Predictive Failure Analysis – Customer Experience

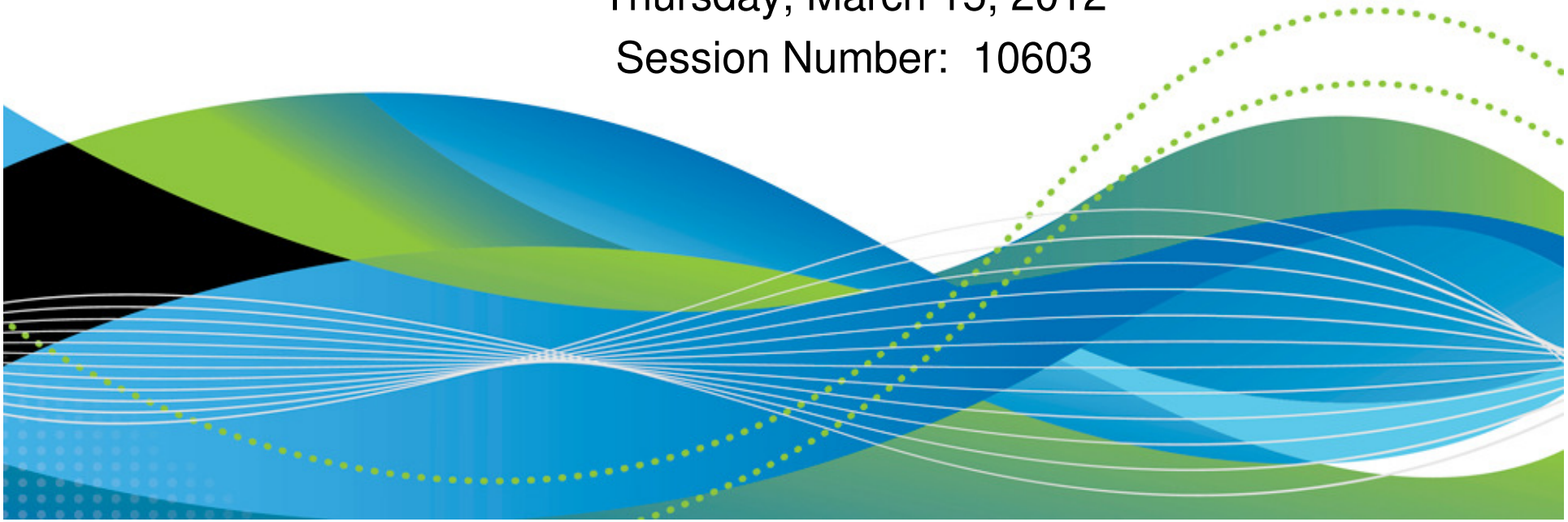
Sam Knutson, GEICO

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

Bob Abrams, IBM

Thursday, March 15, 2012

Session Number: 10603



z/OS Predictive Failure Analysis (PFA)

- **Health Checker runs checks supplied by IBM, vendors or written by you to detect existing or potential problems primarily with variation from understand successful practices**
- **GEICO has been running Health Checker for z/OS since the prototype and the first implementation in z/OS in 2005 in z/OS 1.7 base was optional download for z/OS 1.4 to 1.6**
- ***We run Health Checker all the time from IPL. You should too!***
- **PFA will attempt to detect problems which require analysis of history and current trends to detect. PFA can help you avoid soft failures.**
- **PFA was introduced with z/OS 1.10**
- **We run PFA all the time from IPL**

z/OS Predictive Failure Analysis (PFA)

- PFA checks have been delivered in each z/OS release starting with 1.10 currently for me on 1.12 I run these
  - PFA_COMMON_STORAGE_USAGE
  - PFA_FRAMES_AND_SLOTS_USAGE
  - PFA_LOGREC_ARRIVAL_RATE
  - PFA_MESSAGE_ARRIVAL_RATE
  - PFA_SMF_ARRIVAL_RATE
- IBM plans to extend PFA over time PFA so PFA will be useful today but like Health Checker your investment will be returned with interest in the future
- PFA uses the Health Checker for z/OS infrastructure but PFA checks are different and need to be treated differently
- *"Prediction is very difficult, especially about the future" Niels Bohr*

z/OS Predictive Failure Analysis (PFA)

- **PFA is constructed largely in Java and the majority of the CPU used by PFA is eligible to run on zAAP speciality engines**
- **APAR OA27495 NEW FUNCTION - ZAAP ON ZIIP ENHANCEMENT**
  - **ZZ=YES in IEASYSxx**
  - **zAAP on zIIP does not change in any way what is eligible for specialty processing just allows you to consolidate to have only one pool of specialty engines. If you have z/OS 1.11 or the PTF for APAR OA27495 applied to 1.10 or 1.9 AND you have only zIIP engines but NO zAAP engines AND you set a new operating system parameter ZZ=YES then zAAP eligible workloads will run on the zIIP alongside zIIP eligible workloads.**
- ***Using zAAP on zIIP is like buying colorsafe Cheer for your washing machine and being able to mix the lights and the darks☺***

Dealing with Health Checker Exceptions

- Resolution of a Health Checker exception frequently requires other subject matter experts cooperation
- Understanding of PFA exceptions and tuning has required recurring assistance from IBM Level-2 to adjust the proper “knob” correctly
- Don’t break existing system. Health Checker is reporting a potential problem don’t make an immediate visible one trying to hurry in a change.
- Avoid persistent outstanding exceptions it will lead to 'cry wolf syndrome' and Health Checks being disregarded.
- Running Health Checker and PFA and not looking at the output does not help you☹

Identifying Exceptions

- Automate notifications from Health Checker!
- We trap HZS* WTO using CA-OPSMVS
  - MLWTO avoid processing same event more than once
  - Consider “normal” error HZS0011E READING THE HZSPDATA DATA SET may want to just process HZS0001I, HZS0002E, HZS0003E, HZS0004I
  - EVENTUAL exceptions are emailed
  - CRITICAL exceptions are sent by email and pager
- Most well known SPOOL browsers and MVS monitors including IOF, SDSF, SYSVIEW, TMON, provide on-line interfaces to view and manage exceptions

Sending PFA documentation to IBM

- IBM Level-2 will typically need to see the data that is causing PFA to raise an exception
- Package the directory for the exception using the PAX command in batch
- PFA component ID is 5752SCPFA useful to know when opening a PMR

```
//UNPAX    EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*
//SYSEXEC  DD DSN=SYS1.SBPXEXEC,DISP=SHR
//SYSTSIN  DD *
  oshell cd /u/pfauser/PFA_SMF_ARRIVAL_RATE          ;        +
    pax -wvzf /tmp/PMR93649.499.000.BTST.AUG11.PFASAR.pax .
/*
```

Sending PFA documentation to IBM

- **FTP documentation to IBM this is an example using batch directly from z/OS but you may have different procedures**

```
//FTP   EXEC PGM=FTP,PARM='(EXIT'
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
testcase.software.ibm.com (timeout 720 exit=12
anonymous
you@yoursite.com
cd /toibm/mvs
lcd /tmp
SENDSITE
binary
put PMR93649.499.000.BTST.AUG11.PFASAR.pax
quit
/*
```

PFA : Installation

- RACF userid with OMVS segment we made it unique as doc specified and it's home directory is where PFA will store historical information
- Initially just added /pfauser to our primary USER file system. Most groups are segregated but odd users like ekmserv and others we have just created in /u. Once PFA had been running for a while saw zFS filling up and needing to be grown i.e. IOEZ00078E zFS aggregate OMVS.BTST.U.ZFS exceeds 95% full (2140/2250)
- Historical collection creates LOTS of small files which add up over time so a separate USS filesystem is a good idea.

PFA : Installation

- z/OS 1.12 added the automatic retention of past exception data. Up to 30 snapshots of data related to past exceptions saved each in a unique directory. This can be useful historical data for support.
- It can also contribute to the space use of the USS file system filling it up. This is another good reason to segregate PFA into its own file system.
- IOEZ00551I Aggregate OMVS.ASYS.PFA.ZFS ran out of space.
- IOEZ00445E Error extending OMVS.ASYS.PFA.ZFS.
- You may want to setup a CRON job or manually purge EXC_* directories (example EXC_20111106182) from the exception directories. Requirement to improve the purge and control retention has been voiced to IBM.
- APAR OA38786 (open with a ++APAR available) corrects a problem where file descriptors are left open is the root cause of space problem

PFA : Installation

- **On each system BEFORE starting PFA you need to copy the PROC from SYS1.SAMPLIB(AIRPROC) to your execution PROC. I used SYS1.PROCLIB(PFA)**
- **On each system BEFORE starting PFA you need to use the supplied script to create a directory structure**

```
$ cd /u/pfauser
$ su
# pwd
/u/pfauser
# sh /usr/lpp/bcp/AIRSHREP.sh
All existing data files and directories removed.
 Successfully created the Common Storage Usage Check Directory Structure.
 Successfully created the Logrec Arrival Rate Check Directory Structure.
 Successfully created and populated ini file for the Common Storage Usage Check.
 Successfully created and populated ini file for the Logrec Arrival Rate Check.
```

PFA : Installation zFS

- Allocated zFS and mounted for PFA. Must have RACF access to PFSCTL or UID 0 to format ZFS
- zSeries File System (zFS) PARMLIB(IOEPRM00) has aggrfull(90,5) and aggrgrow=on
- IDCAMS
  - DEFINE CLUSTER (NAME(OMVS.ASYS.PFA.ZFS) -
    LINEAR CYL(500,500) SHAREOPTIONS(3))
- IOEAGFMT
  - PARM=('-aggregate OMVS.ASYS.PFA.ZFS -compat')
- MOUNT FILESYSTEM('OMVS.ASYS.PFA.ZFS') MOUNTPOINT('/u/pfauser') TYPE(ZFS) MODE(RDWR)
- Don't forget to update SYS1.PARMLIB(BPXPRMxx)

```
MOUNT FILESYSTEM('OMVS.&SYSNAME..PFA.ZFS')
      MOUNTPOINT('/u/pfauser')   /* ZFS for /u/pfauser  */
        TYPE(ZFS)                /* Filesystem type ZFS   */
        MODE(RDWR)               /* Mounted  or read/write */
```

z/OS Predictive Failure Analysis (PFA)

- **PFA checks have had to be customized via HZSPRMxx updates to avoid spurious notifications in many cases relief later provided by APARs**

```
/* Predictive Failure Analysis Checks    */
ADDREPLACE POLICY STMT(PFA2) UPDATE
CHECK(IBMPFA,PFA_MESSAGE_ARRIVAL_RATE)
PARM='DEBUG(0) STDDEV(16) COLLECTINT(15) MODELINT(360)
COLLECTINACTIVE(1) TRACKEDMIN(0) EXCEPTIONMIN(5)'
REASON('less sensitive to avoid spurious trips')
DATE(20110221)

ADDREPLACE POLICY STMT(PFA4) UPDATE
CHECK(IBMPFA,PFA_MESSAGE_ARRIVAL_RATE)
PARM='DEBUG(0) STDDEV(3) COLLECTINT(15) MODELINT(720)
COLLECTINACTIVE(1) TRACKEDMIN(0) EXCEPTIONMIN(10)'
REASON('less sensitive to avoid spurious trips')
DATE(20110728)
```

PFA APARs of interest

- OA31644 LOGREC ARRIVAL RATE SPURIOUS AIRH110E MESSAGE
- OA34655 PFA FRAMES AND SLOTS USAGE SPURIOUS. PREDICTIVE FAILURE ANALYSIS
- OA34586 HIGH CPU IN PFA AFTER PTF UA50081 FOR APAR OA29963 APPLIED
- OA35820 PREDICTIVE FAILURE ANALYSIS. Abend0C4 RC4 AIRHMEMA.x'FF' DECIMAL 255 PERSISTENT JOBS CAUSES A LOOP IN PFA THAT OVERLAYS STORAGE
- OA36462 ABEND0C4 IN AIRHMEXC
- OA36837 PREDICTIVE FAILURE ANALYSIS (PFA) TREATS TSO USERID AS PERSISTENT JOBS IN PFA_FRAMES_AND_SLOTS_USAGE CHECK

PFA APARs of interest

- **OA38175 PREDICTIVE FAILURE ANALYSIS ENHANCEMENT TO DETECTING EXCEPTIONS - An enhancement to the logic in detecting exceptions in z/OS 1.13 is being rolled back. This can help with some false positive exceptions. PTF UA63721 for 1.12**

- **OA38416 PREDICTIVE FAILURE ANALYSIS FALSE EXCEPTION JES_SPOOL_USAGE (JSU). (1.13 > only)**

- **OA38279 PREDICTIVE FAILURE ANALYSIS PFA_COMMON_STORAGE_USAGE CHECK FALSE EXCEPTION ESQA - PFA_COMMON_STORAGE_USAGE check can issue false exceptions for ESQA if there is no conversion of ECSA to ESQA storage and PFA detects that ESQA may exceed 100% This will trigger an exception to be issued. This is not valid because ECSA can still be used and the system is not in danger of being exhausted of storage.**

