Oracle Performance Diagnostic Guide (Version 3.20)
Hang/Locking

9/11/2012

Welcome to the Oracle Performance Diagnostic Guide   This guide is intended to help you resolve query tuning, hang/locking, and slow database issues. The guide is not an automated tool but rather seeks to show methodologies, techniques, common causes, and solutions to performance problems.

Most of the guide is finished but portions of the content under the Hang/Locking tab is still under development.
Your feedback is very valuable to us - please email your comments to: Vickie.Carbonneau@oracle.com

Contents

Feedback

We look forward to your feedback. Please email any comments, suggestion to help improve this guide, or any issues that you have encountered with the tool usage to Vickie.Carbonneau@oracle.com, Technical Advisor, Center of Expertise (CoE).

Hang/Locking > Identify the Issue > Overview

To properly identify the issue we want to resolve, we must do three things:

- Recognize a query hang or locking issue
- Clarify the details surrounding the issue
- Verify that the issue is indeed the problem.

Recognize a Hang or Locking Issue

What is a Hang or Locking issue?

A "true" database hang issue can manifest itself as:

- A database that is no longer allowing users to connect
- A database that is no longer performing work
- Select 1 from dual does not produce output
- Create table does not complete

A locking issue can manifest itself as:

- One or more sessions that have completely stopped functioning

You might have identified the hang or lock from:

- benchmarking/testing
- user complaints
- systemstate or hanganalyze showing stuck sessions
- a query appearing to hang
- session consuming a large amount of CPU
- ORA-60 errors appearing in the alert log

These problems might appear after:

- schema changes
- changes in database parameters
- changes in application
- database upgrades

Clarify the Issue

A clear problem statement is critical. You need to be clear on exactly what the problem is. It may be that in subsequent phases of working through the issue, the real problem becomes clearer and you have to revisit and re-clarify the issue.

To clarify the issue, you must know as much as possible of the following:

- The affected users
- The sequence of events leading up to the problem
- Where/how it was noticed
- The significance of the problem
- What IS working
- What is the expected or acceptable result
- What have you done to try to resolve the problem

As an example:

- Many sessions do not appear to be completing the requests that were made
- It was noticed by end users.

Notes

- ODM Reference: Identify the Issue

How-To

- How to Identify Resource Intensive SQL for Tuning

Case Studies

- Resolving High CPU usage in Oracle Servers

- New connections cannot be made to the database
- Nothing seems to be working at the moment
- Normally, the user's requests run in less than 2 seconds.
- We checked the alert log for ORA-60 errors

Why is this step mandatory?

Skipping this step will be risky because you might attack the wrong problem and waste significant time and effort. A clear problem statement is critical to begin finding the cause and solution to the problem.

Verify the Issue

Our objective in this step of the diagnostic process is to ensure a database hang or lock is actually the query at the root of the performance problem. At this point, you need to collect data that verifies the existence of a problem.

Notes
• ODM Reference: Identify the Issue

To verify the existence of the issue you must collect evidence of the hang or locking.

Example:

```
SQL> set timing on
SQL> SELECT 1 from dual;
```

If this query does not return results, chances are this is a "true" database hang.

Further examples and advice on what diagnostic information will be needed to resolve the problem will be discussed in the DATA COLLECTION section.

Once the data is collected, you will review it to either verify there is a database hang or locking issue, or decide it is a different issue.

Why is this step mandatory?

If you skip this step, you might have identified the wrong issue and waste significant time and effort before you realize it. For example, the issue may not actually be a database hang or lock. Instead the issue might be due to an OS performance problem In this case, hang/locking will not help solve this problem.

Next Step - Data Collection

When you have done the above, click "NEXT" to get some guidance on collecting data that will help to validate that you are looking at the right problem and that will help in diagnosing the cause of the problem.

Hang/Locking > Identify the Issue >Data Collection

In this step, we will collect data to verify the database hang is due to the database and not external to it. Note: Collect data when the database is hung.

Gather Operating System (OS) Performance Data

OS data is needed to see the overall performance of the machine(s) where Oracle is running.

How-To

- How to use OS commands to diagnose Database Performance issues

Scripts and Tools

- OS Watcher User's Guide
- LTOM - The On-Board Monitor User's Guide

Automatically Using Tools

This section will discuss how to gather data using scripts or tools. Describe when/why you'd use the particular tool, but use the sidebar refs to point to a doc that gives the details on how to setup and use.

10g or higher: If you have obtained a 10gR2 or higher statspack report, you do not need to collect detailed OS data as described below to verify CPU or memory saturation. However, the data captured using the following tools are thorough and may improve the quality of the diagnosis in some cases.

1. ***OS Watcher (OSW) (Preferred Method)***

OS Watcher (OSW) is a collection of UNIX shell scripts intended to collect and archive operating system and network metrics to aid support in diagnosing performance issues. OSW operates as a set of background processes on the server and gathers OS data on a regular basis, invoking such Unix utilities as vmstat, netstat and iostat.

OSW is the preferred way of gathering data on Unix-based systems because it is very simple to install and will collect files that can be analyzed later by Oracle engineers.

Please read the OS Watcher User's Guide for more information on setting up OSW and collecting data. Use OSWg to graph the data for quick analysis.

2. ***LTOM***

The Lite Onboard Monitor (LTOM) is a java program designed as a real-time diagnostic platform for deployment to a customer site. LTOM differs from other support tools, as it is proactive rather than reactive. LTOM provides real-time automatic problem detection and data collection. LTOM is very well suited to transient and unpredictable performance issues..

Please read the LTOM - The On-Board Monitor User's Guide for more information.

