Detecting and Diagnosing Soft Failures Using z/OS Predictive Failure Analysis and Runtime Diagnostics

Bob Abrams
STSM, z/OS Design & Development (RAS, PD)
Poughkeepsie, NY
abrams@us.ibm.com

(Thanks to Karla Arndt, PFA Development)

Trademarks

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

DB2*
DB2 Connect
DB2 Universal Database
e-business logo
GDPS*
Geographically Dispersed Parallel Sysplex
HyperSwap
IBM*
IBM eServer
IBM logo*
Parallel Sysplex*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Intel is a registered trademark of the Intel Corporation in the United States, other countries or both.

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

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.

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

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

* 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.

All 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

- Soft failures defined
- How PFA detects and reports soft failures
- The PFA checks
- What's new in z/OS 1.13!
- How Runtime Diagnostics helps you diagnose soft failures
- How to get the most out of PFA
- Summary

So, 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

Problem Determination in a Complex Environment

Installation Pain Points

Risk to the business

- The impact of the symptoms
- Risk of recurrence
- Impact in getting system stabilized
- Mean time to recovery too long

Complexity of performing the task

Troubleshooting a live system and recovering from an apparent failure

Data collection very time-consuming

Significant skill level needed to analyze problems, interact with IBM and ISVs to obtain additional diagnostic info

Requirement Areas

Detect "sick, but not dead" event **BEFORE** it causes problems

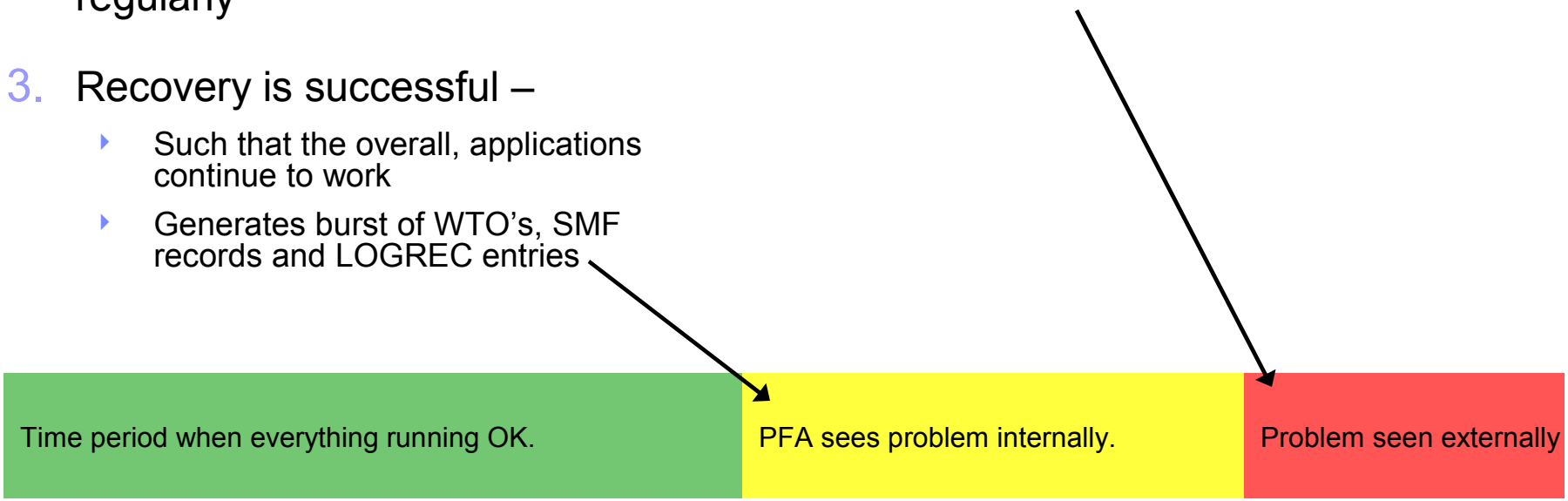
Diagnose the cause in **real time** to allow operations to mitigate event inquiries

Manage / capture data to determine cause of problem

- Allow problem to be fixed to prevent recurrence

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”



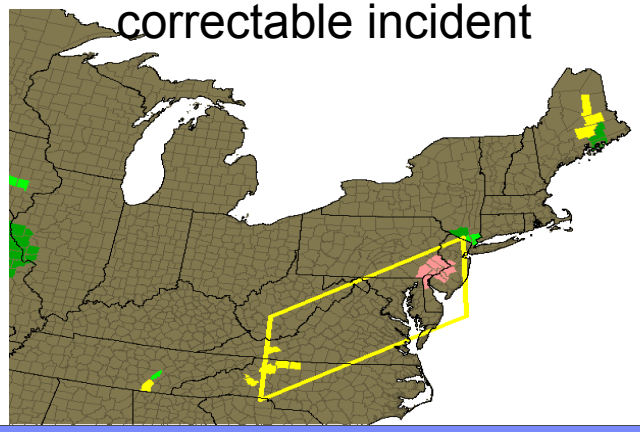
This is a hypothetical problem which is a combination of multiple actual problems

