Identify VSAM/RLS Contention (a.k.a, Follow Blockers)

Element/Component: Runtime Diagnostics

Agenda

- Trademarks
- Session Objectives
- Overview
- Usage & Invocation
- Interactions & Dependencies
- Migration & Coexistence Considerations
- Installation
- Session Summary
- Appendix

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.
- Additional Trademarks:
  - None.

Session Objectives

- Understand functions.
  - “Follow blockers”
  - Correlated events
  - Deadlock detection
- Understand new output from Runtime Diagnostics.

Overview

- Problem Statement / Need Addressed / User Stories
  - Identify SMSVSAM contention that could lead to sysplex hangs, waits or slowdowns.
  - If SMSVSAM contention is found, identify the system that is the root of the sysplex contention (not just the holder).
- Solution
  - Using existing GRS latch and enqueue contention analysis, “follow” the blockers to find last problem that can be identified and correlate events gathered into one event.
  - Blockers can be followed to other systems in the sysplex.
  - Detect deadlocks across all data gather (cross-system and between latches and enqueues)
  - Correlate all events that relate to a specific job or task.
- Benefit / Value
  - Identifies the last system/job/task that had a problem detectable by Runtime Diagnostics (i.e., as close to the “root” cause as we know at this time)
  - Valuable to all address spaces that use GRS latches and enqueues, not just VSAMRLS.
  - Deadlocks can be detected between GRS latches and enqueues and across systems in the sysplex (for which data is gathered).
  - Runtime Diagnostic's event data combined in a more meaningful way.

Usage & Invocation

- F hzr,analyze command
  - Same command as in previous releases gives new functions by default

```
MODIFY HZR,ANALYZE,[SYSNAME=SYSNAME]([,DEBUG=ALL)|(LOOP|NOLOOP),(ENQ|
NOENQ),(LOCK|NOLOCK),(HIGHCPU|NOHIGHCPU),(MSGS|NOMSGS),(OMVS|NOOMVS),
(LATCH|NOLATCH),(SERVERHEALTH|NOSERVERHEALTH),(JES2|NOJES2),(DEADLOCK|
NODEADLOCK)])([,SCOPE=FOLLOW|LOCAL)])
```

- DEBUG=DEADLOCK – When DEADLOCK events are found during the ANALYZE request, HZR dumps the HZR address space along with a maximum of 14 address spaces and their data spaces that are associated with the DEADLOCK events.
- DEBUG=NODEADLOCK – When no DEADLOCK events are found during the ANALYZE request, HZR dumps the HZR address space.
- SCOPE=FOLLOW – Follows blockers to find the potential root cause and invokes Runtime Diagnostics on other systems for those ASIDs that are the blockers, if needed. This value is the default.
- SCOPE=LOCAL – Does not invoke Runtime Diagnostics on other systems to follow blockers to other systems. All analysis is performed only on the system specified in SYSNAME (or the HOME system if SYSNAME is not specified). It will follow blockers only if the blockers are on this system or the data for the blockers from other systems is already available on this system (such as enqueue data for SYSTEMS and SYSPLEX enqueues).