- **OA37801 PFA_SMF_ARRIVAL_RATE DEFAULT SETTINGS MAY CAUSE FALSE EXCEPTIONS PREDICTIVE FAILURE ANALYSIS**

AIRH109E A problem with common storage usage

-----Original Message-----

From: Enterprise Event Management

Sent: Friday, July 24, 2009 3:51 PM

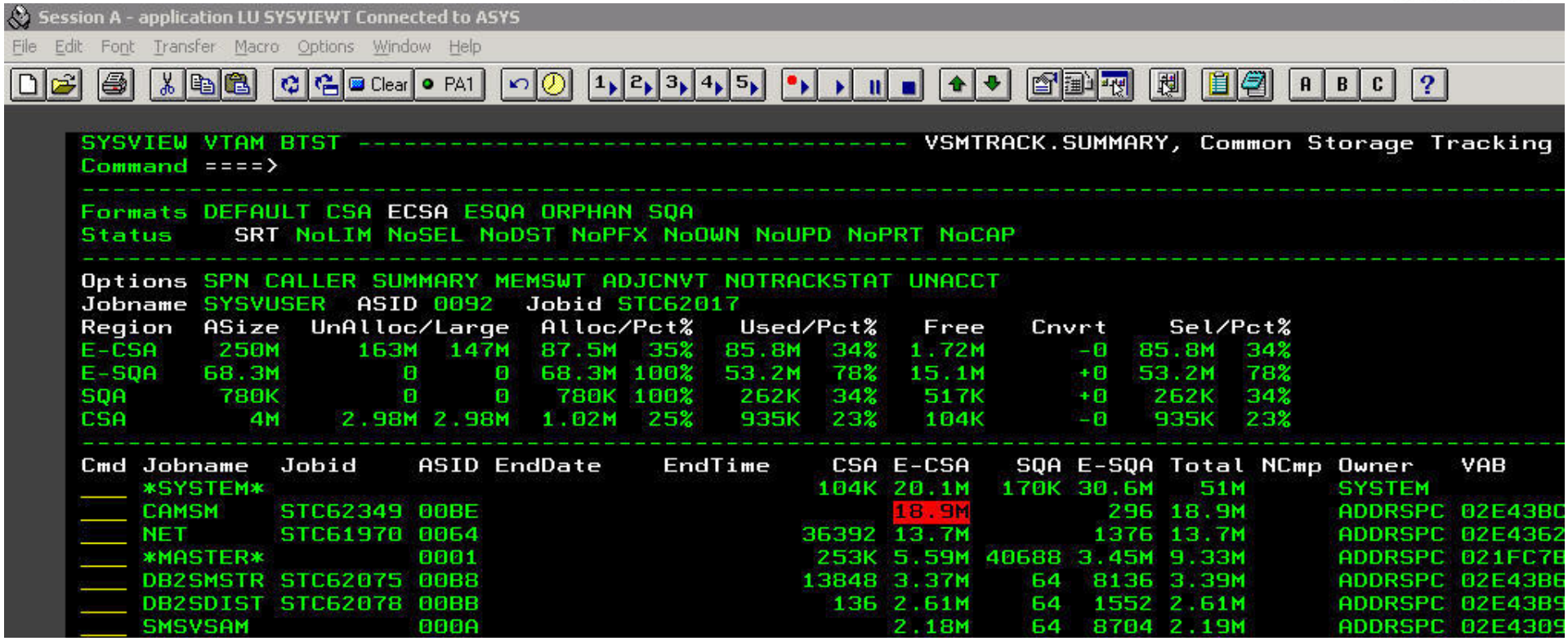
To: Knutson, Sam

Subject: z/OS Health Checker BTST 24 Jul 2009 HZS0004I CHECK|IBMPFA,PFA_COMMON_STO

z/OS Health Checker BTST 24 Jul 2009 HZS0004I CHECK|IBMPFA,PFA_COMMON_STORAGE_USAGE|: AIRH109E A problem with common storage usage |CSA and SQA| above the line was predicted to occur by 07/24/2009 21:51:17. The current usage is higher than expected based on an evaluation of the total capacity plus the threshold, the current usage, the current prediction, and the future prediction modeled at 07/24/2009 15:51:17.

```
----------------------------------------
* EMAIL NOTIFICATION SENT FROM OPS/MVS *
*                                      *
* SYSTEM :BTST                         *
*   RULE :MSGACTN.HZSINFO              *
*  GROUP :SKNUTSON                     *
*   DATE :24 Jul 2009  TIME: 15:51:17  *
----------------------------------------
```

AIRH109E A problem with common storage usage



- **Good exception system is still healthy only 34% used on ECSA but a task was out of profile. Testing on test Sysplex had found new bug.**
- **Recycle of started task temporarily resolved the problem and CA-CCI RO10195 and CA-DATACOM PTFs RO10065 were created**

PFA_LOGREC_ARRIVAL_RATE

CHECK(IBMPFA,PFA_LOGREC_ARRIVAL_RATE)
START TIME: 08/18/2009 12:03:47.499169
CHECK DATE: 20080330 CHECK SEVERITY: MEDIUM
CHECK PARM: DEBUG(0) STDDEV(2) COLLECTINT(60) MODELINT(360) COLLECTINACTIVE(1)

*** Medium Severity Exception ***

The LOGREC entry arrival rate is higher than expected and could cause a system problem by
08/18/2009 17:49:35.
When the LOGREC entry rate is higher than expected, it can indicate recurring failures on the system which can eventually lead to system hang. The prediction used in the comparison was modeled at 08/18/2009 11:49:35.

PFA_LOGREC_ARRIVAL_RATE

LOGREC Arrival Rate Prediction Report
Last successful model time : 08/18/2009 11:49:35
Next model time : 08/18/2009 17:49:35
Model interval : 360
Last successful collection time: 08/18/2009 11:52:37
Next collection time : 08/18/2009 12:52:37
Collection interval : 60

	Key 0	Key 1-7	Key 8-15
Arrivals in last collection interval:	77	364	8
Predicted rates based on...			
1 hour of data:	9	0	5

Jobs having LOGREC arrivals in last collection interval:

Job Name	ASID	Arrivals
DSMDOD	0205	438
NONE-FRR	00D0	1
U38T20	0344	1

PFA_LOGREC_ARRIVAL_RATE

- ABEND rate can vary widely and quickly
- Out of the box check tends to trip spuriously
- Remember this is saying “look at me” not I am broken
- Adjusted STDEV temporarily
- APAR OA31644 LOGREC ARRIVAL RATE SPURIOUS AIRH110E MESSAGE was the real resolution to spurious trips from this check and has been effective

PFA Customer Experience Summary

- I run Health Checker and PFA and you should too
- There are many opportunities for new and improved PFA checks. I would like to see PFA doing global monitoring of private virtual storage use for persistent address spaces
- Please work with IBM to tune your checks to work accurately and improve PFA for everyone
- z/OS 1.12 contained some useful enhancements to PFA which will not be rolled back to previous releases
- The goal of automatically configured proactive detection failures or looming problems is laudable
- This is a hard problem and I am not sure when PFA exceptions will be sufficiently accurate to treat them as operational alerts
- *HAL: I've just picked up a fault in the AE35 unit. It's going to go 100% failure in 72 hours. 2001: A Space Odyssey (1968)*

PFA : References

- Session 2866 IBM Experience Building Remote Checks for the IBM HEALTH CHECKER for z/OS presented at SHARE in Austin by James Caffrey
- Manual: z/OS Problem Management G325-2564-04 April 2009 not included in -02 on April 2009 z/OS V1R10 and Software Products DVD Collection need to obtain this from the web or a later DVD
- WSC Flash WP101454 by Riaz Ahmad can be found on the web, www.ibm.com/support/techdocs under the category of "White Papers."
- Session 2858 Health Checker: User Experience Beyond Installation presented at SHARE in Tampa by Sam Knutson and Dave Danner
- Session 2208 Bit Bucket x'26' presented at SHARE in Denver

Acknowledgements (Knowing or unKnowing)

- James Caffrey (IBM)
- Vincent Bonanno (IBM)
- Bob Abrams (IBM)
- Karla Arndt (IBM)
- Ashvin Desai (IBM)
- Keith Martens (GEICO)
- John Wallingford (GEICO)
- Mary Anne Matyaz (CBP)
- Bob Rogers (IBM)
- Geoff Miller (IBM)

Bob's Agenda:
Detecting and Diagnosing Problems when z/OS "Thinks" it is Running Okay

Soft Failures: Detection, Prevention, Diagnosis

- 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
  - **Business Application view:** Systems Management products

- Lessons learned on reducing impact of soft failures

All elements work together for an integrated IBM solution approach

What is a soft failure?

"Your systems don't break. They just stop working and we don't know why."

"Sick, but not dead" or Soft failures

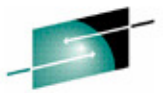
Soft Failures

- Exhaustion of shared resources
- Recurring or recursive failures
- Serialization problems (deadlocks, priority inversions)
- Unexpected state transition

Manifested as

- Stalled / hung processes
  - Single system, sysplex members
  - Sympathy Sickness
- Resource Contention
- Storage growth
- CF, CDS growth
- I/O issues (channel paths, response time)
- Repetitive errors
- Queue growth
- Configuration
  - SPOF, thresholds, cache structure size, not enabling new features

Integrated z/OS Soft Failure Detection & Prevention



Multi-domain consolidation
Networks
Business applications
Security

Performance analysis
Resource management

Automation

Systems Management Products

Soft Failure Detection
- **Performance**
- **Events**
- **Take corrective actions**

Operating System

Console / Operlog

Runtime Diagnostics

z/OS 1.13

Soft Failure Problem Determination

Soft Failure Avoidance

Health Checker

Predictive Failure Analysis (PFA)

Soft Failure Detection

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

Some general considerations …

- The key to reducing the impact of soft failures is
  - Avoid them using z/OS Health Checker
  - Enable system checking where possible
  - Automate alerts
    - Display, take action

- z/OS can survive / recover from most soft failures
  - Take advantage of what the base operating system has to offer
    - Soft failure detection across many z/OS components
    - Start Health Checker, PFA, RTD (R13) at IPL (e.g., COMMNDxx)
  - Predictive trend analysis is not intended to find immediate problems that will bring down a system
    - PFA Sampling minimum is 1 minute … 15 minutes for some checks

Detection of Soft Failures by z/OS Components

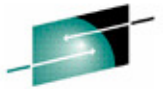
- z/OS attempts to detect soft failures as close to the source as possible
  - Uses the least amount of resources
  - Requires the smallest amount of the stack to do detection
- Detection of a soft failure requires ability to identify when something is wrong
  - Thresholds set by the installation
- *Whenever possible, components try to avoid soft failures*
- Examples …

Component Examples: Detection, Identification of soft failures … Single system

Component	Features
GRS	Enhanced contention analysis for ENQ, Latch GRS Latch Identify string WLM management of blocking units
UNIX System Services	Latch identity exploitation XCF communication improvements (R13) System limits D OMVS,WAITERS to diagnose file system latch contention
JES2	JES2 Monitor
IOS	Missing Interrupt Handler Identify systems sharing a reserve Captured UCB protection I/O timing facility Detect & remove “flapping links” Dynamic Channel Path Management
DFSMS	CAS contention detection VSAM RLS index traps Media Manager recovery from I/O errors

Details in backup section

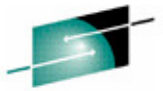
Component Examples: Detection of soft failures … Sysplex

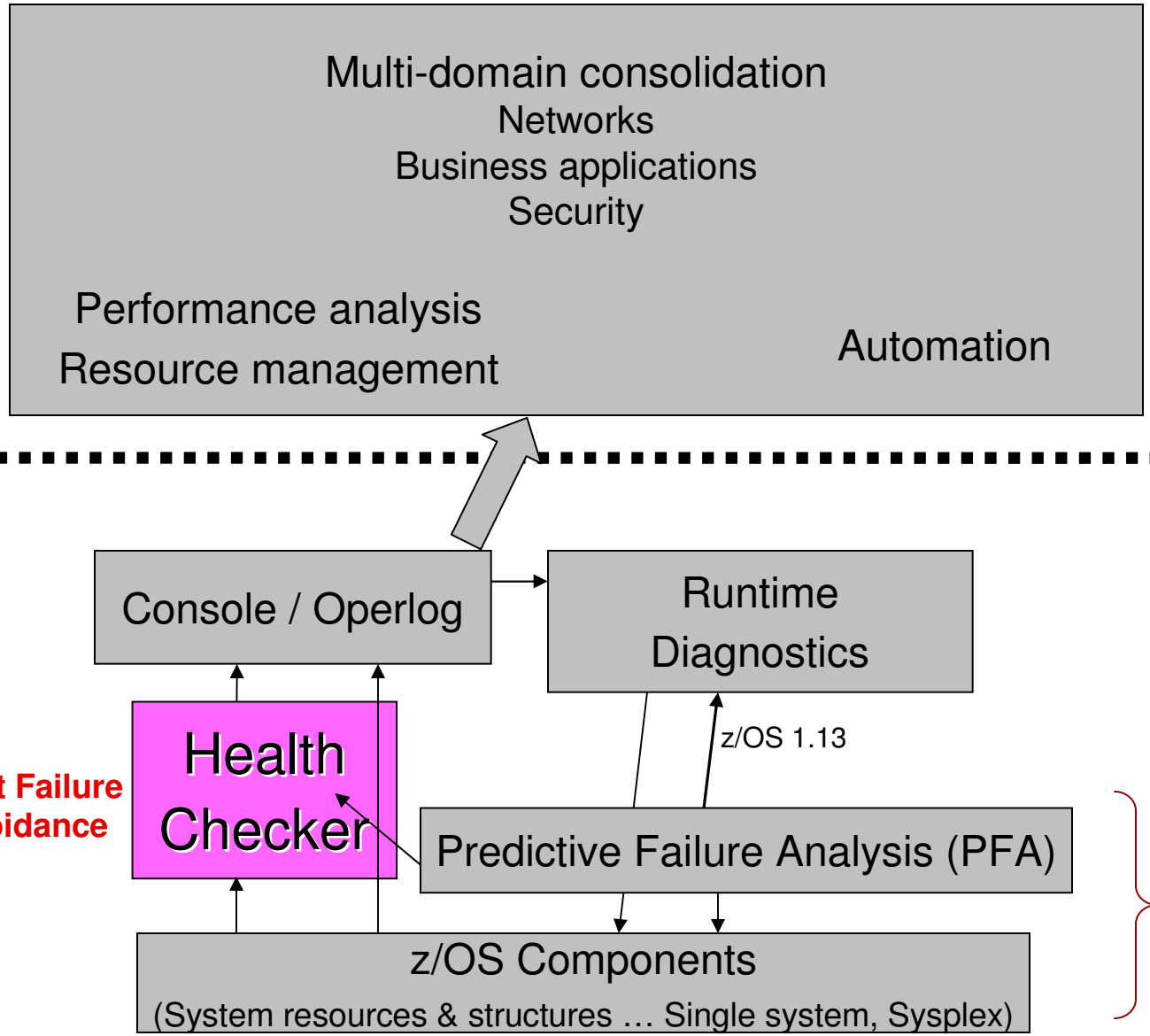


Component	Features	Functions
XCF / XES	Stalled member support	Identify unresponsive system, restore to normal operation OR remove it to avoid sympathy sickness
	Exploitation of BCPii to determine dead system more quickly	Avoid waiting the Failure Detection Interval (FDI) if the system is truly dead …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