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

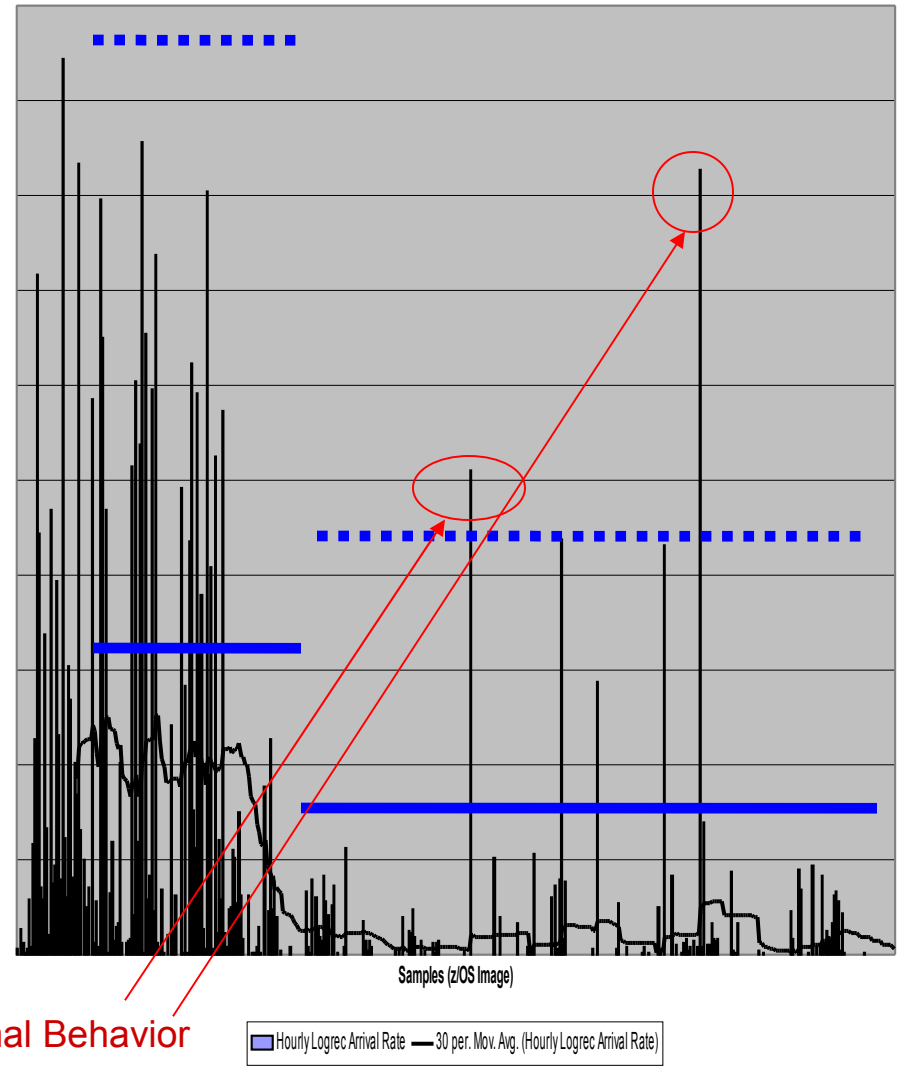
- Predictive failure analysis uses
  - *Historical data*
  - *Machine learning and mathematical modeling*