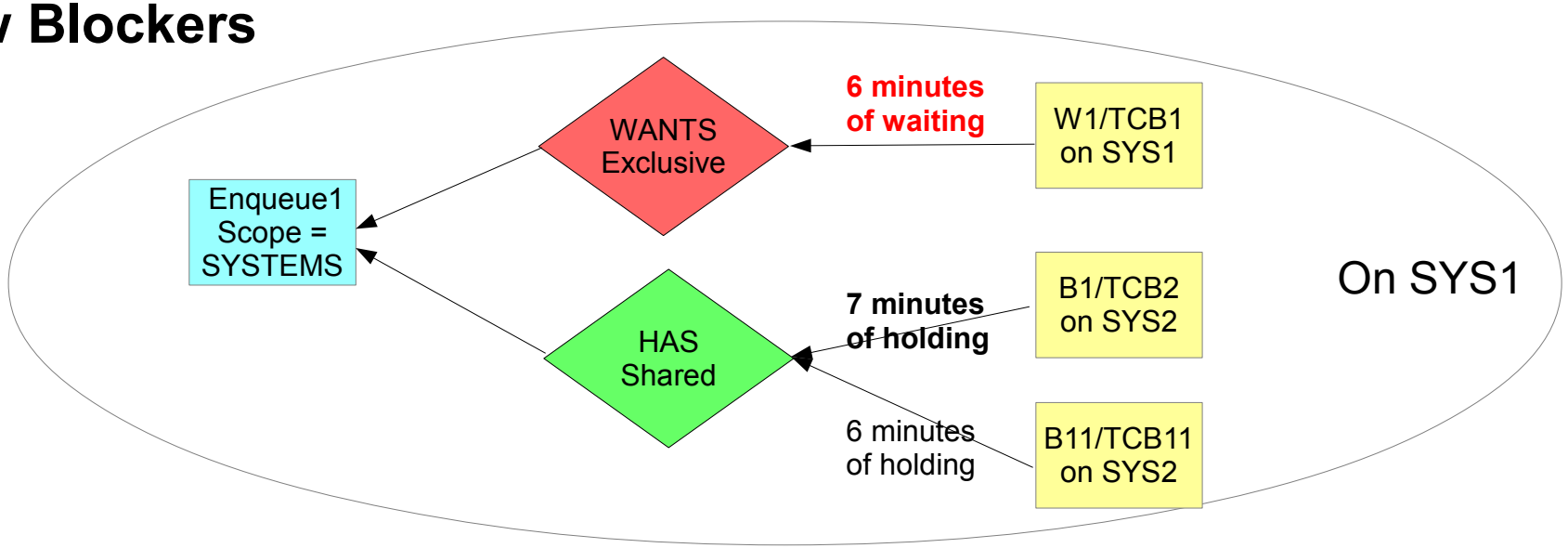
Usage & Invocation

- **Runtime Diagnostics GRS Contention Defined**
  - **Latch contention (R13 and up)**
    - Any address space waiting >= 5 minutes
    - Runtime Diagnostics reports top waiter and top blocker
  - **Enqueue contention**
    - R12 and up: A “system address space” waiting >= 5 seconds
      - Hardcoded list of 32 address spaces
      - Includes SMS, SMSPDSE, SMDPDSE1, SMSVSAM
      - Does NOT include CATALOG (5 seconds is “normal”)
    - Added in V2R2: Any other address waiting >= 5 minutes (including CATALOG)
- Prior to V2R3, latch contention and enqueue contention events are **TOTALLY INDEPENDENT** of each other (even on one system)

Runtime Diagnostics does NOT care if a resource is being HELD for a long time! It only cares if something is WAITING for a resource a long time!