Details in backup section

Integrated z/OS Soft Failure Detection & Prevention





Systems Management Products

Soft Failure Detection
- **Performance**
- **Events**
- **Take corrective actions**

Operating System

Soft Failure Problem Determination

Soft Failure Avoidance

Soft Failure Detection

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)

Health Checker: Soft Failure avoidance Important examples

Component	Health Check	Functions
XCF	XCF_CDS_SPOF	Evaluate primary & secondary CDS configuration to determine if Sysprog inadvertently created a **single point of failure**
	XCF_SFM_SUM_ACTION	Check ISOLATETIME value, to allow SFM to fence and partition a system without operator intervention and without undue delay.
	XCF_SFM_SUMLIMIT	Checks status update missing (SUMLIMIT) value
	XCF_SFM_ACTIVE	Verifies SFM active, policy values
	XCF_SFM_CFSTRHANG TIME	Verifies CFSTRUCTURE hang time
	XCF_SFM_CONNFAIL	Threshold for loss of connectivity
RACF	RACF_GRS_RNL	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)

Details in backup section

Health Checker: Soft Failure avoidance examples

Component	Health Check	Functions
Serviceability	DAE_SUPPRESSING	DAE suppressing duplicate SVC dumps, saving system resources for unnecessary dumps
	SVA_AUTOIPL_DEFINED	Check whether Program-Directed IPL and not GDPS, and whether AUTOIPL policy is active
	SVA_AUTOIPL_DEV_VALIDATION	Validates SADMP, MVS IPL devices
UNIX System Services	USS_PARMLIB	Validate current system against parmlib IPL'd with Remind you to update parmlib (due to dynamic changes)
	USS_CLIENT_MOUNTS	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*
	USS_FILESYS_CONFIG	Checks if mount attribute access is read only; whether HFS's in Sysplex root

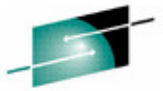
Details in backup section

Important considerations when enabling z/OS Health Checks

1. Don't just change the configuration … investigate the exception and then take appropriate action
2. There are 187 Health Checks in z/OS R13
   a. Start Health Checker, activating all checks and try to resolve exceptions
   b. *Don't* think that you must activate all health checks at once to get benefit
   c. Goal should be to remove all exceptions
      - by fixing the condition
      - by tuning the check so that it looks for what you need it to look for
      - *(as a last resort)* by deactivating the check
   d. Consider categorizing health checks by
      1) Checks I expect no exceptions from
      2) Checks not turned on because exceptions not cleaned up yet
      3) Plan to move checks to group 1 as you clean up exceptions
   e. Once you can run cleanly, you will be in the ideal position to know that an exception indicates something has changed

3. GDPS recommendations for changing z/OS checks trump z/OS in a GDPS environment

Integrated z/OS Soft Failure Detection & Prevention



Multi-domain consolidation
Networks
Business applications
Security

Performance analysis
Resource management

Automation

Systems Management Products

Soft Failure Detection
- **Performance**
- **Events**
- **Take corrective actions**

Operating System

Console / Operlog

Runtime Diagnostics

z/OS 1.13

Soft Failure Problem Determination

Soft Failure Avoidance

Health Checker

Predictive Failure Analysis (PFA)

Soft Failure Detection

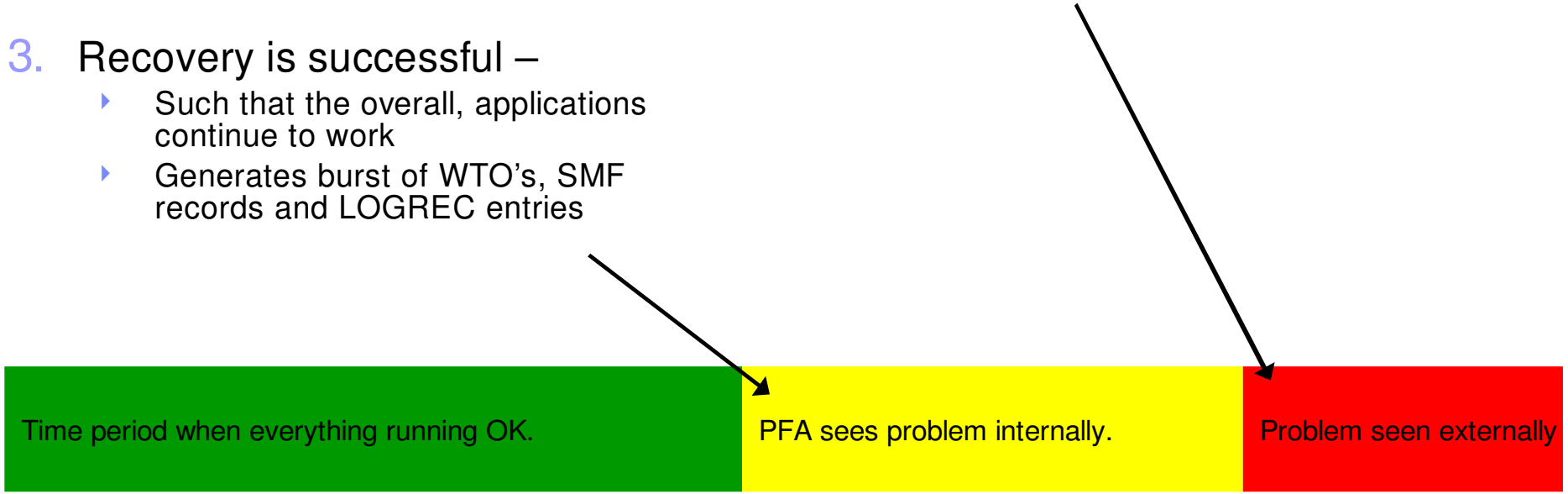
z/OS Components
(System resources & structures … Single system, Sysplex)

Problem Determination in a complex environment

Installation Pain Points	Requirement Areas
Risk to the business • The impact of the symptoms • Risk of recurrence • Impact in getting system stabilized • Mean time to recovery too long	**Requirement: Detect** "sick, but not dead" event **BEFORE** it causes problems; turn it into a correctable incident **Solution:** Predictive Failure Analysis
Complexity of performing the task	**Requirement: Diagnose** the cause in **real time** to allow operations to mitigate event inquiries **Solution:** Runtime Diagnostics
Troubleshooting a live system and recovering from an apparent failure	
Data collection very time-consuming	**Requirement: Manage / capture** data to determine cause of problem **Solution:** z/OSMF Incident Log
Significant skill level needed to analyze problems, interact with IBM and ISVs to obtain additional diagnostic info	

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"



This is a hypothetical problem which is a combination of multiple actual problems

Soft Failure Detection: Predictive Failure Analysis

- Models trends in certain z/OS system resources & metrics
- Predicts expected, normal behavior as well as future behavior; identifies exceptions as Soft Failures
  - ... when the resources will run out, or when metrics become abnormal when compared to expected prediction
  - Machine-learning technology used to determine what's normal
  - Statistical analysis used to identify exceptions … focusing on metrics affecting different layers of the software stack
    - Exceptions alerted and reports written using Health Checker for z/OS
    - Tune comparison algorithms using configurable parameters such as STDDEV; defaults selected based on IBM test systems and customer data
    - Tunable sensitivity and configuration parameters per check
  - Identifies areas related to
    - resource exhaustion
    - damaged address spaces and damaged systems
  - Invokes Runtime Diagnostics to check for hung address spaces (R13); RTD validates and suggests next steps

How PFA Chooses Address Spaces to Track

- **Some metrics require data for the entire system to be tracked**
  - Exhaustion of common storage for entire system
  - LOGREC arrivals for entire system grouped by key
- **Some metrics call for tracking only persistent address spaces**
  - Those that start within the first hour after IPL.
  - For example, tracks frames and slots usage to detect potential virtual storage leaks in persistent address spaces.
- **Some metrics are most accurate when using several categories**
  - *"Chatty" persistent* address spaces tracked individually
    - Start within the first hour after IPL and have the highest rates after a warm-up period
    - Data from first hour after IPL is ignored.
    - After an IPL or PFA restart, if all are running, same address spaces are tracked.
    - Duplicates with the same name are not tracked
    - Restarted address spaces that are tracked are still tracked after restart.
  - *Other persistent* address spaces as a group
  - *Non-persistent* address spaces as a group
  - *Total system rate* ("chatty" + other persistent + non-persistent)

PFA Functions

- **PFA address space …**
  - *Collects data* from the system
  - *Models the data* from the system to create predictions (complex algorithms)
  - *Performs comparisons* on current vs. predictions
  - *Issues exceptions or "OK" messages and reports* via IBM Health Checker for z/OS

- **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

Example Report: Logrec Arrival Rate Prediction Report

- Available in SDSF (s.ck)
- Heading information
  - Configuration and status
  - Current and predicted information for metric
- Top predicted users
  - Tries to pinpoint potential villains
- IBM Health Checker for z/OS message in its entirety

```
LOGREC Arrival Rate Prediction Report
(heading information intentionally omitted)
                                 Key 0      Key 1-7     Key 8-15
                               _________   _________   _________
Arrivals in last
   collection interval:              1           0           2
Predicted rates based on...
     1 hour of data:                 1           0           1
   24 hours of data:                 0           0           1
     7 days of data:                 0           0           1
    30 days of data:                 0           0           1
Jobs having LOGREC arrivals in last collection interval:
   Job Name        ASID        Arrivals
   ________        ____        ________
   LOGREC08        0029               2
   LOGREC00        0027               1
```

The PFA Health Checks

- 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
  - Frames and slots usage check
    - Tracks all address spaces that start within an hour after IPL (persistent)
  - Message arrival rate (WTO/WTOR) check
    - Chatty, persistent address spaces
    - Non-chatty, persistent address spaces
    - Non-persistent address spaces
    - Total system
- z/OS 1.12
  - SMF arrival rate check
    - Same categories as message arrival rate check
  - Modeling improvements
    - More granular common storage check
    - Supervised learning (exclude jobs)
    - Dynamic modeling
  - Performance and serviceability
- z/OS 1.13
  - JES spool usage check
    - Tracks all persistent address spaces
    - JES2 only
  - Enqueue request rate check
    - Chatty, persistent address spaces
    - Total system
  - Integration with Runtime Diagnostics to detect rates that are “too low”

How often do you see soft failures on your systems?