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

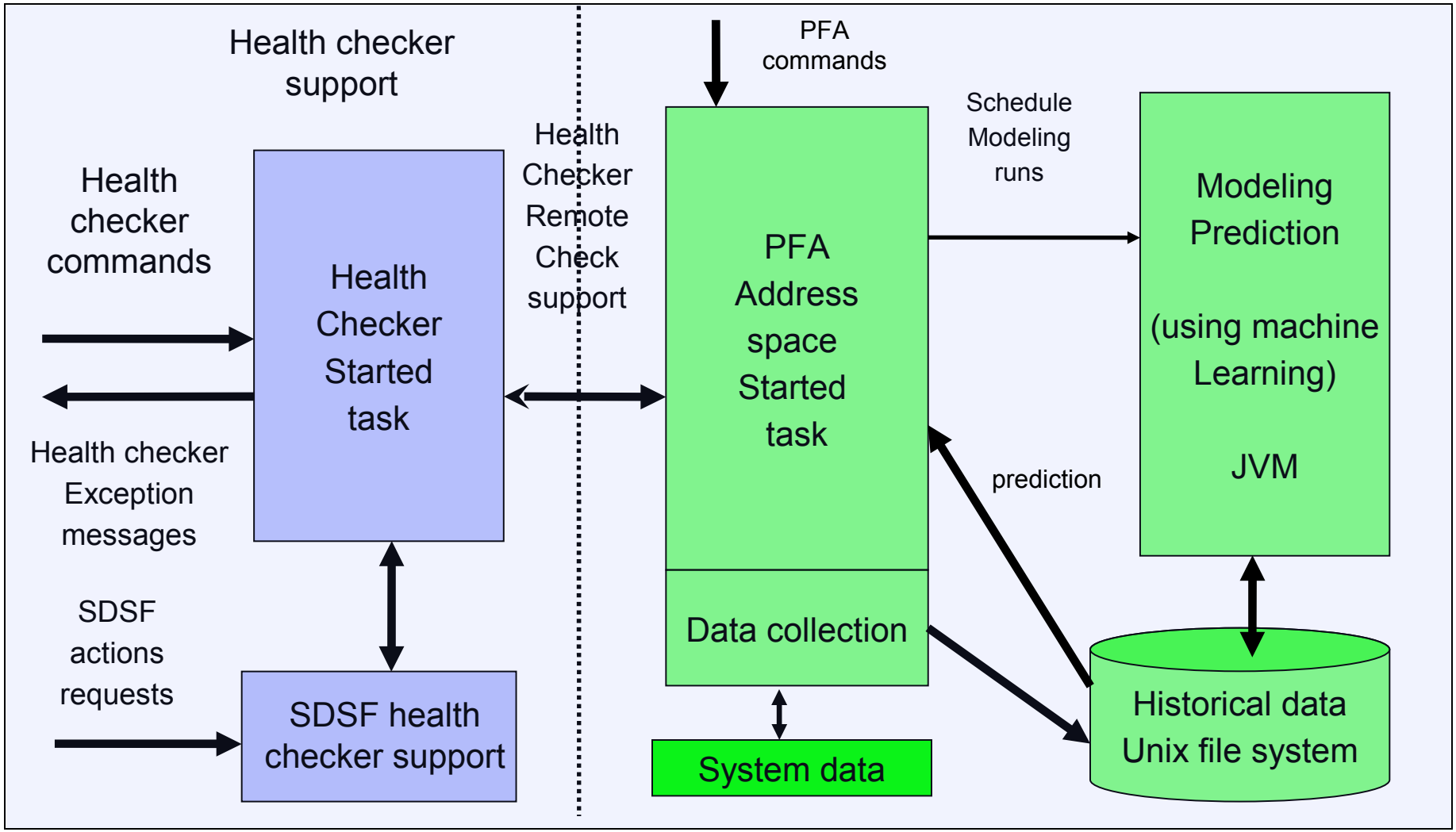


How PFA Determines Normal Values

- **Behavior of z/OS system is a function of**
  - Workload
  - Type of work
  - Hardware and Software configuration
  - System automation
  - …
- **Same type of work runs at approximately same time**
- **Use historical data to calculate future or expected value to eliminate variables**
  - Hardware and Software configuration
  - System automation
  - …
- **Expected value = fn(workload, time)**
- **Future value = fn(workload, time projected into future)**
- **Cluster metric by time to calculate expected or future value**
  - Can compare different time ranges such as 1 hour ago, 24 hours ago, 7 days ago



PFA's Relationship to IBM Health Checker for z/OS



PFA Functions and Dependencies

- PFA is its own address space – recommend to start at IPL
  1. *Collects data* from the system
  2. *Models the data* from the system to create predictions
  3. *Performs comparisons* on current vs. predictions
  4. *Issues exceptions or "OK" messages and reports* via IBM Health Checker for z/OS
- Dependencies
  - IBM Health Checker for z/OS – recommend to start at IPL
  - z/OS UNIX file system – where we store the data
  - Java (31-bit only) – used primarily during modeling
    - Java 1.4 or later for z/OS 1.10
    - Java 5.0 or later for z/OS 1.11 and z/OS 1.12
    - zAAP eligible (recommended)
  - z/OS 1.13 – The Runtime Diagnostics address space must be active.

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)

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

- *Best practices and more information* in *z/OS Problem Management*

- *Invokes Runtime Diagnostics to check for hung address spaces (R13); RTD validates and suggests next steps*

- *PFA exceptions are telling you "look at me", not "I am broken"*

- Customer experience feedback, SHARE 3/2012

Example Report: z/OS 1.12 Common Storage Usage Report

- **PFA_COMMON_STORAGE_USAGE** → predicts *exhaustion of common storage* by the z/OS image
- Available in SDSF (s.ck)
- Heading information
  - Configuration and status
  - Current and predicted information for metric
- Top predicted users
  - Tries to pinpoint potential villains
- Other information
  - CSA – expansion information
  - IBM Health Checker for z/OS message in its entirety

```
                Common Storage Usage Prediction Report
(heading information intentionally omitted)
                                                 Capacity When   Percentage
Storage     Current Usage     Prediction          Predicted     of Current
Location     in Kilobytes   in Kilobytes       in Kilobytes     to Capacity
__________ _______________ _______________ _______________ _____________
*CSA                  2796            3152            2956             95%
SQA                    455             455            2460             18%
CSA+SQA               3251            3771            5116             64%
ECSA                114922          637703          512700             22%
ESQA                  8414            9319           13184             64%
ECSA+ESQA           123336          646007          525884             23%

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:
  Job          Storage     Current Usage     Predicted Usage
  Name         Location    in Kilobytes      in Kilobytes

  __________   ________    ________________  ________________
  JOB3         *CSA                  1235              1523
  JOB1         *CSA                   752               935
  JOB5         *CSA                   354               420
  JOB8         *CSA                   152               267
  JOB2         *CSA                    75                80
  JOB6         *CSA                    66                78
  JOB15        *CSA                    53                55
  JOB18        *CSA                    42                63
  JOB7         *CSA                    36                35
  JOB9         *CSA                    31                34

* = Storage locations that caused the exception.
```