Usage & Invocation

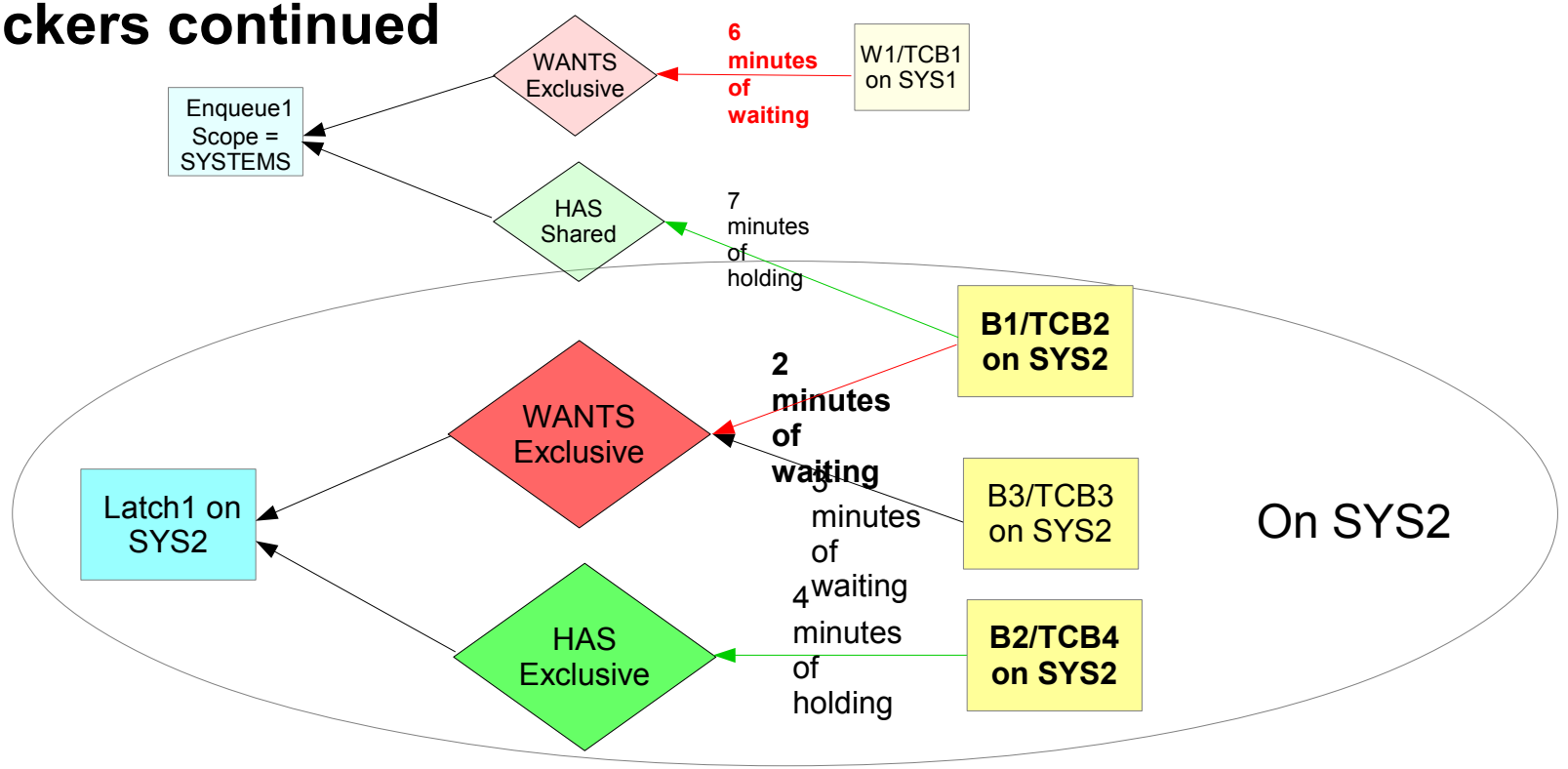
Follow Blockers



1. Runtime Diagnostics finds enqueue contention: **SYS1**/W1/TCB1 waiting >= 5 minutes
2. Follow top blocker **SYS2**/B1/TCB2: See if it is a *waiter* for an enqueue
   - Based on GRS data available on SYS1
   - *Does NOT need to be waiting >= 5 minutes – just needs to be waiting*
3. In this example, Runtime Diagnostics doesn't find SYS2/B1/TCB2 waiting.
4. Runtime Diagnostics now **invokes itself on SYS2** asking if B1 has *anything* wrong with it and if SYS2/B1/TCB2 is *a waiter for a latch or an enqueue* (next slide).

Usage & Invocation

Follow Blockers continued



5. Is there any problem with B1 on SYS2? In this example, Runtime Diagnostics finds none.
6. Is B1/TCB2 on SYS2 a waiter for a latch or an enqueue?
   - Does NOT need to be waiting >= 5 minutes.
   - YES! B2/TCB4 on SYS is the top blocker.
7. **Follow SYS2/B2/TCB4 (on this system):** Is there *anything* wrong with B2 on this system or is B2/TCB4 is a *waiter* for a latch or an enq?
8. Here, we don't find anything wrong with B2 and TCB4 is not a waiter.
9. Runtime Diagnostics on SYS2 returns to SYS1 giving it the following information:
   - B1/TCB2 waiting for a latch held by B2/TCB4
   - Do OPERLOG processing for SYS2 because nothing wrong with B2/TCB4.
10. Runtime Diagnostics on SYS1 looks for OPERLOG messages for SYS1 **AND** SYS2.