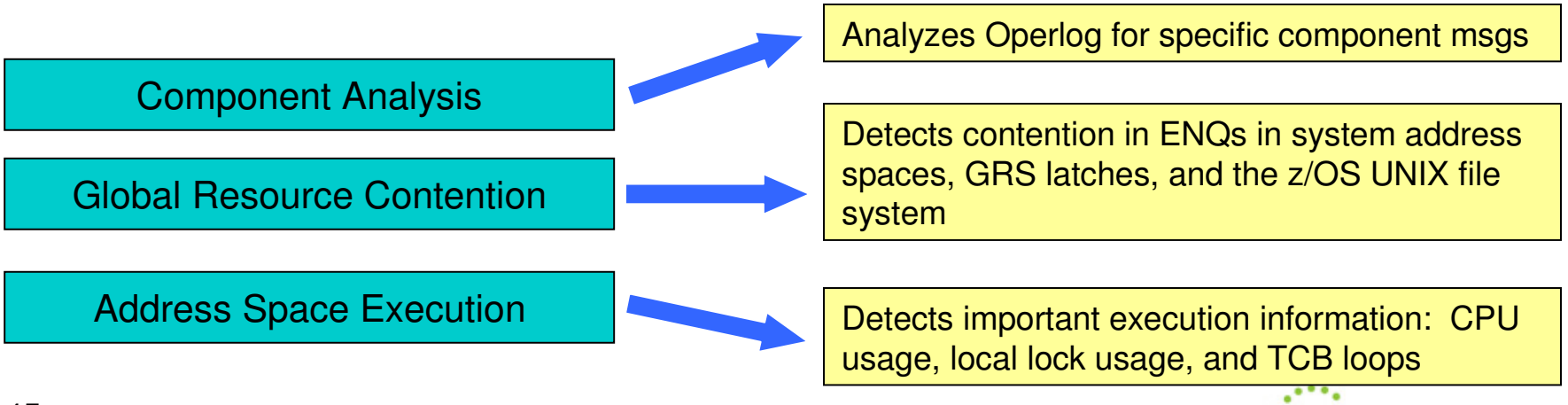
PFA updates

- Eliminate setup issues
  - Use zFS size recommendations in Doc APAR (OA33776)
    - Problem where PFA not closing files (++OA38786 – HIPER)
    - Use supported Java version and configure location
    - Java 5.0 and up (31-bit only)
    - Use one PFA ini file (/etc/PFA/ini) (R12)
  - Follow instructions in *z/OS Problem Management*
- Tune comparison algorithms
  - Adjust sensitivity -- STDDEV, EXCEPTIONMIN, etc.
  - EXCLUDED_JOBS (exclude jobs with erratic behavior)
- Use a zAAP and start PFA at IPL
- *Get the latest PTFs!!*
  - Some algorithms tuned -- FSU, MAR, CSA, JSU checks
  - Changed EXCEPTIONMIN & STDDEV defaults
  - Design changes
    - Exclude interactive users from being persistent jobs
    - Skip comparisons for ESQA

Details in backup

Runtime Diagnostics

- Analyzes a “sick, but not dead” system in a timely manner
- Performs analysis similar to a very experienced system programmer
  - But faster – goal of 60 seconds or less
  - More comprehensive
  - Looks for specific evidence of “soft failures”
  - Provides suggested next steps
- Runtime Diagnostics
  - Is not automation or a monitor
  - Takes no corrective action, but recommends next steps
  - Has no background processing and minimal dependencies on system services



Runtime Diagnostics Invocation and Output

- z/OS 1.12 → Started task -- “Run” the analysis via a START command
  - START HZR,SUB=MSTR
- z/OS 1.13 → Address space – Start with command above and Run with modify command
  - f hzr,analyze

```
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.
```

Example output

- Left → HIGHCPU and LOOP
- Below → GRS Latch Contention

```
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 MINTES
 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.
```

Runtime Diagnostics Symptoms Detected

- 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 system 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 TCP 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

z/OS 1.13 PFA Integration with Runtime Diagnostics

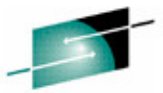
- **Detects damaged or hung system or address space based on rates being "too low"**
  - When PFA detects too low, Runtime Diagnostics is executed
- **Output**
  - **"Too low" exception** message sent as WTO by default
  - **Runtime Diagnostics output** included in PFA report
  - Prediction report and result message **available in SDSF** (sdsf.ck)
  - **PFA current rates and predictions** relevant to category causing exception
- **Supported for** Message Arrival Rate, SMF Arrival Rate, Enqueue Request Rate

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

Persistent address spaces with low rates:
                                                 Predicted Message
                                                   Arrival Rate
   Job                  Message
   Name      ASID       Arrival Rate      1 Hour       24 Hour          7 Day
   ________  ____  _____________ _____________ _____________ _____________
   JOBS4     001F           1.17         23.88         22.82         15.82
   JOBS5     002D           2.01          8.34         11.11         12.11