LOGREC Arrival Rate Prediction Report

- **PFA_LOGREC_ARRIVAL_RATE** → detects *a damaged system* by predicting and comparing LOGREC arrival rates in a collection interval

- Exceptions produced for any key grouping for any time range

- "Jobs having LOGREC arrivals in last collection interval"
  - Lists the jobs contributing to the arrival count.
  - Only displayed if the arrival count > 0.

```
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
```

Frames and Slots Usage Prediction Report

- **Frames and slots usage check** → detects a *damaged system by predicting resource exhaustion* by detecting abnormal increased usage of frames and slots by persistent address spaces
- "Address spaces with the highest increased usage"
  - Lists the jobs whose frames and slots usage recently increased the most
  - Sorted by expected usage; List exists even when no problem
- Exception raised when one or more jobs use substantially more frames and slots than expected
  - Only jobs causing exception listed when exception produced

```
            Frames and Slots Usage Prediction Report
(heading information intentionally omitted)

Address spaces with the highest increased usage:

   Job                    Current Frames      Expected Frames
   Name        ASID       and Slots Usage     and Slots Usage
   ________    ____    ________________       ________________
   ZFS         0029                12223                 12329
   XCFAS       0048                 1593                  1601
   VTAMOSR3    0027                 1885                  1881
   TRACE       0036                  367                   367
   SMS         0025                  682                   687
```

Message Arrival Rate Prediction Report

- **Message arrival rate check** → detects *damaged address spaces or a damaged LPAR* by tracking WTO and WTORs normalized by CPU across time ranges
- Perform comparisons after every collection rather than on an INTERVAL schedule in IBM Health Checker for z/OS
- An appropriate report is printed for each type of exception. Example "no problem" report and "total system" exception report shown

```
                    Message Arrival Rate Prediction Report
(heading lines intentionally omitted)
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:
                                              Predicted Message
                                                Arrival Rate
  Job                    Message
  Name      ASID          Arrival
                             Rate        1 Hour       24 Hour         7 Day
  ________ ____  ____________  ____________  ____________  ____________
  TRACKED1 001D         58.00         23.88         22.82         15.82
  TRACKED2 0028         11.00          0.34         11.11         12.11
  TRACKED3 0029         11.00         12.43          2.36          8.36
...
```

z/OS 1.12 – Many Enhancements

- **SMF arrival rate check** → detects a *damaged system* based on an SMF arrival rate (normalized by CPU) across time ranges that is too high
- Common storage usage and LOGREC arrival rate check **performance improvements**
- Checks enhanced to **improve dynamic modeling** and comparison algorithms
- **Supervised learning** → Excluded jobs list
- **Usability** – One ini file for all checks
- **Serviceability** – Separate log files and files and report data copied at exception to EXC_*timestamp* directory

z/OS 1.13 PFA_ENQUEUE_REQUEST_RATE Check

- Detects a *damaged address space* or *damaged system* by comparing the number of enqueue requests per CPU second to the rate expected.
- Enqueue request rate = Number of enqueues requested / CPU Utilization
- Two categories compared across three time ranges
  - *"Chatty" persistent* address spaces tracked individually and total system rate
  - 1 hour, 24 hour, and 7 day comparisons
- Ignores first hour after IPL and last hour prior to shutdown

```
                Enqueue Request Rate Prediction Report

     (Heading information intentionally omitted.)

     Enqueue request 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:

                                                Predicted Enqueue
                                                   Request Rate
       Job                 Enqueue
       Name      ASID      Request Rate      1 Hour       24 Hour         7 Day
       ________  ____  _____________  _____________  _____________  _____________
       TRACKED1  001D          58.00          23.88          22.82          35.82
       TRACKED2  0028          11.00          10.34          11.11          12.11
       TRACKED3  0029          11.00          12.43          12.36           8.36
```

z/OS 1.13 PFA_JES_SPOOL_USAGE Check

- Detects a *damaged address space or system* based on persistent jobs usage of the number of track groups (JES2 only)
- Models 15 persistent jobs with the highest *increase* in their track group usage from one collection to the next
- The exception is issued based on an *unexpected increase in the number of track groups used* from one collection to the next (rather than the number of track groups used).
- The current number of track groups used is provided as additional information.

JES Spool Usage Prediction Report

(Heading information intentionally omitted.)

