFA detects abnormal behavior
Runtime Diagnostics Analysis Types

Component Analysis
- Component problems that emit critical messages in OPERLOG

Global Resource Contention
- ENQ contention for system address spaces
- GRS latch contention (R13)
- z/OS file system latch contention (R13)

Address Space Execution
- Address spaces using high CPU
- Address spaces appearing to be in a TCB-enabled loop
- Address spaces with high local lock suspension rate
Runtime Diagnostics Invocation → R13

**Bold = Difference from R12**

- **Start Address Space**
  - START HZR, SUB=MSTR

- **Modify Command to run analysis**
  - F HZR, ANALYZE
  - F HZR, MODIFY, OPTIONS=(SYSNAME=SYSB)

- Can specify another system on modify command, but only ENQ and OPERLOG scans done

- z/OS R13 pgm is HZRINIT
  - HZR PROC → PGM=HZRINIT

- Output can be written to physical sequential dataset
  - HZROUT DD DUMMY
  - or
  - HZROUT DD <sequential dataset>

© 2012, 2013 IBM Corporation
Success ➞ Events detected

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: 02 - PRIORITIES: HIGH:02 MED:00 LOW:00
TYPES: HIGHCPU:01
TYPES: LOOP:01

EVENT 01: HIGH - HIGHCPU - SYSTEM: SY1 2010/12/21 - 13:51:33
ASID CPU RATE:99% ASID:002E JOBNAME:IBMUSERX
STEPNAME:STEP1 PROCSTEP: JOBID:JOB000045 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 02: HIGH - LOOP - SYSTEM: SY1 2010/12/21 - 13:51:14
ASID:002E JOBNAME:IBMUSERX TCB:004FF1C0
STEPNAME:STEP1 PROCSTEP: JOBID:JOB000045 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.

Example: When both a HIGHCPU event and a LOOP event are detected for the same system the job is very likely looping.
• The normal corrective action is to cancel the job
Runtime Diagnostics Events

z/OS 1.12

• Component-specific, critical messages in OPERLOG
  • Looks one hour back, if available
  • Additional analysis for some msgs
  • Message summary found in output
  • Can analyze messages in other systems in sysplex

• Enqueue Contention Checking
  • Looks for system address space waiting > 5 seconds
  • Lists both waiter and blocker
  • Can detect contention in other system in sysplex

• Local Lock Suspension
  • Any address space whose local lock suspension time is > 50%

z/OS 1.12 (continued)

• CPU Analysis
  • Takes 2 samples over 1 sec. interval
  • Any task using > 95% is considered a potential problem

• Loop Detection
  • Investigates all tasks in all address spaces looking for TCB loops

z/OS 1.13

• z/OS UNIX Latch Contention
  • Looks for z/OS UNIX latch contention or waiting threads that exit for > 5 minutes.

• GRS Latch Contention
  • Obtains latch contention info from GRS
  • Omits z/OS UNIX file system latch contention
  • Returns longest waiter for each latch set

Runtime Diagnostics is simple to set up, simple to use, and ready to go when you need it most!
PFA and Runtime Diagnostics Integration

Runtime Diagnostics

Get events if rate seems low

Predictive Failure Analysis

Get data from system

Comparisons

Use predictions and current usage values in statistical algorithms

Predictions
z/OS UNIX file system

z/OS Components
System resource and control blocks
Single system or Sysplex-scope
PFA and Runtime Diagnostics Integration