Usage & Invocation

Correlated Events

- New, combined events: CORRELATED
- Correlates based on SYS/ASID/JOB (and TCB, if available)
  - For example, HIGHCPU and LOOP events for the same SYS/ASID/JOB will be correlated.
- For contention events, blocker correlates with waiter of other event
- Correlates any number of events as long as they match
- Doesn't correlate JES2 or OPERLOG events.
- Only gives ACTION of “final” event

```
------------------------------------------------------------------------
EVENT 01: HIGH - CORRELATED       SYSTEM: SYS1     2014/11/18 16:09:54
FINAL EVENT FOR EVENT 01:  HIGH: LATCH          SYSTEM: SYS2
 LATCH SET NAME: SYSTEST.LATCH_TESTSET
 LATCH NUMBER: 4         CASID: 0368  CJOBNAME: B2
              ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 WAITER     : 0365  B1        004E2A90  SYS2      00:02:53
 TOP BLOCKER: 0368  B2        004FF048  SYS2
 OTHER WAITERS FOR THIS RESOURCE:
  TOP WAITER: 0245  B3        004EF980  SYS2      00:03:27
ERROR : ADDRESS SPACES MIGHT BE IN LATCH CONTENTION ON SYS2.
ACTION: D GRS,AN,LATCH,DEP,CASID=0368,LAT=(SYSTEST.L*,4),DET
ACTION: TO ANALYZE THE LATCH DEPENDENCIES ON SYS2. USE YOUR
ACTION: SOFTWARE MONITORS TO INVESTIGATE BLOCKING JOBS AND ASIDS.
       ---------------------------------------------------
RELATED EVENT FOR EVENT 01:  HIGH: ENQ            SYSTEM: SYS1
 QNAME: MYQNAME2  SCOPE: SYSPLEX
 RNAME: MYRNAME2
              ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 005A  W1        004E2A70  SYS1      00:06:35
 TOP BLOCKER: 0365  B1        004E2A90  SYS2
ERROR: ADDRESS SPACES MIGHT BE IN ENQ CONTENTION ON SYS1.
```

Usage & Invocation

Correlated Events

- **Summary record**

```
F HZR,ANALYZE
HZR0200I RUNTIME DIAGNOSTICS RESULT 048
SUMMARY: SUCCESS
 REQ: 6 TARGET SYSTEM: SY1       HOME: SY1        2016/01/28 15:21:28
 INTERVAL:   60 MINUTES
 EVENTS FOUND: 2
  PRIORITIES:  HIGH:2  MED:0  LOW:0
  TYPES: ENQ:1 CORRELATED:1
  CORRELATED TYPES: ENQ:3
 PROCESSING BYPASSED FOR SYSTEM SY3:
  ALL TYPES....... FOLLOW BLOCKERS BYPASSED. NO HZR XCF SERVER.
 PROCESSING BYPASSED FOR SYSTEM SY1:
  OPERLOG......... OPERLOG IS NOT ACTIVE.
```

TARGET SYSTEM: The system specified as the SYSNAME on "f hzr,analyze,sysname=SY1
HOME SYSTEM: System where the "f hzr,analyze" was invoked.

EVENTS FOUND, PRIORITIES, TYPES: The total number of events found plus their priorities and types. Types WITHIN correlated events are NOT counted as separate events.

CORRELATED TYPES: has the events found within correlated events.

BYPASS (and FAILURE) lines: One heading per system. Event lines can be duplicated – one per system.
System name is where the bypass or failure occurred. (For example, SY1 tried to invoke SY3, but SY3 failed to execute the request because there was no HZR server defined to XCF. XCF on SY3 sent back a bad RC to Runtime Diagnostics on SY1.)