Address spaces causing exception:

Job Name	ASID	Current Change in Number of Track Groups Used	Expected Change in Number of Track Groups Used	Current Number of Track Groups Used
JOB1	0019	252	10	892
JOB55	000E	129	3	400

How to Get the Most Out of PFA

- Use check-specific tuning parameters to adjust sensitivity of comparisons if needed
- Use check-specific parameters to affect other behavior
- z/OS 1.12 – Eliminate jobs causing false positives
  - Create EXCLUDED_JOBS file in the check's /config directory
    - /u/pfauser/PFA_LOGREC_ARRIVAL_RATE/config/EXCLUDED_JOBS
    - Comma-separated value file
      - Jobname,system,date_time,reason_added
      - Jobname and system name are required
    - Sample in /usr/lpp/bcp/samples/PFA

How to Get the Most Out of PFA (continued)

- 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
- Start IBM Health Checker for z/OS at IPL
- Start Runtime Diagnostics at IPL (z/OS 1.13)
- Start PFA at IPL
- 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)')
- Get the latest PTFs

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)
```

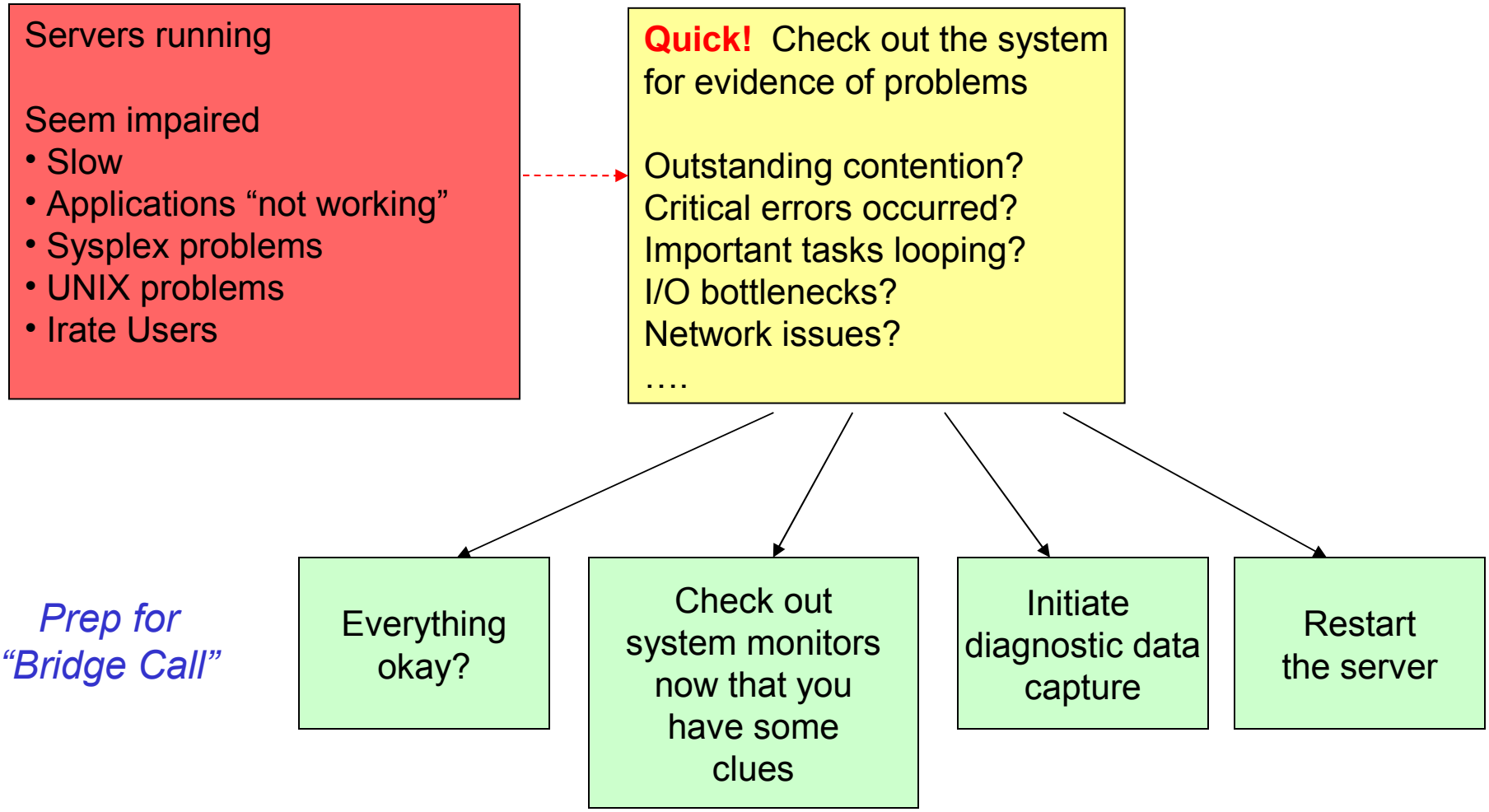
DETAIL examples:

f pfa,display,check(pfa_logrec_arrival_rate),detail

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

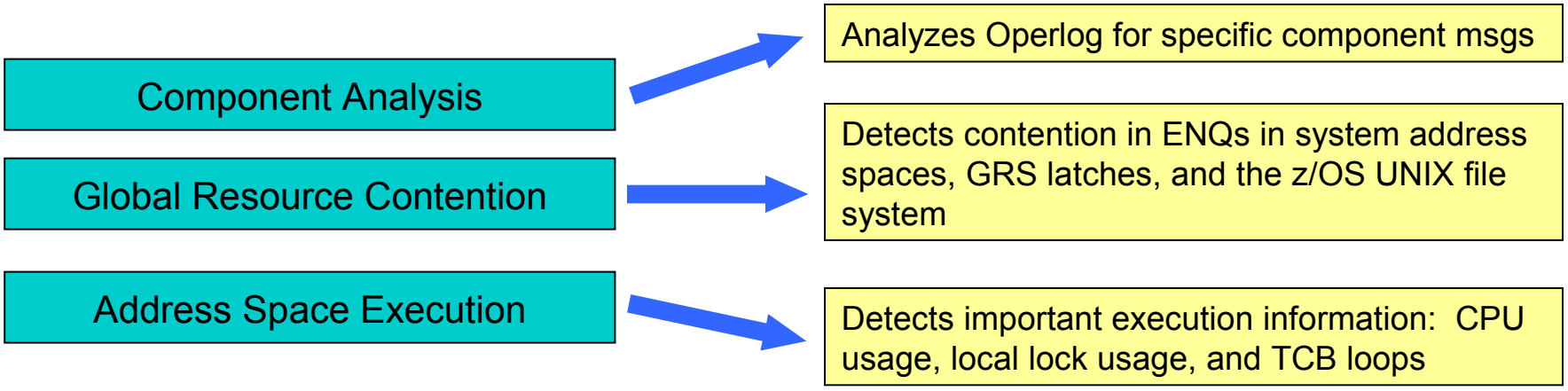
```
AIR018I 02.22.54 PFA CHECK DETAIL
CHECK NAME:  PFA_LOGREC_ARRIVAL_RATE
    ACTIVE                          : YES
    TOTAL COLLECTION COUNT          : 5
    SUCCESSFUL COLLECTION COUNT     : 5
    LAST COLLECTION TIME            : 04/05/2008 10.18.22
    LAST SUCCESSFUL COLLECTION TIME: 04/05/2008 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)
```

What if a problem or soft failure happens … is this typical?



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 Benefits

- Reduces the skill level needed by a system programmer for investigating soft failures
  - Provides timely, comprehensive analysis at a critical time period
  - *Also great productivity aid for experienced system programmers!*