- “Too low” exception message sent as WTO by default
- Runtime Diagnostics output included in PFA report
- Prediction report and result message available in SDSF (sdsf.ck)
- PFA current rates and predictions relevant to category causing exception

```plaintext
Message Arrival Rate Prediction Report
(Heading information intentionally omitted.)

Persistent address spaces with low rates:

<table>
<thead>
<tr>
<th>Job Name</th>
<th>ASID</th>
<th>Message Arrival Rate</th>
<th>1 Hour</th>
<th>24 Hour</th>
<th>7 Day</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOBS4</td>
<td>001F</td>
<td>1.17</td>
<td>23.88</td>
<td>22.82</td>
<td>15.82</td>
</tr>
<tr>
<td>JOBS5</td>
<td>002D</td>
<td>2.01</td>
<td>8.34</td>
<td>11.11</td>
<td>12.11</td>
</tr>
</tbody>
</table>

Runtime Diagnostics Output:

Runtime Diagnostics detected a problem in job: JOBS4
EVENT 06: HIGH - HIGHCPU - SYSTEM: SY1 2009/06/12 - 13:28:46
ASID CPU RATE: 96% ASID: 001F JOBNAME: JOBS4
STEPNAME: PFATEST PROCSTEP: PFATEST JOBID: STC00042 USERID:

+++
JOBS4

OBJSTART: 2009/06/12 - 13:28:35
Error: ADDRESS SPACE USING EXCESSIVE CPU TIME. IT MAY BE LOOPING.
Action: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
-------------

EVENT 07: HIGH - LOOP - SYSTEM: SY1 2009/06/12 - 13:28:46
ASID: 001F JOBNAME: JOBS4 TCB: 004E0050
STEPNAME: PFATEST PROCSTEP: PFATEST JOBID: STC00042 USERID:
+++

OBJSTART: 2009/06/12 - 13:28:35
Error: ADDRESS SPACE APPEARS TO BE IN A LOOP.
Action: USE YOUR SOFTWARE MONITORS TO INVESTIGATE THE ASID.
(Additional output intentionally omitted.)
```
IBM zAware:
Smarter Computing Needs Smarter Monitoring

- New technology based on machine learning developed by IBM Research

- Cutting edge pattern recognition techniques look at the health of a system to pinpoint deviations from the ‘norm’

- High speed analytics facilitates the ability to consume large quantities of message logs

- Improves problem diagnosis across a set of System z servers

- Speeds up the time to decide on appropriate corrective actions on problems before they get bigger

- Allow establishment of procedures to prevent reoccurrence

zAware’s capacity as a ‘watch dog’ can help to detect unusual behavior in near real time
IBM zAware – IBM System z Advanced Workload Analysis Analysis Reporter

- Monitors **z/OS OPERLOG** including all messages written to z/OS console, including ISV and application generated messages
- Detects things typical monitoring systems miss due to:
  - Message suppression (message too common) Useful for long-term health issues
  - Uniqueness (message not common enough) Useful for real-time event diagnostics
- Color coded easy to use GUI via web browsers
- Output can be queued up to existing monitoring systems.
- Early detection and focused diagnosis can help improve time to recovery
Inside IBM zAware

zAware Partition
Shipped as firmware with EC12

File System

Persistent Storage

zAware GUI

Customer network

View zAware results

Control zAware-specific knobs

Manage zAware Firmware partition

z/OS pieces
Shipped with z/OS v1.13 + PTF

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog

Operlog
Inside IBM zAware Analytics

- OPERLOG is processed per-system
- zAware recognizes any well-formed message IDs
  - including IBM and non-IBM products and customer applications
- zAware summarizes the common message text and records the occurrences
- zAware builds a **model** of normal behavior based on the last 90 days
  - Called “Training”
  - Automatically trains every 30 days
  - Can be forced manually
  - Customizable
  - Unusual days can be excluded from future models
- z/OS utility is used to load historical logs into zAware
Inside IBM zAware Analytics

- Real-time OPERLOG data is compared to the model
- Assigns a message anomaly score to indicate deviation from the model
  - Rare messages
  - Out of context from normal patterns
  - High counts
- Uses z/OS-specific knowledge to influence the scores
- Generates an interval anomaly score per 10 minute interval
  - Current interval is updated every 2 minutes
  - GUI shows number of unique message IDs (bar height)
  - GUI shows interval anomaly score (bar color)
- Drill down on interval shows the message scores
- XML output available via HTTP APIs
Analysis View

Analysis

The System Anomaly Scores graph shows message analysis data for each system in ten minute intervals. For each interval, the bar height indicates the number of unique message IDs. Click on an interval bar to access detailed message information. To view messaging analyses from other dates, use the date selector. To customize which systems are displayed, select from the available options.

Date:

Interval Anomaly scores may vary.