Usage & Invocation

Correlated Events

```
EVENT 2: HIGH:CORRELATED       SYSTEM: SY1      2016/01/29 16:20:13
FINAL EVENT FOR EVENT 2: HIGH:LATCH                SYSTEM: SY1
 LATCH SET NAME: VAR078#SET1
 LATCH NUMBER: 0          CASID: 0041  CJOBNAME: SVRASID3
               ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 0046  SVRASID2  004F8460  SY1       00:20:14
 TOP BLOCKER: 0041  SVRASID3  004F8460  SY1
ERROR : ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
ACTION: D GRS,AN,LATCH,DEP,CASID=0041,LAT=(VAR078#SE*,0),DET
ACTION: TO ANALYZE THE LATCH DEPENDENCIES. USE YOUR SOFTWARE MONITORS
ACTION: TO INVESTIGATE BLOCKING JOBS AND ASIDS.
        ------------------------------------------------------
RELATED EVENT FOR EVENT 2: HIGH:ENQ                SYSTEM: SY1
 QNAME: SYSZFCT0  SCOPE: SYSPLEX
 RNAME: RES1
               ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 0047  SVRASID1  004F8460  SY1       00:25:20
 TOP BLOCKER: 0045  SVRASID1  004F8460  SY2
ERROR : ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
        ------------------------------------------------------
RELATED EVENT FOR EVENT 2: HIGH:LATCH              SYSTEM: SY2
 LATCH SET NAME: VAR078#SET1
 LATCH NUMBER: 0          CASID: 0041  CJOBNAME: SVRASID2
               ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 0045  SVRASID1  004F8460  SY2       00:25:17
 TOP BLOCKER: 0041  SVRASID2  004F8460  SY2
ERROR : ADDRESS SPACES MIGHT BE IN LATCH CONTENTION.
        ------------------------------------------------------
RELATED EVENT FOR EVENT 2: HIGH:ENQ                SYSTEM: SY2
 QNAME: SYSZFCT0  SCOPE: SYSPLEX
 RNAME: RES2
               ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 0041  SVRASID2  004F8460  SY2       00:25:17
 TOP BLOCKER: 0046  SVRASID2  004F8460  SY1
ERROR : ADDRESS SPACES MIGHT BE IN ENQ CONTENTION.
```

FINAL EVENT is last event we found while following.

- If final is a contention event, the blocker is having some issue Runtime Diagnostics doesn't identify.
- If final is an event with only one job (e.g., loop, lock, serverhealth, etc.), that event "should" be the root problem (at least that Runtime Diagnostics can identify).

To "follow" events:

1) Start with first related event after FINAL.

2) Blocker of that event must be waiter of next related event, etc.

3) Last related event's blocker must be waiter of final (or address space having an issue such as loop, etc.)

System name of the event = SYSNAME

System name of final event and related events is system where the problem was discovered

Related events: Everything else we found while following the problem.