- Allows you to *quickly discover next actions* to take such as
  - 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
- 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 Invocation

- z/OS 1.12 – Started task – “Run” the analysis via a START command
  - START HZR,SUB=MSTR
  - Invokes HZR PROC
  - Will only run on R12 system, but other systems in the Sysplex do not need to be R12
  - Can override HZROUT to specify a data set, for example:
    - //HZROUT DD DISP=SHR,DSN=MY.DATA
    - START HZR,SUB=MSTR,DSN=MY.DATA,DISP=SHR
- z/OS 1.13 – Address space – started with the START command above
  - Address space needs to be available for PFA integration
    - Recommend to start address space at IPL
  - “Run” the analysis via a MODIFY command
    - f hzr,analyze
  - Migration Action: If you used Runtime Diagnostics in z/OS 1.12, ensure you update the hzrproc to point to PGM=HZRINIT instead of PGM=HZRMAIN.

Runtime Diagnostics Invocation (continued)

- The output of Runtime Diagnostics is a multi-line WTO
  - Can also be directed to a sequential dataset using HZROUT DD
- SYSNAME option targets system other than HOME
  - Operlog and ENQ analysis are done for specified system
    - Operlog is suggested to allow message analysis
  - Example: OPTIONS=(SYSNAME=SYS2)
  - z/OS 1.12 – SYSNAME option on START command
  - z/OS 1.13 – SYSNAME option on MODIFY command
- DEBUG option for use under IBM Service guidance
  - Takes a dump to help debug analysis
  - Options specific to type of analysis and when found or not found
  - Example: OPTIONS=(DEBUG=(LOOP,NOENQ))
  - z/OS 1.12 – DEBUG option on START command
  - z/OS 1.13 – DEBUG option on MODIFY command

Runtime Diagnostics Output

- Success

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 974
SUMMARY: SUCCESS
 REQ: 001 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 11:30:57
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 05 - PRIORITIES: HIGH:05  MED:00  LOW:00
  TYPES: CF:04
  TYPES: HIGHCPU:01
```