System | Anomaly Scores
--- | ---
SVPLEX4.C00 (UTC)-5
SVPLEX4.C05 (UTC)-5
SVPLEX4.C06 (UTC)-5
SVPLEX4.C08 (UTC)-5

Color shows anomaly score

Height shows number of unique message IDs

Anomaly score shows difference from normal patterns

Interval anomaly score key:

No Difference  |  Significantly Different

Zoom level:
Analysis View

Analysis

The System Anomaly Scores graph shows message analysis data for each system in ten minute intervals. For each interval, the bar height indicates the number of unique message ids. Click on an interval bar to access detailed message information. To view messaging analyses from other days, use the date selector. To customize which systems are displayed, click on the Analysis Source link.

Date: January 6, 2013

Interval Anomaly Scores by System

<table>
<thead>
<tr>
<th>System</th>
<th>Anomaly Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVPLEX4.C00</td>
<td></td>
</tr>
<tr>
<td>Time: 03:30 -- 03:40</td>
<td></td>
</tr>
<tr>
<td>Unique Msg Ids: 146</td>
<td></td>
</tr>
<tr>
<td>Anomaly Score: 100.0</td>
<td></td>
</tr>
<tr>
<td>SVPLEX4.C05</td>
<td></td>
</tr>
<tr>
<td>SVPLEX4.C06</td>
<td></td>
</tr>
<tr>
<td>SVPLEX4.C08</td>
<td></td>
</tr>
</tbody>
</table>

Timeline (UTC)

Zoom level:

[Image of zoom levels from 1 hr to 24 hrs]

Interval anomaly score key:

- No Difference
- 99.5
- 99.6 - 100
- 101

Significantly Different
Interval View

Several messages never seen in the model

Time Line shows occurrences within interval

Message ID is a link to LookAt

z/OS specific rules affect anomaly score
Identify unusual behavior quickly

- **Which z/OS image is having unusual message patterns?**
  - High score generated by unusual messages or message patterns
  - GUI shows all systems or selected subsets

- **Which subsystem or component is abnormal?**
  - Examine high-scoring messages

- **When did the behavior start?**
  - Current 10 minute interval or earlier?
  - Which messages are unusual?
  - How often did the message occur?
  - When did the messages start to occur?

- **Were similar messages issued previously**
  - Easily examine prior intervals or dates
Integration with other Systems Management products

- **z/OSMF**
  - Configure a new external link
  
  Administration > Links > Actions > New
  - Provide link name, SAF suffix, zAware GUI URL
  - Category – recommend Problem Determination
  - Define authority required to use the link

- **APIs**
  - Provides **XML** equivalent to GUI
    - Analysis page
    - Interval View page

- **HTTP GET/POST requests**
  - Connect and authenticate to IBM zAware server
    - UserID known as a zAware user (e.g. LDAP)
  
  - Retrieve analysis for a monitored client
    - LPAR Interval scores for date
    - INTERVAL Message scores for a 10-minute interval
Integration with other Systems Management products

- IBM Tivoli **NetView** for z/OS
  - Can use the APIs to get IBM zAware results
  - Sample programs are available from

  - Described in detail in the Redbook:
    - *Extending z/OS System Management Functions with IBM zAware*
    - The samples can be tailored to drive NetView message automation and raise alerts on anomaly score.

- IBM plans to leverage IBM zAware results in the Tivoli Integrated Service Management products

- Other products can exploit the XML format results
IBM zAware - Summary

- 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, detection 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
IBM provides an integrated solution approach to Avoiding, Detection, Diagnosing Soft Failures

- IBM Tivoli Omegamon XE for z/OS
- IBM Tivoli Systems Automations
- IBM Tivoli Netcool/OMNibus
- IBM zAware
- IBM z/OSMF
- Consoles and OPERLOG
- Runtime Diagnostics
- IBM Health Checker for z/OS
- Predictive Failure Analysis

- Problem Avoidance
- Problem Detection
- Problem Diagnosis
- Problem Recovery

- Auto IPL BCPii & SSDPP
- Message Flooding SMF Record Flooding

z/OS components
(System resources & structures … Single system, Sysplex)