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:
   ++++++++
     JOBSTART: 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: 004E6850
     STEPNAME: PFATEST PROCSTEP: PFATEST JOBID: STC00042 USERID:
   ++++++++
     JOBSTART: 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.)
```

Extending to Systems Management Products

- Many (ISV) Systems Management products support
  - Actions based on WTO message events
  - Automation of Health Check events
    - PFA Health Check events = soft failures
  - Performance analysis
  - Policy to control corrective actions
  - Integration of Alert displays, performance exceptions, event based actions

Integrated z/OS Soft Failure Detection & Prevention



Multi-domain consolidation
Networks
Business applications
Security

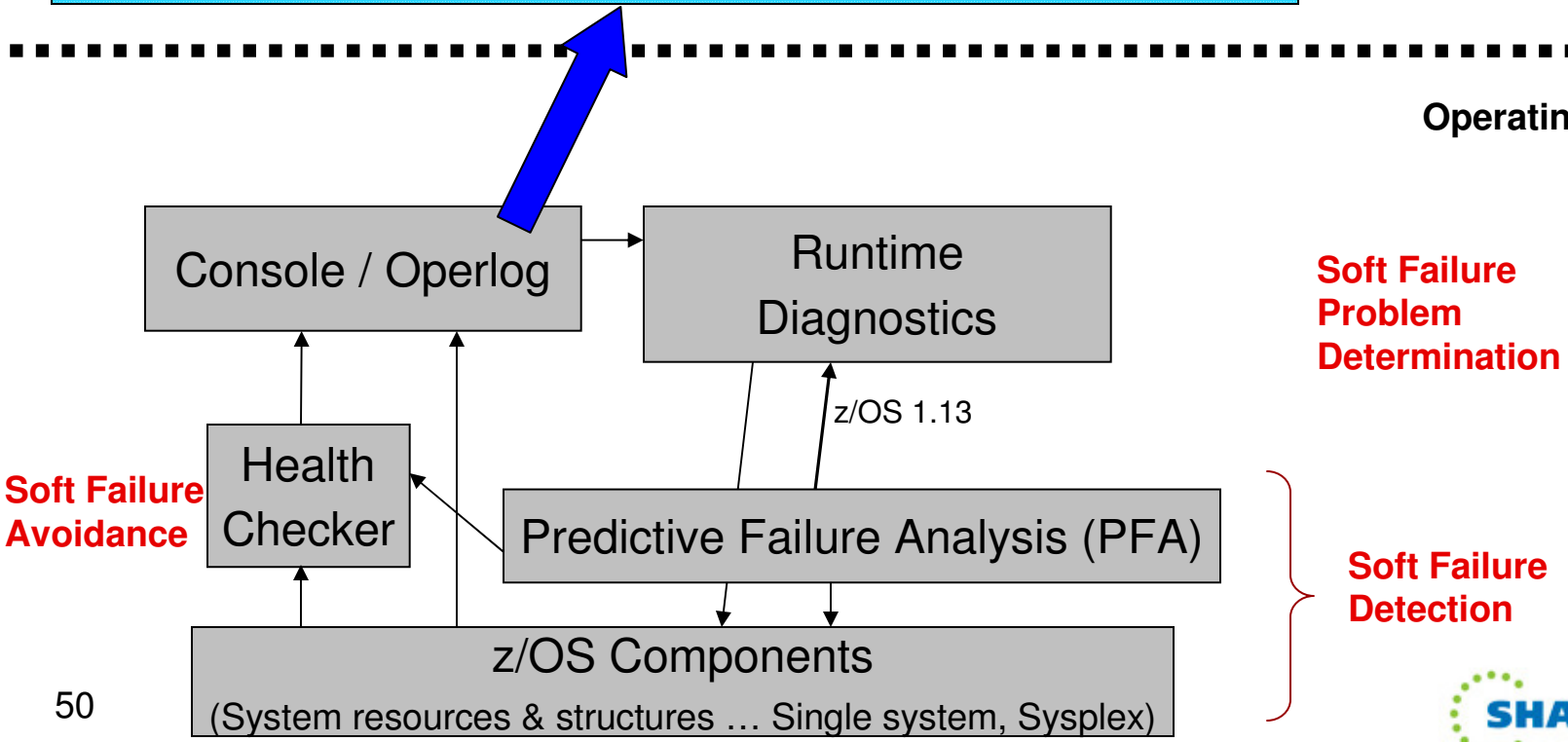
Performance analysis
Resource management

Automation

Systems Management Products

Soft Failure Detection
- **Performance**
- **Events**
- **Take corrective actions**

Operating System

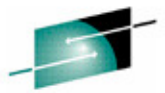


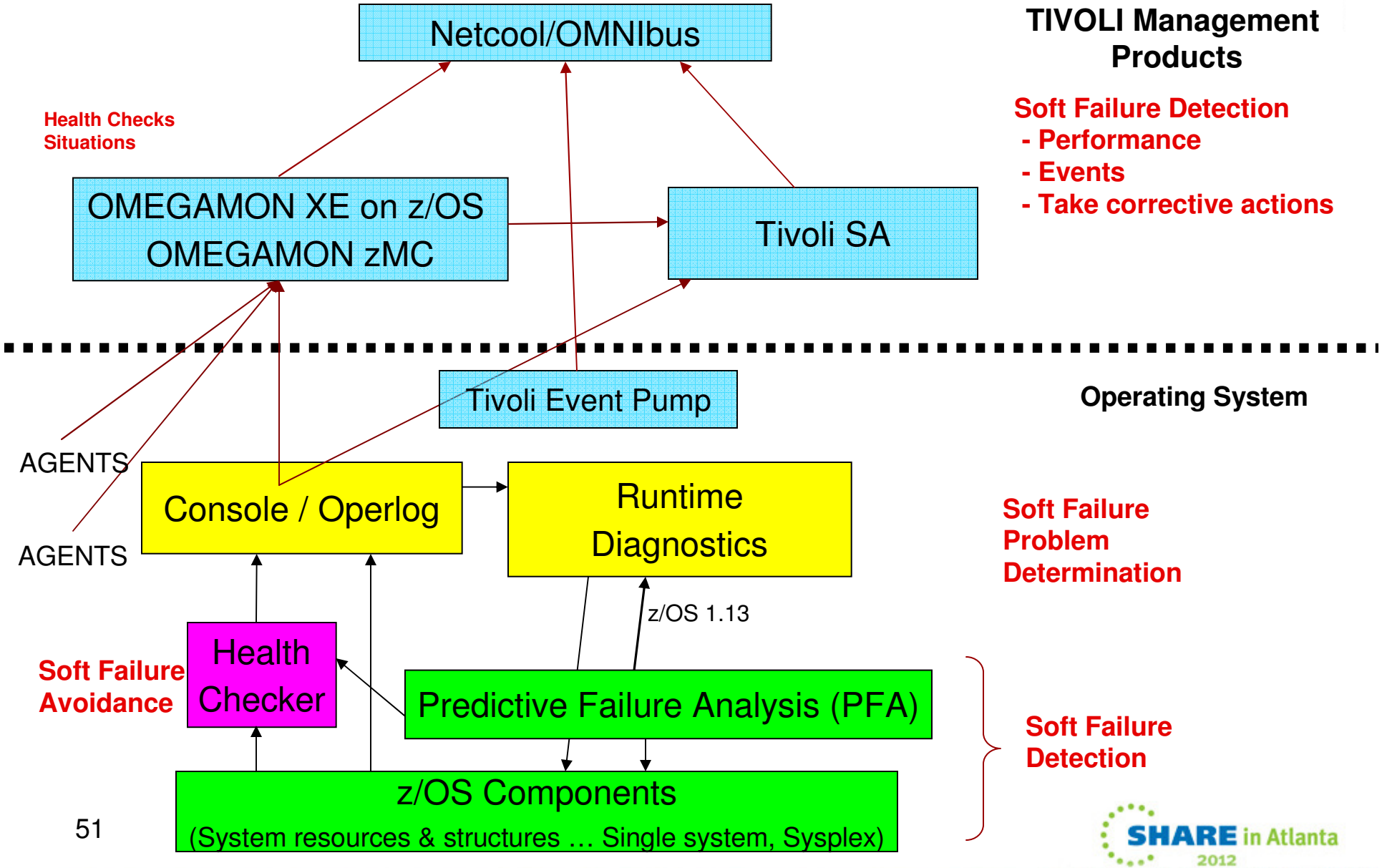
Soft Failure Problem Determination

Soft Failure Avoidance

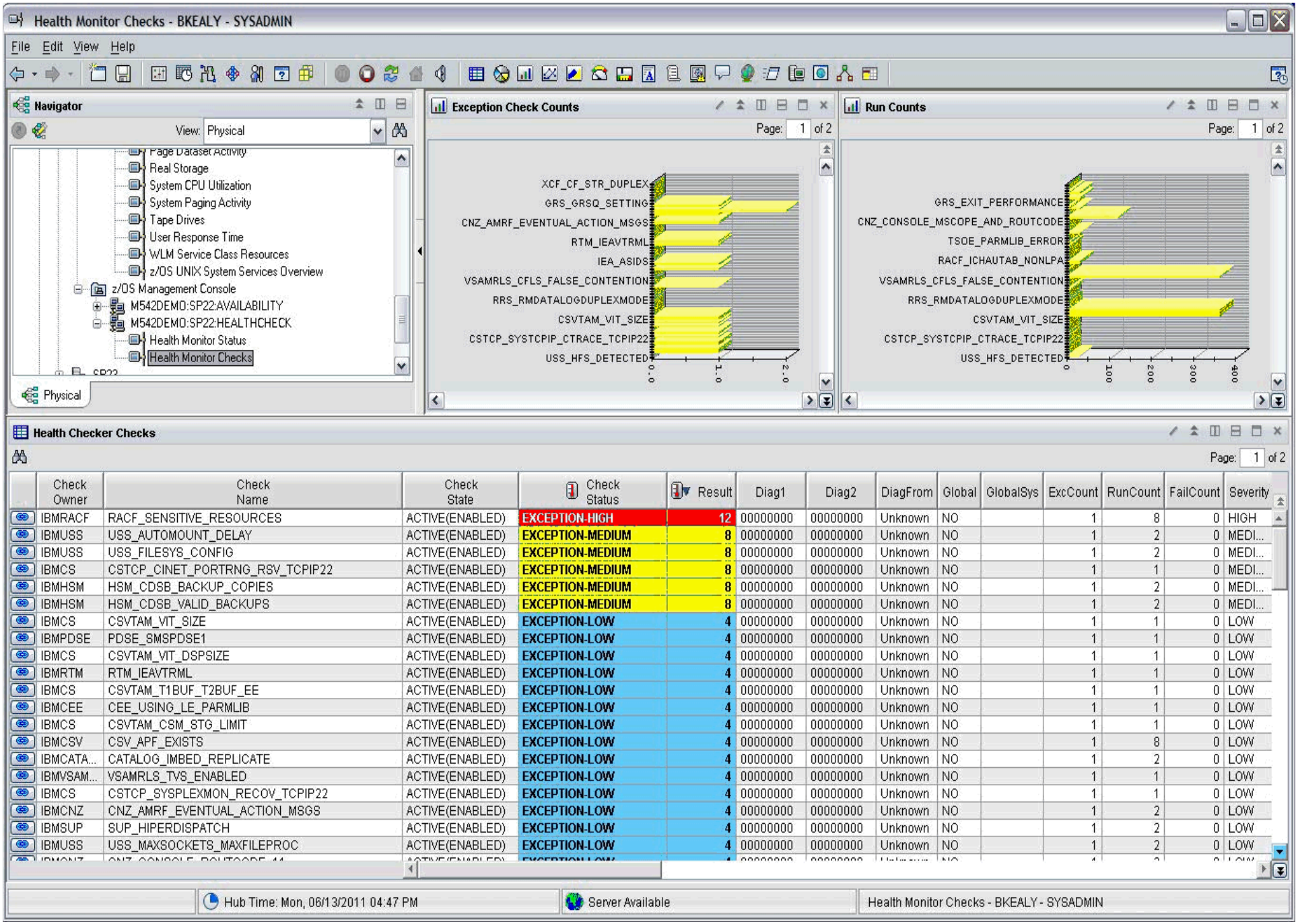
Soft Failure Detection

Integrated z/OS Soft Failure Detection & Prevention

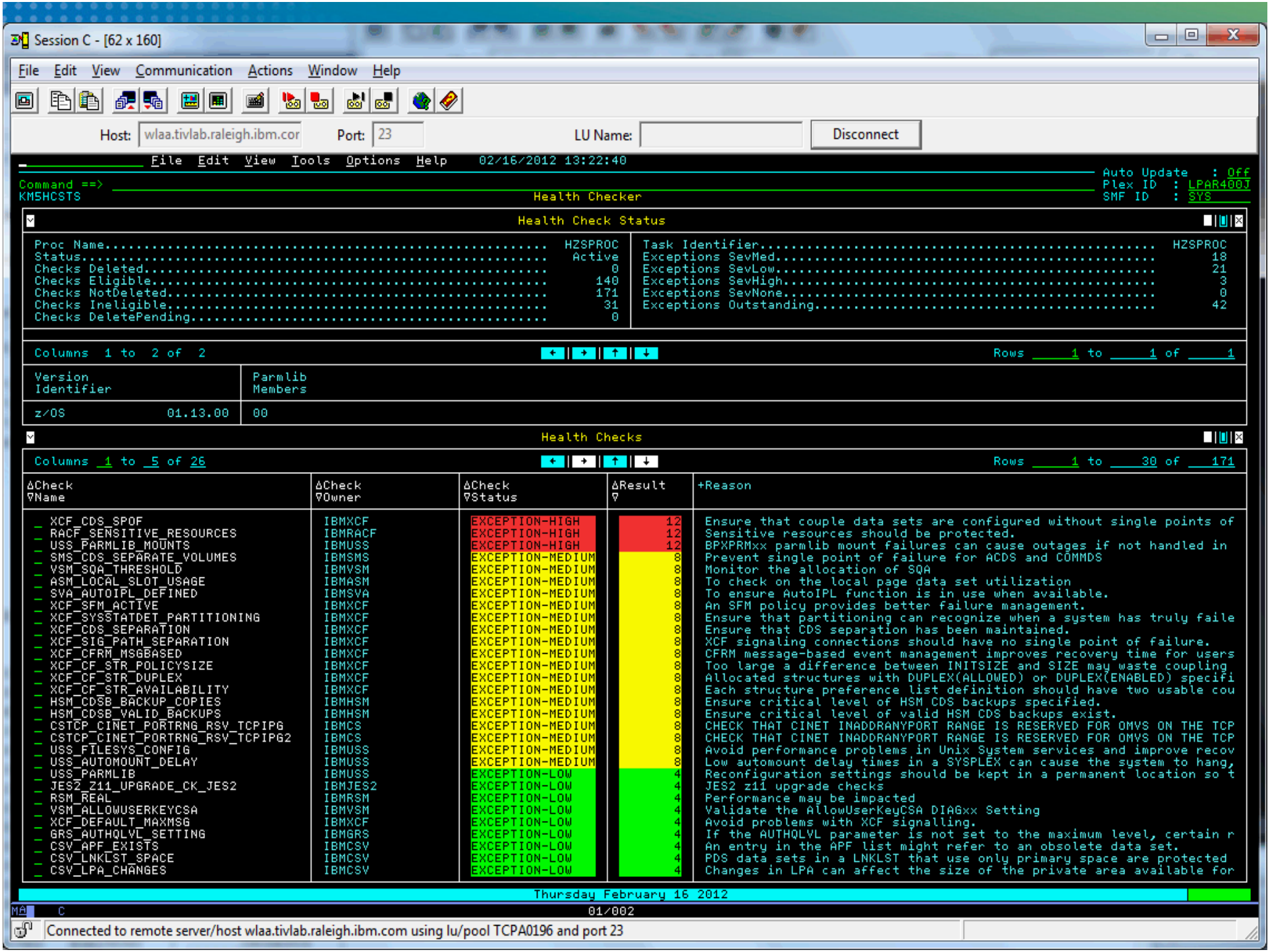




zMC Health Check display



Check Owner	Check Name	Check State	Check Status	Result	Diag1	Diag2	DiagFrom	Global	GlobalSys	ExcCount	RunCount	FailCount	Severity
IBMRACF	RACF_SENSITIVE_RESOURCES	ACTIVE(ENABLED)	EXCEPTION-HIGH	12	00000000	00000000	Unknown	NO		1	8	0	HIGH
IBMUSS	USS_AUTOMOUNT_DELAY	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000	00000000	Unknown	NO		1	2	0	MEDI...
IBMUSS	USS_FILESYS_CONFIG	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000	00000000	Unknown	NO		1	2	0	MEDI...
IBMCS	CSTCP_CINET_PORTRNG_RSV_TCPIP22	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000	00000000	Unknown	NO		1	1	0	MEDI...
IBMHSM	HSM_CDSB_BACKUP_COPIES	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000	00000000	Unknown	NO		1	2	0	MEDI...
IBMHSM	HSM_CDSB_VALID_BACKUPS	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000	00000000	Unknown	NO		1	2	0	MEDI...
IBMCS	CSVTAM_VIT_SIZE	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMPDSE	PDSE_SMSPDSE1	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCS	CSVTAM_VIT_DSPSIZE	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMRTM	RTM_IEAVTRML	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCS	CSVTAM_T1BUF_T2BUF_EE	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCEE	CEE_USING_LE_PARMLIB	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCS	CSVTAM_CSM_STG_LIMIT	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCSV	CSV_APF_EXISTS	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	8	0	LOW
IBMCATA...	CATALOG_IMBED_REPLICATE	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	2	0	LOW
IBMVSAM...	VSAMRLS_TVS_ENABLED	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCS	CSTCP_SYSPLEXMON_RECOV_TCPIP22	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	1	0	LOW
IBMCNZ	CNZ_AMRF_EVENTUAL_ACTION_MSGS	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	2	0	LOW
IBMSUP	SUP_HIPERDISPATCH	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	2	0	LOW
IBMUSS	USS_MAXSOCKETS_MAXFILEPROC	ACTIVE(ENABLED)	EXCEPTION-LOW	4	00000000	00000000	Unknown	NO		1	2	0	LOW

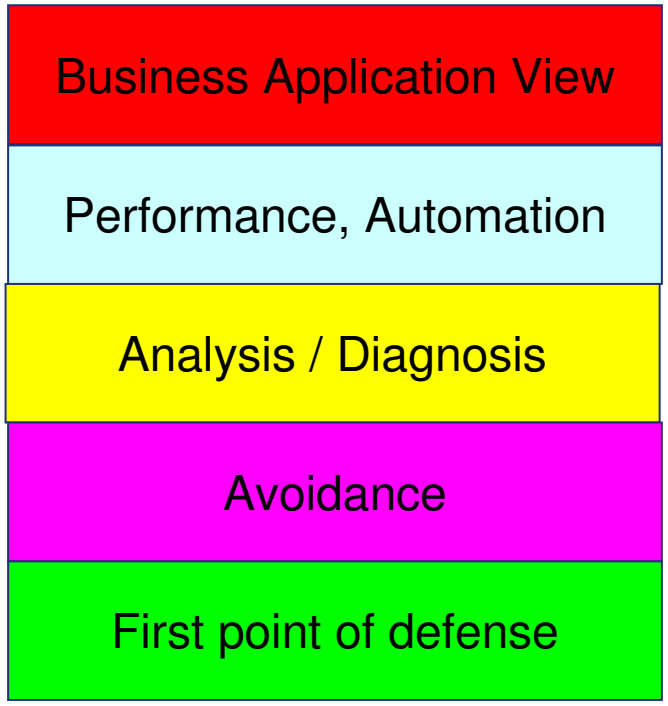

Session C - [62 x 160]
File Edit View Communication Actions Window Help
Host: wlaa.tivlab.raleigh.ibm.cor
Port: 23
LU Name:
Disconnect
File Edit View Tools Options Help 02/16/2012 13:22:40
Auto Update : Off
Command ==>
Plex ID : LPAR400J
KM5HCSTS
Health Checker
SMF ID : SYS
Health Check Status
Proc Name..... HZSPROC
Status..... Active
Checks Deleted..... 0
Checks Eligible..... 140
Checks NotDeleted..... 171
Checks Ineligible..... 31
Checks DeletePending..... 0
Task Identifier..... HZSPROC
Exceptions SevMed..... 18
Exceptions SevLow..... 21
Exceptions SevHigh..... 3
Exceptions SevNone..... 0
Exceptions Outstanding..... 42
Columns 1 to 2 of 2
Rows 1 to 1 of 1
Version Identifier
Parmlib Members
z/OS 01.13.00
00
Health Checks
Columns 1 to 5 of 26
Rows 1 to 30 of 171
ΔCheck ∇Name
ΔCheck ∇Owner
ΔCheck ∇Status
ΔResult ∇
+Reason
XCF_CDS_SPOF IBMXCF EXCEPTION-HIGH 12 Ensure that couple data sets are configured without single points of
RACF_SENSITIVE_RESOURCES IBMRACF EXCEPTION-HIGH 12 Sensitive resources should be protected.
USS_PARMLIB_MOUNTS IBMUSS EXCEPTION-HIGH 12 BPXPRMxx parmlib mount failures can cause outages if not handled in
SMS_CDS_SEPARATE_VOLUMES IBMSMS EXCEPTION-MEDIUM 8 Prevent single point of failure for ACDS and COMMDS
VSM_SQA_THRESHOLD IBMVSM EXCEPTION-MEDIUM 8 Monitor the allocation of SQA
ASM_LOCAL_SLOT_USAGE IBMASM EXCEPTION-MEDIUM 8 To check on the local page data set utilization
SVA_AUTOIPL_DEFINED IBMSVA EXCEPTION-MEDIUM 8 To ensure AutoIPL function is in use when available.
XCF_SFM_ACTIVE IBMXCF EXCEPTION-MEDIUM 8 An SFM policy provides better failure management.
XCF_SYSSTATDET_PARTITIONING IBMXCF EXCEPTION-MEDIUM 8 Ensure that partitioning can recognize when a system has truly faile
XCF_CDS_SEPARATION IBMXCF EXCEPTION-MEDIUM 8 Ensure that CDS separation has been maintained.
XCF_SIG_PATH_SEPARATION IBMXCF EXCEPTION-MEDIUM 8 XCF signaling connections should have no single point of failure.
XCF_CFRM_MSGBASED IBMXCF EXCEPTION-MEDIUM 8 CFRM message-based event management improves recovery time for users
XCF_CF_STR_POLICYSIZE IBMXCF EXCEPTION-MEDIUM 8 Too large a difference between INITSIZE and SIZE may waste coupling
XCF_CF_STR_DUPLEX IBMXCF EXCEPTION-MEDIUM 8 Allocated structures with DUPLEX(ALLOWED) or DUPLEX(ENABLED) specifi
XCF_CF_STR_AVAILABILITY IBMXCF EXCEPTION-MEDIUM 8 Each structure preference list definition should have two usable cou
HSM_CDSB_BACKUP_COPIES IBMHSM EXCEPTION-MEDIUM 8 Ensure critical level of HSM CDS backups specified.
HSM_CDSB_VALID_BACKUPS IBMHSM EXCEPTION-MEDIUM 8 Ensure critical level of valid HSM CDS backups exist.
CSTCP_CINET_PORTRNG_RSV_TCPIPG IBMCS EXCEPTION-MEDIUM 8 CHECK THAT CINET INADDRANYPORT RANGE IS RESERVED FOR OMVS ON THE TCP
CSTCP_CINET_PORTRNG_RSV_TCPIPG2 IBMCS EXCEPTION-MEDIUM 8 CHECK THAT CINET INADDRANYPORT RANGE IS RESERVED FOR OMVS ON THE TCP
USS_FILESYS_CONFIG IBMUSS EXCEPTION-MEDIUM 8 Avoid performance problems in Unix System services and improve recov
USS_AUTOMOUNT_DELAY IBMUSS EXCEPTION-MEDIUM 8 Low automount delay times in a SYSPLEX can cause the system to hang,
USS_PARMLIB IBMUSS EXCEPTION-LOW 4 Reconfiguration settings should be kept in a permanent location so t
JES2_Z11_UPGRADE_CK_JES2 IBMJES2 EXCEPTION-LOW 4 JES2 z11 upgrade checks
RSM_REAL IBMRSM EXCEPTION-LOW 4 Performance may be impacted