- Qualified Success – Example of Operlog not connecting

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 751
SUMMARY: QUALIFIED SUCCESS - SOME PROCESSING FAILED
 REQ: 001 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 11:25:55
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 02 - PRIORITIES: HIGH:02  MED:00  LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01
  PROCESSING FAILURES:
   OPERLOG....IXGCONN REQ=CONNECT ERROR.......RC=00000008 RS=0000080B
```

Runtime Diagnostics: **Critical Message Analysis**

- Component-specific, critical messages in OPERLOG
  - "Needles in a haystack"
  - Looks one hour back, if available
  - For some messages, additional analysis done
    - Groups related messages into a single event
    - Weeds out shortage and relieved critical messages
    - In some cases, will only show last message if a critical message for the same resource name is repeated, say every 10 minutes
  - Message summary found listed in Runtime Diagnostics output

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

Runtime Diagnostics: **ENQ Contention Checking**

- Looks for a system address space that is an ENQ “waiter” for over 5 seconds
- Lists both waiter and blocker
- Equivalent to D GRS,AN,WAITER

```
f hzr,analyze
HZR0200I RUNTIME DIAGNOSTICS RESULT 581
SUMMARY: SUCCESS
 REQ: 004 TARGET SYSTEM: SY1      HOME: SY1      2010/12/21 - 13:51:32
 INTERVAL:  60 MINUTES
 EVENTS:
  FOUND: 04 - PRIORITIES: HIGH:04  MED:00  LOW:00
  TYPES: HIGHCPU:01
  TYPES: LOOP:01 ENQ:01 LOCK:01
------------------------------------------------------------------------
EVENT 01: HIGH - ENQ          - SYSTEM: SY1      2010/12/21 - 13:51:32
ENQ WAITER  - ASID:0038 - JOBNAME:IBMUSER2 - SYSTEM:SY1
ENQ BLOCKER - ASID:002F - JOBNAME:IBMUSER1 - SYSTEM:SY1
QNAME: TESTENQ
RNAME: TESTOFAVERYVERYVERYVERYLOOOOOOOOOOOOOOOOOOOOONGRNAME1234567...
  ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
 ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND
 ACTION: ASIDS.
```

Runtime Diagnostics: **Local Lock Suspension**

- Lists any address space where its local lock suspension time is over 50%

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

Runtime Diagnostics: **CPU Analysis**

- Takes two quick samples over 1 second interval
- Any task using > 95% of a single CPU is considered a potential problem
- The usage reported might be > 100% if an address space has multiple TCBs and several are using a high percentage of the capacity of a CPU