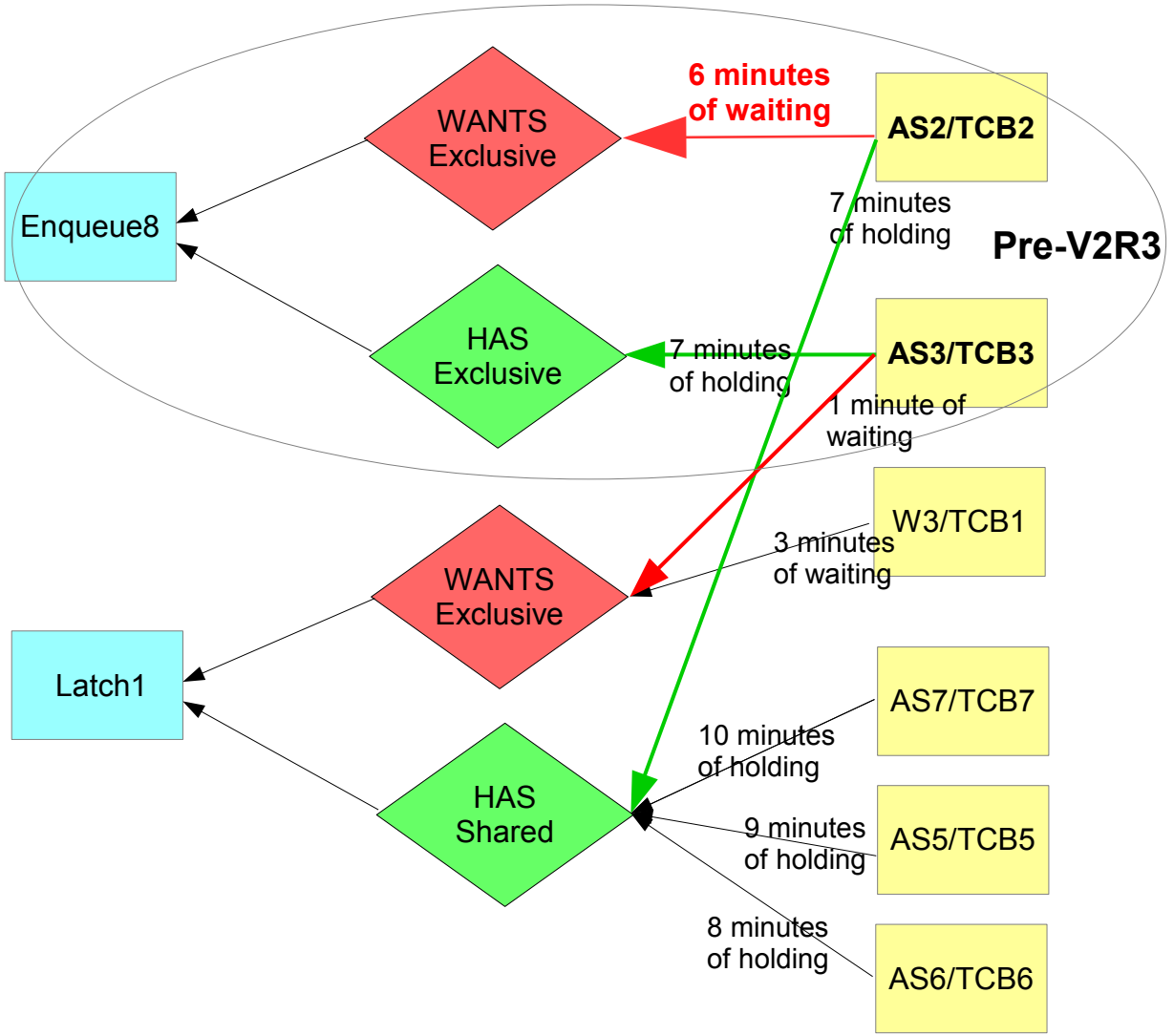
Remember:

1) There must be an event in the list for THIS system where the waiter was on THIS system and it was waiting the required length of time before we start following blockers.

2) We ONLY go to systems in the SYSPLEX where there is a blocker in the chain to follow. We don't "run around" to all systems.