VSM_ALLOWUSERKEYCSA IBMVSM EXCEPTION-LOW 4 Validate the AllowUserKeyCSA DIAGxx Setting
XCF_DEFAULT_MAXMSG IBMXCF EXCEPTION-LOW 4 Avoid problems with XCF signalling.
GRS_AUTHQLVL_SETTING IBMGRS EXCEPTION-LOW 4 If the AUTHQLVL parameter is not set to the maximum level, certain r
CSV_APF_EXISTS IBMCSV EXCEPTION-LOW 4 An entry in the APF list might refer to an obsolete data set.
CSV_LNKLST_SPACE IBMCSV EXCEPTION-LOW 4 PDS data sets in a LNKLST that use only primary space are protected
CSV_LPA_CHANGES IBMCSV EXCEPTION-LOW 4 Changes in LPA can affect the size of the private area available for
Thursday February 16 2012
MA C 01/002
Connected to remote server/host wlaa.tivlab.raleigh.ibm.com using lu/pool TCPA0196 and port 23

Overall: Reducing Impact of Soft Failures

- Automation of alerts is key
  - Display, take action
- z/OS can survive / recover from most soft failures
  - But, take advantage of what the base operating system has to offer
    - Soft failure detection across many z/OS components
    - Start Health Checker, PFA, RTD (R13) at IPL (e.g., COMMNDxx)
- Most metrics are very time sensitive
  - Defaults selected based on z/OS test environments; should be good for most
- Predictive trend analysis typically not done on a Machine-time scale
  - PFA not designed to detect anomalies that could terminate a system on machine-time scale
  - Shortest data comparison is once a minute; identification of a program consuming CSA make take a couple minutes
  - PFA has tuned comparison algorithms using what is learned from your system
    - Configuration parameters are tunable to make the algorithms more accurate for your workloads
    - All checks have configurable parameters, e.g. STDDEV (Lower → more sensitive)

Summary

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



Systems Management Products

Predictive Failure Analysis
Runtime Diagnostics

Health Checker

z/OS Components

All elements work together for an integrated IBM soft failure solution … *Set Them Up!*

Acknowledgements

- Thank you to all who contributed information for this presentation

Jim Caffrey	z/OS Predictive Technologies
Karla Arndt	PFA / RTD
Scott Bender	USS Kernel
Ron Bretschneider	DFSMS - Media Manager
Mark Brooks	XCF, XES, CF
John Case	USS File System
Brian Kealy	Omegamon
Nick Matsakis	GRS, Availability
Terri Menendez	DFSMS - RLS
Peter Relson	Health Checker
Dale Riedy	IOS
Wayne Rhoten	DFSMS
Dave Surman	z/OS Architect
Tom Wasik	JES2
Doug Zobre	System Logger

Backup Reference – Recent PFA Enhancements

How to Get the Most Out of PFA

- **Use check-specific tuning parameters** to adjust *sensitivity of comparisons* if needed
  - To minimize customer configuration
    - Default parameter values constructed from in-house and external data
    - Some defaults changed via PTFs using customers' data

Parameter	Description
STDDEV	Increase value to decrease sensitivity. Not available on the Common Storage Usage check.
EXCEPTIONMIN	Increase value to decrease exceptions issued for relatively low rates. Not available on the Common Storage Usage check or Frames and Slots Usage check.
THRESHOLD	Increase value to decrease sensitivity. Common Storage Usage check only
STDDEVLOW	Increase value to decrease sensitivity for “too low” checking. Available on checks where “too low” checking is supported.
LIMITLOW	Defines the maximum rate where “too low” checking is performed Available on checks where “too low” checking is supported.

How to Get the Most Out of PFA (continued)

- **Use PFA check-specific parameters** to *affect other behavior*

Parameter	Description
COLLECTINT	Number of minutes between collections
MODELINT	Number of minutes between models • PFA automatically and dynamically models more frequently when needed • z/OS 1.12 default updated to 720 minutes. First model will occur within 6 hours (or 6 hours after warm-up)
COLLECTINACTIVE	Defines whether PFA should collect and model if check not active/enabled in IBM Health Checker for z/OS
DEBUG	Use only if IBM service requests it
CHECKLOW	z/OS 1.13 – Turns on/off "too low" checking with RTD for checks that support it
TRACKEDMIN	Requires a persistent job to have this minimum rate at the end of the warm-up in order to be tracked (where supported)
Health Checker parameters	For example, SEVERITY -- All PFA checks default = SEVERITY(MED): Eventual action WTO

How to Get the Most Out of PFA (continued)

- z/OS 1.12 – *Eliminate jobs* causing false positives
  - **Unsupervised learning** is the machine learning that PFA does automatically.
  - **Supervised learning** allows you to exclude jobs that are known to cause false positives. For example,
    - Exclude test programs that issue many LOGRECs and cause exceptions.
    - Exclude address spaces that issue many WTOs, but are inconsistent or spiky in their behavior and cause message arrival rate exceptions.

How to Get the Most Out of PFA (continued)

- z/OS 1.12 -- Implementing supervised learning
  - Supported by all checks except Common Storage Usage
  - Create EXCLUDED_JOBS file in the check's /config directory
    - Simple comma-separated value format
      - JobName,Systems,Date,Reason
    - Supports wildcards in both job name and system name
      - KKA*,*,04/05/2011,Exclude all KKA* jobs on all systems
  - Use f pfa,update,check(*check_name*) if PFA running
  - PFA creates an EXCLUDED_JOBS file for some checks during installation
  - See *z/OS Problem Management* for more information

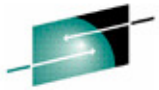
How to Get the Most Out of PFA (continued)

- Use a zAAP to offload PFA's Java Processing
- Start z/OS Resiliency functions at IPL
  - IBM Health Checker for z/OS
  - PFA
  - Runtime Diagnostics (z/OS 1.13)
- Automate the PFA IBM Health Checker for z/OS exceptions
  - Simplest: Add exception messages to existing message automation product
  - More complex: Use exception messages and other information to tailor alerts
  - See *z/OS Problem Management* for exceptions issued for each check
- Create a policy in an HZSPRMxx member for persistent changes
  - Not all check-specific parameters are required on an UPDATE of PFA checks!
    - UPDATE CHECK=(IBMPFA,PFA_COMMON_STORAGE_USAGE)
      PARM('THRESHOLD(3)')

How to Get the Most Out of PFA (continued)

- Get the latest PTFs!
  - Configuration value default changes
  - Comparison algorithm tuning changes
  - Changes to design
    - Exclude interactive users from being persistent jobs
    - Skip comparisons for ESQA
  - zFS growth problem – in progress (OA38376)
- Help us to make PFA's results better for everyone!

PFA Serviceability



Modify command to display status

STATUS examples:

f pfa,display

f,pfa,display,status