```
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.
```

Runtime Diagnostics: **Loop Detection**

- Investigates all tasks in all address spaces looking for TCB loops
  - Takes a snapshot of the system trace
  - Looks for consistent, repetitive activity that typically indicates a loop
- When both HIGHCPU and LOOP events occur for the same job, there is a high probability that the task in the job is in a loop.
- Normal, corrective action is to cancel the job.

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

Runtime Diagnostics: **z/OS UNIX Latch Contention**

- New in z/OS 1.13
- If z/OS UNIX latch contention or waiting threads exist for > 5 minutes in z/OS UNIX, a Runtime Diagnostics OMVS event is created.
- Normal action is to issue D OMVS,W,A to get the ASID and job names of the w

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

Runtime Diagnostics: **GRS Latch Contention**

- New in z/OS 1.13
- Obtains latch contention information from GRS
- Omits z/OS UNIX file system latch contention
- Returns the longest waiter for each latch set

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

z/OS 1.13 PFA Integration with Runtime Diagnostics

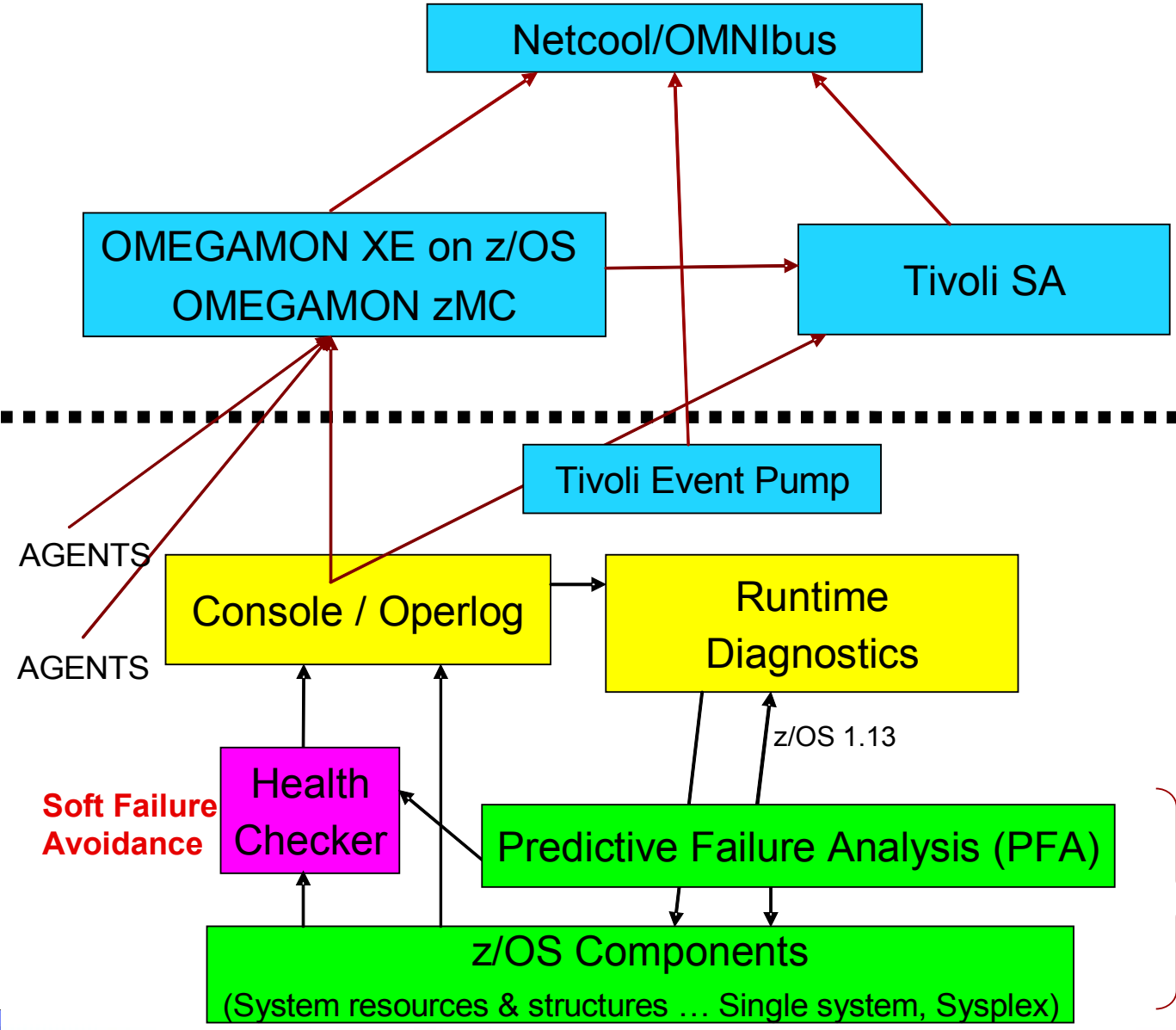
- Detects a *damaged or hung address space or system* based on rates being too low
- When PFA detects an abnormally low condition, Runtime Diagnostics is executed
  - If the results of Runtime Diagnostics indicate a problem,
    - –the PFA exception is issued
    - –the PFA prediction report includes the Runtime Diagnostics output
- **Supported by three checks** → Message Arrival Rate, SMF Arrival Rate, and Enqueue Request Rate
- **Supported by three categories** (if supported by the check) → "Chatty" persistent jobs, other persistent jobs as a group, and total system
- **The Runtime Diagnostics address space (HZR) must be active**

Exception Report for PFA Integration with Runtime Diagnostics

- **"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

```
                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.)
```

Integrated z/OS Soft Failure Detection & Prevention



TIVOLI Management Products

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

Operating System

Soft Failure Problem Determination

Soft Failure Avoidance

Soft Failure Detection

PFA and Runtime Diagnostics Summary

- PFA uses *historical data* and *machine learning algorithms* to detect and report soft failures *before they can impact your business*
- PFA is focused on *damaged systems* and *resource exhaustion*
- Runtime Diagnostics helps you analyze a soft failure, diagnose the problem, and take corrective action in a timely manner
- Automate exception messages
- Using the PFA reports and Runtime Diagnostics to help diagnose problems
- Tuning the PFA checks using the configuration parameters and the EXCLUDED_JOBS list if necessary.
- Using other products to do deep investigation of system or address space problems.

Additional Resources

- One main source of information for both: *z/OS Problem Management* G325-2564-XX
- PFA IEA presentations
  - http://publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp?topic=/com.ibm.iea.zos/zos/1.11/Availability/V1R11_PFA/player.html
  - http://publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp?topic=/com.ibm.iea.zos/zos/1.12/Availability/V1R12_Availability_PFA_Enhancements/player.html
- *z/OS Hot Topics* Newsletters: http://www.ibm.com/systems/z/os/zos/bkserv/hot_topics.html
  - #20 (GA22-7501-16) -- *Fix the Future with Predictive Failure Analysis* by Jim Caffrey, Karla Arndt, and Aspen Payton
  - #23 (GA22-7501-19) – *Predict to prevent: Let PFA change your destiny* by Jim Caffrey, Karla Arndt, and Aspen Payton
  - #23 (GA22-7501-19) – *Runtime to the Rescue! Using Runtime Diagnostics to find out your problems fast* by Bob Abrams, Don Durand, and Dave Zingaretti
- *IBM Systems Magazine - Mainframe Edition*
  - PFA *A Soft Touch* by Karla Arndt, Jim Caffrey, and Aspen Payton
  - http://www.ibmsystemsmagmainframedigital.com/nxtbooks/ibmsystemsmag/mainframe_20101112/index.php#/48