Usage & Invocation

Deadlocks

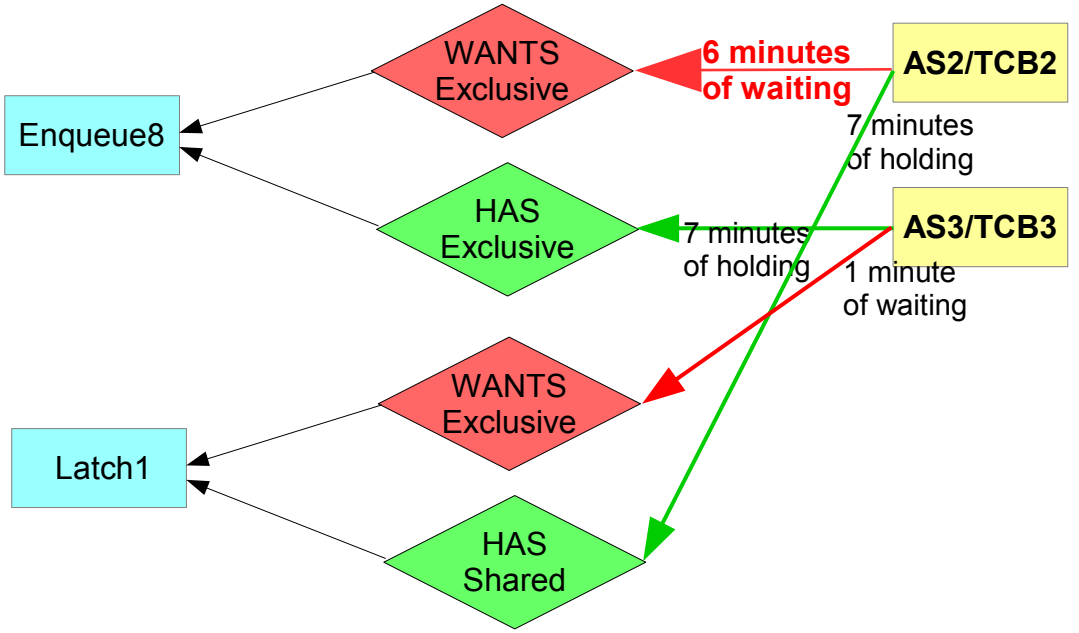


Prior to V2R3:

- *No correlation between enqueues and latches (not even done by GRS)*
- Enqueue event for AS2 waiting for AS3.
- Latch not waiting >= 5 minutes so no latch event
- No Deadlock event!

Usage & Invocation

Deadlocks in V2R3




```
--------------------------------------------------------------------------
EVENT 01: HIGH - DEADLOCK          SYSTEM: SYS1     2014/11/18 16:09:54
RESOURCE: ENQ
 QNAME: ENQUEUE8  SCOPE: SYSPLEX
 RNAME: MYRNAME2
              ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 002C  AS2       004E2A80  SYS1      00:06:35
 TOP BLOCKER: 005A  AS3       004E2A90  SYS1
RESOURCE: LATCH
 LATCH SET NAME: SYSTEST.LATCH_TEST
 LATCH NUMBER: 1          CASID: 005A  CJOBNAME: AS3
              ASID  JOB NAME  TCB/WEB   SYSTEM    WAIT TIME
 TOP WAITER : 005A  AS3       004E2A90  SYS1      00:01:25
 TOP BLOCKER: 002C  AS2       004E2A80  SYS1
ERROR : ADDRESS SPACES WERE DEADLOCKED AT THE TIME OF THE ANALYZE
ERROR : REQUEST.
ACTION: USE YOUR SOFTWARE MONITORS TO INVESTIGATE BLOCKERS TO
ACTION: DETERMINE IF THE DEADLOCK STILL EXISTS AND TAKE APPROPRIATE
ACTION: ACTION TO ELIMINATE THE DEADLOCK.
```

- Can have many resources involved.
- Can be across systems in the sysplex – any system to which we followed blockers.
- Can be a combination of GRS latches and enqueues.

Interactions & Dependencies

- In GRS RING mode, a call to ?ISGECA is needed to get contention times
  - ?ISGECA returns a maximum of 99 enqueues in contention
  - They are not necessarily those with the longest waiters
  - Therefore, in RING mode,
    - Performance may be slower due to the extra call
    - If more than 99 enqueues in contention, not all data will be returned.
    - You may receive Bypass event “Enqueue contention bypassed when no request time.”

Migration & Coexistence Considerations

- Runtime Diagnostics can only invoke itself on other systems that have an HZR server (IXCSRVR) that supports following blockers.
  - If the HZR address space is not active on a system for which HZR tries to invoke itself, a BYPASS event is created in the invoking system's output.
  - If no HZR IXCSRVR exists on a system to which HZR is invoking itself (*such as a pre-V2R3 system*), a BYPASS event is created in the invoking system's output.

Installation

- None
  - Runtime Diagnostics is part of z/OS and in V2R3, starts at IPL by default.
  - All function is available by default.

Session Summary

- Runtime Diagnostics now “follows blockers” of contention events.
  - Will follow to other systems in the sysplex that support the HZR XCF server.
- Runtime Diagnostics correlates events
  - Final event is the last event found in our analysis
  - Related events are the events that led to the final event
- Runtime Diagnostics detects deadlocks
  - Among GRS latches and enqueues
  - Across systems in the sysplex to which Runtime Diagnostics invoked itself to follow blockers

Appendix

- *z/OS V2R3 Problem Management (G325-2564)*