```
AIR017I 10.31.32 PFA STATUS
NUMBER OF CHECKS REGISTERED     : 5
NUMBER OF CHECKS ACTIVE         : 5
COUNT OF COLLECT QUEUE ELEMENTS: 0
COUNT OF MODEL QUEUE ELEMENTS   : 0
COUNT OF JVM TERMINATIONS       : 0
```

SUMMARY examples:

f pfa,display,checks

f pfa,display,check(pfa*),summary

```
AIR013I 10.09.14 PFA CHECK SUMMARY
                                  LAST SUCCESSFUL     LAST SUCCESSFUL
CHECK NAME                 ACTIVE      COLLECT TIME        MODEL TIME
PFA_COMMON_STORAGE_USAGE    YES     04/05/2008 10.01    04/05/2008 08.16

PFA_LOGREC_ARRIVAL_RATE     YES     04/05/2008 09.15    04/05/2008 06.32
(all checks are displayed)
```

DETAIL examples:

f pfa,display,check(pfa_logrec_arrival_rate),detail

f pfa,display,check(pfa_l*),detail

```
AIR018I 02.22.54 PFA CHECK DETAIL
CHECK NAME:  PFA_LOGREC_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 5
    SUCCESSFUL COLLECTION COUNT     : 5
    LAST COLLECTION TIME            : 04/05/2008 10.18.22
    LAST SUCCESSFUL COLLECTION TIME: 04/05/2008 10.18.22
    NEXT COLLECTION TIME            : 04/05/2008 10.33.22
    TOTAL MODEL COUNT               : 1
    SUCCESSFUL MODEL COUNT          : 1
    LAST MODEL TIME                 : 04/05/2008 10.18.24
    LAST SUCCESSFUL MODEL TIME      : 04/05/2008 10.18.24
    NEXT MODEL TIME                 : 04/05/2008 16.18.24
    CHECK SPECIFIC PARAMETERS:
       COLLECTINT                   : 15
       MODELINT                     : 360
       COLLECTINACTIVE              : 1=ON
       DEBUG                        : 0=OFF
       STDDEV                       : 10
       EXCEPTIONMIN                 : 25
   EXCLUDED_JOBS:
        (excluded jobs list here)
```

Backup Reference – Component Soft Failure detection

Component Examples: Detection, Identification of soft failures … Single system

Component	Features	Functions
GRS	Enhanced contention analysis for ENQ, Latch	Identify Blocker/Waiter, Deadly embraces, Job name, Creator ASID
	GRS Latch identity string	Associate name with Latch number
	WLM management of blocking units	Prevent deadlocks caused by starvation
	GRS ENF 51	Prevent exhaustion of common storage resulting from GRSQSCAN processing
UNIX System Services	Latch identity exploitation	Explanations for latch usage on D GRS
	XCF communication improvements (R13)	Detected lost messages in sysplex, via message ordering
	System Limits	Checks for buildup of processes, pages of shared storage (process & system level)
	D OMVS,WAITERS to diagnose file system latch contention (enhanced R13: file latch activity)	Identifies holders, waiters, latches, file device numbers, file inode numbers, latch set identifiers, file names, and owning file systems
JES2	JES2 Monitor	Assists in determining why JES2 is not responding to requests "Monitor" msgs issued for conditions that can seriously impact JES2 performance

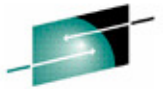
Details in backup section

Component Examples: Detection, Identification, recovery of soft failures … Single system

Component	Features	Functions
IOS	Missing Interrupt Handler	Detect incomplete I/O operations, within a policy driven time period (device, CU, fabric); recover, FFDC
	Identify systems sharing a reserve	Identify partner system sharing device D U,VOL= … D GRS,DEV= …
	Captured UCB protection	Prevent accidental overlays of real UCBs in SQA by Legacy applications
	I/O timing facility	Abnormally end I/O requests exceeding I/O timing limits for device; Hyperswap devices as well
	Detect & remove “Flapping Links”	Improved channel recovery (hardware)
	Dynamic Channel Path Management	WLM dynamically move channel paths from one CU to another, in response to workload changes
DFSMS	CAS contention detection	Identify, terminate service tasks beyond a monitored wait time
	VSAM RLS index traps	Checks the structure of all index CIs before writing them to DASD
	Media manager	Recover channel program error retry from I/O errors, using a lower level protocol

Details in backup section

Component Examples: Detection of soft failures … Sysplex



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

Details in backup section

Detection of Soft Failures on a z/OS image: GRS serialization

- Enhanced contention analysis for ENQ / Latch
  - D GRS,ANALYZE,BLOCKER / WAITER / DEPENDENCY
  - D GRS,ANALYZE,LATCH,BLOCKER / WAITER / DEPENDENCY
    - Blocker/Waiter, Deadly embraces, Job name, Creator ASID, etc.

- GRS Latch identity string
  - Associate name with latch number
  - Included in D GRS latch analysis responses
  - Exploited by USS, RRS, Logger, RACF

- GRS interacts with WLM to manage priority of blocking units of work
  - Prevent deadlocks causing starvation
  - WLM's "trickle" support ensures that critical work is given cycles gradually to resolve any deadlocks

- GRS monitor
  - ENF 51 generates blocks in common storage (SQA)
  - SRBs suspended due to stuck receiver (e.g., RMF)
    - Therefore too many requests can cause common storage outage
  - GRS piped the requests elsewhere to avoid exhausting common storage

- Exploits XCF Critical member support (see XCF critical member support)

Detection of Soft Failures on a z/OS image:
UNIX System Services serialization

- Latch identity explanations for the latches used by USS (R13)
  - FS: <fs name> … MOUNT … MessageQ ID=<msg-ID in decimal>
    - System traversing or modifying structures related to the message queue
- XCF communication improvements
  - Lost XCF message detection (R13)
    - Utilizes XCF message ordering to detect lost messages
    - Activate with parmlib option, SETOMVS LOSTMSG=ON/OFF
  - Member Gone detects stall, attempts fix; if takeover fails, initiates sysplex-wide dump
- USS System Limits (R10)
  - Checks for buildup of processes, pages of shared storage (process & system level)
  - When 85% process utilization is reached, WTO messages are issued
  - For example: MAXASSIZE, MAXCPUTIME, MAXFILEPROC, MAXPROCUSER, MAXQUEDSIGS, MAXTHREADS
  - Displayed via D OMVS,LIMITS
- DISPLAY OMVS,WAITERS to diagnose file system latch contention problems
  - Enhanced in R13 to show a table for file latch activity
  - Holders, waiters, latches, file device numbers, file inode numbers, latch set identifiers, file names, and owning file systems

Detection of Soft Failures on a z/OS image: IOS examples

Missing Interrupt Handler

- Incomplete I/O: Prevents an application or system outage due to an error in any one of the following places:
  - Device
  - Control Unit
  - Fabric
  - Operator/CE error (IML, cable pulls, etc…)
- Outage is prevented by:
  - Detecting when an I/O operation has not completed within a policy driven time period
  - Invoking system diagnostic routines to understand the scope of the error
  - Driving hardware and software recovery mechanisms
  - First Failure Data Capture

Identify sharing systems holding a reserve

- Start-pending MIH condition → D U,VOL= to identify device number
- D GRS,DEV=dddd to determine reserve status
- Identify other system with reserve, in message (IOS431I device reserve to CPU …)

Captured UCB protection

- Creates a temporary copy of UCBs for Legacy applications
- Prevents accidental overlays of real UCBs in SQA

I/O Timing Facility – Identify slow I/O response time

- Times the entire I/O request
  - If exceeds timing limit …
  - Abnormally ends I/O requests exceeding I/O timing limits for device
  - Application posted with permanent error, error logged to Logrec
- Facility can trigger a Hyperswap when I/O timeout occurs for a device monitored
  - Whether I/O operation should be terminated or started on the “swap TO” device

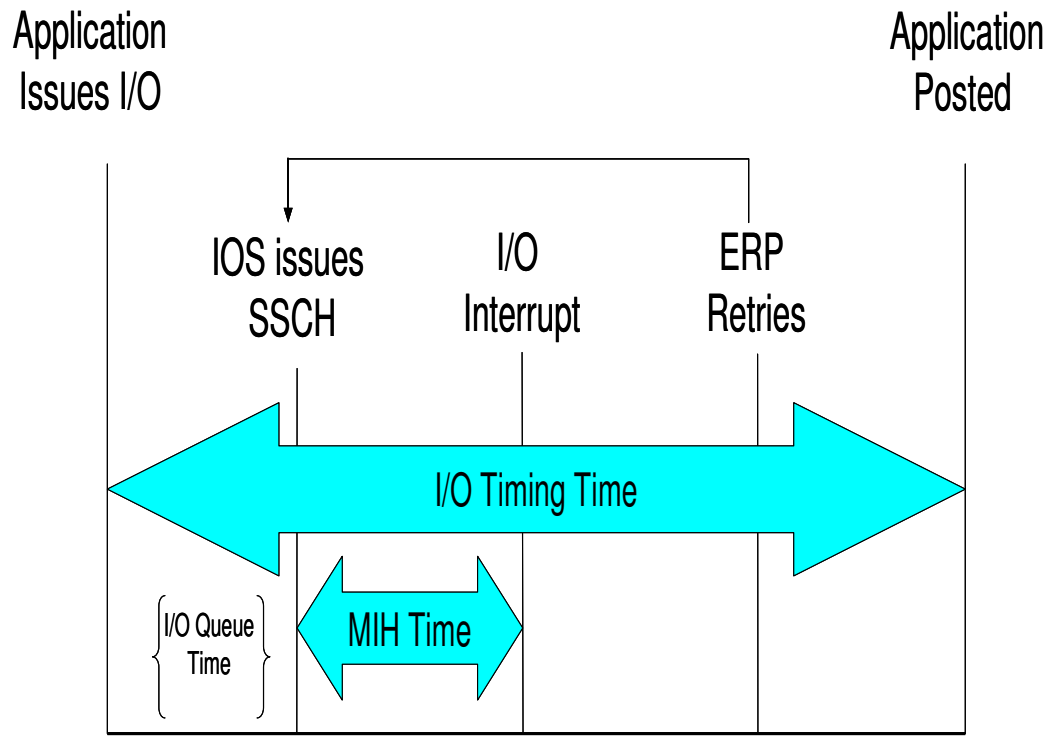


Figure 1 - MIH vs. I/O Timing

Improved Channel Recovery

- For frequently-occurring path errors, better to have hardware problem cause path taken offline than continue to cause problems
  - IOS recovery delays application I/O even when there are other paths

Proactively Removing Paths – Flapping Links

- Logic path between channel & control unit becomes available, unavailable multiple times over a short period
- Drives IOS recovery for all devices on the affected link
- May cause application I/O delays
- When channel detects that link has “flapped” 5-9 times in 5 minutes, it stops attempting to establish a logical path

Dynamic Channel Path Management

- Simplify I/O configuration definition task
- Static channel path definitions needed to be re-evaluated when workloads shift
- DCM lets WLM dynamically move channel paths from 1 CU to another, in response to workload changes
- Improve workload management
- DASD I/O resources are used more efficiently
- Improves Availability
- Foundation for auto-configuration
- Balance mode, Goal mode

Detection of Soft Failures on a z/OS image: DFSMS examples

- **CAS Contention Detection**
  - Runs as part of the CAS analysis task
  - Periodically checks the Catalog Address Space (CAS) service tasks list (every 30 seconds or upon request)
    - Based on a set wait time and reason class, determines those tasks which are beyond the wait time.
      - *Checks for service tasks that are active and waiting on the SYSZTIOT enqueue. It sets timer for each waiting task (10 min)*
    - Creates a symptom record for each task past the limit
    - Terminates some of the violating tasks, which were considered safe to terminate
- **VSAM RLS index traps**
  - Set the trap using a V SMS,MONDS command
  - Checks the structure of all index CIs before writing them to DASD.
    - If problem, abend is issued and write is avoided
- **Media Manager**
  - Channel program error retry from I/O errors, using a lower level protocol supported by the device
    - zHPF transport mode channel program
    - Command mode channel program with MIDAWs
    - Command mode channel program with IDAWs
  - Media Manager will retry the I/O with one of the lower level protocols

Detection of Soft Failures on a z/OS image: JES2 Monitor

- Assists in determining why JES2 is not responding to requests (single system)
- "Monitor" messages issued when conditions exist that can seriously impact JES2 performance (z/OS or JES2 issues)
- Automatically started when JES2 is started
- Results displayed via $JD STATUS command
  - Any conditions the monitor detected that could impact JES2
- Corresponding monitor address space for each JES2 address space
  - $JD MONITOR displays status info for each monitor task
  - Samples values at regular intervals
- Incident categories:
  - Normal processing
  - Tracking: processing time exceeds threshold
  - Alerts: Incident being tracked crosses a second (sampling) threshold
    - Exclusive incidents focus attention on primary incident
- Resource utilization
  - Low, high, average, current utilization
- $JD HISTORY displays up to 72 hours of resource utilization & CPU sample statistics

For more information, see JES2 Diagnosis book, GA22-7531

Detection of Soft Failures in a Sysplex: XCF stalled member support

- A system may appear to be healthy with respect to XCF system status monitoring:
  - Updating status in the sysplex CDS
  - Sending signals
- But is the system actually performing useful work?
  - There may be critical functions that are non-operational, making the system unusable
    - Could induce sympathy sickness elsewhere in the sysplex
    - Waiting for a response; waiting to get an ENQ, latch, lock
  - Causes include
    - Dead system
    - Loops (spin, SRB)
    - Low weighted LPAR
    - Loss of a Coupling Facility
- *Long periods of sympathy sickness may have a greater negative impact on the sysplex than termination of an XCF group member, address space, structure connector, or even a system*
- Action should be taken to restore the system to normal operation OR remove it to avoid sympathy sickness
  - Helps reduce the incidence of sysplex-wide problems that can result from unresponsive critical components

Detection of Soft Failures in a Sysplex: Sysplex Failure Management (SFM)

- Single system “Sick but not dead” issues can escalate to cause sysplex-wide problems
  - Typically holds resources needed by other systems in the sysplex
- Implements best practices of a resilient sysplex
- Enables automatic, timely, corrective action to be taken when applications or systems appear to be causing sympathy sickness
- Protects your sysplex when your operators and/or your automation are inattentive, unable, or incapable of resolving the problem
- Define an SFM policy to help meet availability and recovery objectives
  - Applications or systems are not permitted to linger in an extremely sick state such that they adversely impact other systems in the sysplex
  - Applications or systems are not terminated prematurely
  - Failure Detection Interval (FDI): amount of time a system is permitted to appear unresponsive (Not updating heartbeat, Not sending signals)
- Use of BCPii to determine a system is down dramatically improves this detection (over use of heartbeat) (see BCPii topic)

Detection of Soft Failures in a Sysplex: SFM

- System Not Updating Status, Not Sending Signals
  - ISOLATETIME(0)
    - n seconds after the FDI exceeded fencing is initiated by all systems
    - Commands are sent across the coupling facility to the target system and I/O is isolated
    - After fencing completes successfully, sysplex partitioning continues
- System updating status, not sending signals
  - Loss of connectivity: CONNFAIL(YES)
    - SFM determines sets of systems that do have full signal connectivity
    - Selects a set with largest combined system weights
    - Systems in that set survive, others are removed
    - Ensure the weights assigned to each z/OS system adequately reflect the relative importance of the system
- System Not Updating Status, But IS Sending Signals
  - SSUMLIMIT(900)
    - Indicates the length of time a system can remain in the state of not updating the heartbeat and sending signals
    - *This is the amount of time a system will remain in a "semi-sick" state.*
    - Once the SSUMLIMIT has been reached the specified action will be initiated against the system
- Sysplex Member Stalled
  - MEMSTALLTIME (600-900)
    - Enable XCF to break out of an XCF signaling traffic jam
    - SFM automatically starts removing the largest build-up, adversely impacting other systems in the sysplex
    - Action XCF will take: terminate the stalled member with the highest quantity of signals backed up

Detection of Soft Failures in a Sysplex: SFM

Taking Action When a Connector Does Not Respond

- Connectors to CF structures participate in processes, respond to relevant events
  - XES monitors the connectors, reports unresponsive connectors
  - Users of the structure may hang until offending connector responds or is terminated
- CFSTRHANGTIME (z/OS R12)
  - How long the system should allow a structure hang condition to persist before taking action
  - Enables XES to automatically take action if a connector does not respond to a structure event in a timely fashion
- XES corrective actions:
  - Stop rebuild
  - Force user to disconnect
  - Terminate connector task, address space or system
  - RAS: ABEND026 dumps collected
  - CFSTRHANGTIME(900-1200)

Detection of Soft Failures in a Sysplex: SFM

BCPii: Avoid waiting the FDI+ if the system is truly dead !

- BCPii allows XCF to query the state of other systems via authorized interfaces through the support element and HMC network
- Benefits:
  - XCF can detect and/or reset failed systems more quickly
  - Works in scenarios where fencing cannot work
    - CEC checkstop or powered down
    - Image reset, deactivated, or re-IPLed
    - No CF
  - Eliminates the need for manual intervention, which may lead to data corruption problems
  - **Reduction in sympathy sickness time**
  - ***Set this up. It is a critical component of Resiliency AND Soft Failure Avoidance***

Detection & Prevention of Soft Failures in a Sysplex: Critical Member support

- A Critical Member is a member of an XCF group that identifies itself as "critical" when joining its group
- If a critical member is "impaired" for long enough, XCF will eventually terminate the member
  - Per the member's specification: task, space, or system
  - SFM parameter MEMSTALLTIME determines "long enough" before terminating the stalled member with the highest quantity of backed up signals

- **GRS declares itself a "critical member"**
  - If GRS cannot perform work for as long as the FDI, GRS is said to be "impaired"
  - XCF will remove a system from the sysplex if GRS on that system becomes "impaired" (key tasks not operating) to avoid sympathy sickness
    - Based on SFM MEMSTALLTIME(n)
    - For MEMSTALLTIME(NO), N=MAX(FDI, 120 seconds)

Health Check details

Health Checker: Soft Failure avoidance Important examples

Component	Health Check	Functions
XCF	XCF_CDS_SPOF	Evaluates primary & secondary CDS configuration to determine if Sysprog inadvertently created a single point of failure
	XCF_SFM_SUM_ACTION	Checks ISOLATETIME value, to allow SFM to fence and partition a system without operator intervention and without undue delay.
	XCF_SFM_SUMLIMIT	Checks status update missing (SUMLIMIT) value
	XCF_SFM_ACTIVE	Verifies SFM active, policy values
	XCF_SFM_CFSTRHANGTIME	Verifies CFSTRUCTURE hang time
	XCF_SFM_CONNFAIL	Threshold for loss of connectivity
RACF	RACF_GRS_RNL	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)

Details in backup section

Health Checker: Soft Failure avoidance examples

Component	Health Check	Functions
Serviceability	DAE_SUPPRESSING	DAE suppresses duplicate SVC dumps so that system resources (processor cycles and dump space) are not used for a dump which provides little or no additional diagnostic value
	SVA_AUTOIPL_DEFINED	Check whether Program-Directed IPL and not GDPS, and whether AUTOIPL policy is active
	SVA_AUTOIPL_DEV_VALIDATION	Validates SADMP, MVS IPL devices
UNIX System Services	USS_PARMLIB	Validate current system against parmlib IPL'd with Remind you to update parmlib (due to dynamic changes)
	USS_CLIENT_MOUNTS	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
	USS_FILESYS_CONFIG	Checks if mount attribute access is read only; whether HFS's in Sysplex root

Details in backup section

Health Checker: Soft Failure avoidance examples

Component	Health Check	Functions
IOS	IOS_CAPTUCB_PROTECT	UCB capture protection is enabled, allowing UCBs to be temporarily copied to 24-bit storage for legacy software access
	IOS_CMRTIME_MONITOR	Detects if any control units in the system are reporting inconsistent average initial command response (CMR) time (round trip delay) for their attached channel paths. Exception issued when a CU has a path with highest avg CMR time greater than a threshold/ratio
System Logger	IXGLOGR_STRUCTUREFULL	Primary structure full; need to offload
	IXGLOGR_ENTRYTHRESHOLD	High number of entries in element pools
	IXGLOGR_STAGINGDSFULL	Full staging data space

Details in backup section

z/OS Health Check: *Example Categories*

Category	Examples
Detect Single points of failure	VSAMRLS_SINGLE_POINT_FAILURE (SHCDS data sets) XCF_CDS_SPOF (XCF Couple Data Sets) XCF_CF_CONNECTIVITY (CF links, SPOF)
Security	RACF_GRS_RNL (for RACF datasets) SDSF_CLASS_SDSF_ACTIVE (SDSF settings)
Address space checks	IEA_ASIDS (number of ASIDs remaining) IEA_LXS (number of LX's remaining) SUP_LCCA_ABOVE_16M
GRS	GRS_MODE (system configured in STAR mode) GRS_SYNCHRES (GRS synchronous reserve processing enabled) GRS_CONVERT_RESERVES (reserves converted to ENQs)
I/O	IOS_CAPTUCB_PROTECT IOS_CMRTIME_MONITOR (Check for inconsistent average initial command response (CMR)) IOS_MIDAW (MIDAW enabled)

z/OS Health Check: *Example Categories*

Category	Examples
Optimal component settings	ALLOC_* (Allocation) CNZ_* (Consoles) CSRES (Comm Server), CSTCP_* (TCP/IP) SDSF_*, …
Sysplex configuration	XCF_* XCF_CF_* CSTCB_* RRS_* IXGLOGR_* VSAMRLS_* XCF_SFM_* CNZ_* Etc.

z/OS Health Check: *Example Categories*

Category	Examples
Serviceability (Dump, Trace options)	SDUMP_AVAILABLE SDUMP_AUTO_ALLOCATION (auto-alloc SDUMP data sets) CSTCP_SYSTCPIP_CTRACE (CTRACE active, options) CSVTAM_VIT_SIZE (VTAM Internal Trace table size) CSVTAM_VIT_DSPSIZE (VTAM Internal Trace) SVA_AUTOIPL_DEFINED SVA_AUTOIPL_DEV_VALIDATION DAE_SHAREDSN DAE_SUPPESSING
Buffer sizes, storage limits	CSTCP_TCPMAXRCVBUFRSIZE CSVTAM_CSM_STG_LIMIT VSAMRLS_CFCACHE_MINIMUM_SIZE XCF_MAXMSG_NUMBUF_RATIO RSM_MEMLIMIT RSM_MAXCADS RSM_AFQ RSM_REAL RSM_RSU VSM_*

z/OS Health Check: *Example Categories*

Category	Examples
Hardware	SUP_HIPERDISPATCH (Verify Hiperdispatch enabled) SUP_HiperdispatchCPUConfig (monitors the number of CPUs installed and Hiperdispatch state of the system)
Other component specifics	Console configuration HSM control data set backups JES2 ready to upgrade Reconfiguration SMS CDS configuration System logger Staging data sets full, entry thresholds, structure full USS/ zFS: File system issues VSAM RLS: false contention, monitor contention, monitor unresponsive CICS regions, TVS enabled
Migration checks	

Health Checker: Soft Failure avoidance examples

- **DAE_SUPPRESSING**
  - DAE suppresses duplicate SVC dumps so that system resources (processor cycles and dump space) are not used for a dump which provides little or no additional diagnostic value
  - IBM recommendation is to activate this function.
    - If turned off, then health checker will issue an exception to alert the team to this sub optimal configuration.

- **XCF_CDS_SPOF**
  - z/OS uses two coupling data sets (CDS) to manage a parallel sysplex, primary and alternative.
  - This check evaluates the I/O configuration to determine if the I/O configuration has inadvertently created a single point of failure (SPOF) when accessing the data on the primary and alternative CDS.
  - Alternative CDS created to handle a problem with a switch or a storage device.

- **SVA_AUTOIPL_DEFINED, SVA_AUTOIPL_DEV_VALIDATION**
  - Check whether environment can support AUTOIPL, whether active
  - Validates SADMP, MVS IPL devices

Health Checker: Soft Failure avoidance examples

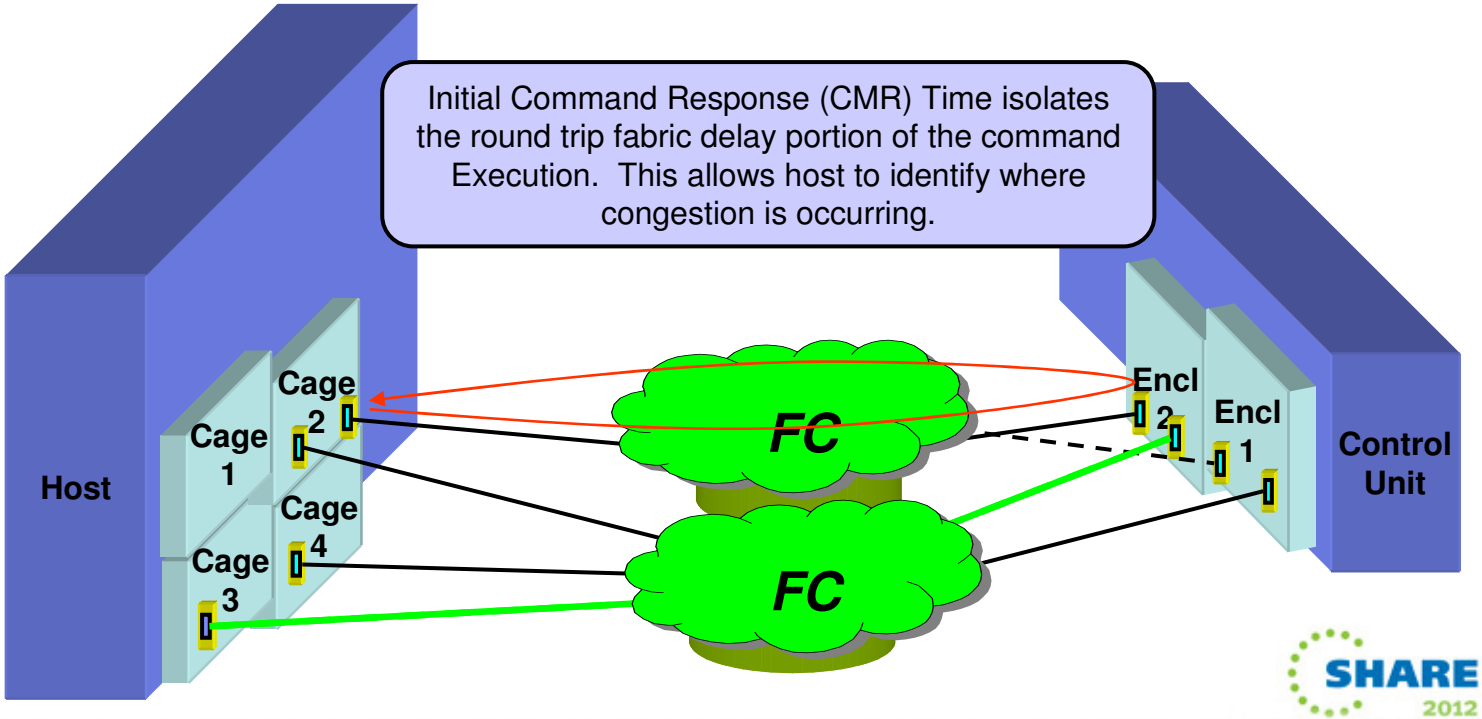
- **System Logger**
  - IXGLOGR_STRUCTUREFULL
    - Primary structure full; need to offload
  - IXGLOGR_ENTRYTHRESHOLD
    - High number of entries in element pools
  - IXGLOGR_STAGINGDSFULL
    - Full staging data space

- **UNIX System Services**
  - USS_PARMLIB
    - Validate current system against parmlib IPL'd with
    - Remind you to update parmlib (due to dynamic changes)
  - USS_CLIENT_MOUNTS
    - 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
    - Check if function ship but could be done locally (performance awareness)
  - USS_FILESYS_CONFIG
    - Checks if mount attribute access is read only
    - HFS's in Sysplex root

- **Sysplex Failure management**
  - Examines / validates SFM values
    - XCF_SFM_ACTIVE
    - XCF_SFM_CFSTRHANGTIME
    - XCF_SFM_CONNFAIL
    - XCF_SFM_SSUMLIMIT
    - XCF_SFM_SUM_ACTION

Health Checker: Soft Failure avoidance examples

- IOS_CAPTUCB_PROTECT
  - UCB capture protection is enabled, allowing UCBs to be temporarily copied to 24-bit storage for legacy software access
- IOS_CMRTIME_MONITOR
  - Fabric issues have resulted in unacceptable I/O service times
    - RMF device activity reports show average service times to be higher than normal
    - I/O queuing reports show abnormally high “initial command response” times on a subset of the paths to a device (5x)



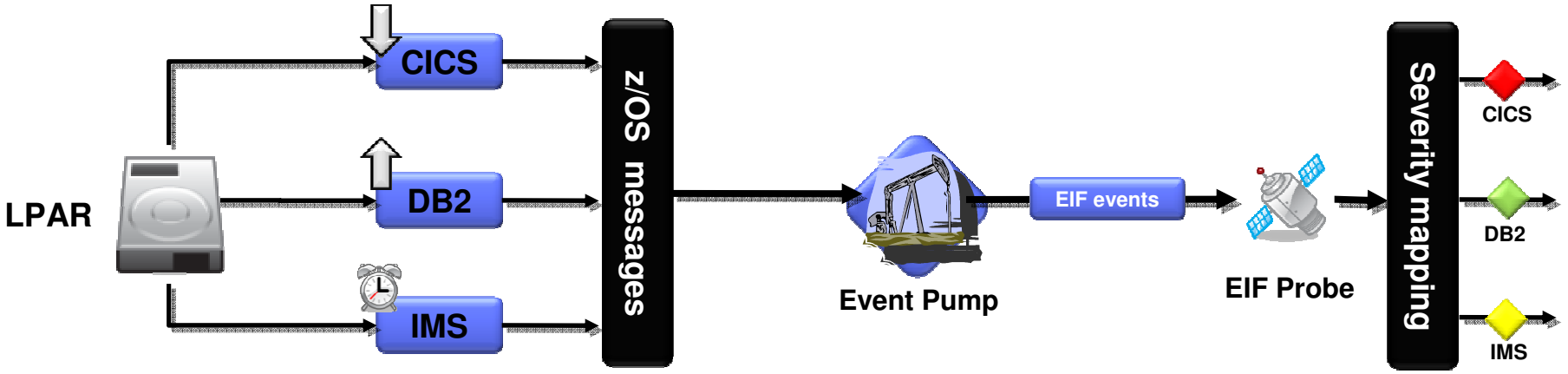
Tivoli Management Products

- Tivoli Management Products integrate
  - Soft Failures detected by PFA
    - Health Check exceptions surfaced by zMC (to be supported on *Omegamon XE*)
    - *Tivoli System Automation* policy to control of corrective actions
  - Performance issues detected by *Omegamon*
    - Evaluate entire software stack
    - Customer-defined model, selection of critical events
  - *Netcool/OMNIbus* provide centralized monitoring of Health Check Alerts, Performance, Situations, Network activity, etc.

Event Pump for z/OS

Each subsystem writes messages to z/OS

- These messages may contain **state** and **status** information
- The Event Pump parses the message, interprets the resource information, and converts the message to an EIF event



OMNIbus
TBSM
TIP

How PFA Detects Soft Failures

- Causes of “sick, but not dead”
  - ***Damaged systems***
    - Recurring or recursive errors caused by software defects anywhere in the software stack
  - Serialization
    - Priority inversion
    - Classic deadlocks
    - Owner gone
  - ***Resource exhaustion***
    - Physical resources
    - Software resources
  - Indeterminate or unexpected states

- Predictive failure analysis uses
  - *Historical data*
  - *Machine learning and mathematical modeling*

  to detect abnormal behavior and the potential causes of this abnormal behavior
- Objective
  - Convert “sick, but not dead” to a correctable incident