3. ***Enterprise Manager***

Enterprise Manager's performance management pages in Grid Control or DB Control include charts that show CPU and memory performance data. To see these charts, click on the host target name (in the "General" section) and then click on the "Performance" page link. From here you will see charts for CPU, memory, and I/O utilization as well as detailed process information.

Enterprise Manager is very good for real-time analysis; however, screen captures will be required for later analysis by Oracle Support.

Manually

OS Watcher is preferred over manual methods. But if you are unable to use OS Watcher and wish to manually collect OS data, please read How to use OS commands to diagnose Database Performance issues for more information.

Note:
Data should be gathered at the same time as the database hang data is gathered

Gather Hanganalyze and SystemState

This section will discuss how to collect diagnostic data using the HANGANALYZE and Systemstate commands.

About Hanganalyze

HANGANALYZE uses internal kernel calls to determine if a session is waiting for a resource, and reports the relationships between blockers and waiters. In addition, it determines which processes are "interesting" to be dumped, and may perform automatic PROCESSSTATE dumps and ERRORSTACKS on those processes, based on the level used while executing the HANGANALYZE.

The "HANGANALYZE" command has been available since Oracle Release 8.1.6. In Oracle9i it was enhanced to provide "cluster wide" information in Real Application Cluster (RAC) environments on a single shot. The meaning of this is that it will generate information for all the sessions in the cluster regardless of the instance that issued the command.

HANGANALYZE syntax:

3 Syntax Examples:
```
ALTER SESSION SET EVENTS 'immediate trace name HANGANALYZE level <level>';
ORADEBUG hanganalyze <level>
ORADEBUG -g def hanganalyze <level>   (Cluster wide syntax)
```

The <level> sets the amount of additional information that will be extracted from the processes found by HANGANALYZE (ERROSTACK dump) based on the "STATE" of the node.

The levels are defined as follows:

10	Dump all processes (IGN state)
5	Level 4 + Dump all processes involved in wait chains (NLEAF state)
4	Level 3 + Dump leaf nodes (blockers) in wait chains (LEAF,LEAF_NW,IGN_DMP state)
3	Level 2 + Dump only processes thought to be in a hang (IN_HANG state)
1-2	Only HANGANALYZE output, no process dump at all

It is advisable not to use levels higher than 3 due to the potentially large number of trace files that may be produced (and could overwhelm the I/O subsystem). Since HANGANALYZE will be mostly used to diagnose "true" hangs, a level 3 will dump the processes that are involved in the hang condition - this usually involves fewer than 4 processes.

NOTE: In some cases hanganalyze could appear to be hung. Hanganalyze will only return when all processes are dumped. To verify that hanganalyze is working on non-clustered environments, monitor the size of the trace files in the user_dump_destination. If the trace files are growing in size, hanganalyze is not hung. To verify that hanganalyze is working on clustered environments, monitor the size of the "diag" trace file in the background_dump_destination.

About Systemstate

A system state is a set of process states for all processes on the instance when the dump is taken. A system state dump is useful in determining the interaction between processes. A systemstate dump will report on resources that are being held by each process.

SYSTEMSTATE syntax:

3 Syntax Examples:
```
ALTER SESSION SET EVENTS 'immediate trace name SYSTEMSTATE level <level>';
ORADEBUG dump systemstate  <level>
ORADEBUG -g all dump systemstate <level>   (Cluster wide syntax)
```

The <level> sets the amount of additional information that will be extracted from the processes found by SYSTEMSTATE dump based on the "STATE" of the node.

The levels are defined as follows:

1	Very basic process information only
2	process + session state objects
10	Most common level - includes state object trees for all processes

Notes
- Steps to generate HANGANALYZE trace files
- Interpreting HANGANALYZE trace files to diagnose hanging and performance problems

Scripts and Tools
- LTOM User Guide
- HANGFG User Guide

Level + 256	Adding 256 to the level will try to dump short stack info for each process. *** IMPORTANT *** Dumping short stacks in the systemstate dump is an enhancement that has been included in patchsets 9.2.0.6 and 10.1.0.4. It is also included in base release 10.2.0.1. Short stacks are produced reasonably quickly on Solaris and Linux but on other platforms including short stacks in dumps can take a very long time. Hence it is advisable to test if the overhead is acceptable before using this option. Typical levels are 266 (Solaris or Linux) and 10 (other platforms)

Automatically Gathering Hanganalyze and Systemstate dumps using Tools

This sections describes two tools which can be used to automatically collect the hanganalyze and systemstate dumps. It is not necessary to use both tools. Choose the tool that works best for your environment.

1. ***LTOM - Transient Hang Conditions***

The Lite Onboard Monitor (LTOM) is a java program designed as a real-time diagnostic platform for deployment to a customer site. LTOM differs from other support tools, as it is proactive rather than reactive. LTOM provides real-time automatic problem detection and data collection. LTOM runs on the customer's UNIX server, is tightly integrated with the host operating system and provides an integrated solution for detecting and collecting trace files for system performance issues. The ability to detect problems and collect data in real-time will hopefully reduce the amount of time it takes to solve problems and reduce customer downtime.

One of the features of LTOM is the Automatic Hang Detection. If you are encountering hangs on a regular basis, this feature of LTOM should be considered.

NOTE: In production environments it is not advisable to leave this feature on as a standard practice. Only use it if you suspect that the database is encountering hang issues. The amount of tracing collected for hang issues could impact the system and should be monitored.

Please read the LTOM - The On-Board Monitor User's Guide for more information. In most cases the default collection is recommended.

The default collection is as follows...

- ***HangAnalyze Level 3***
- ***Systemstate Level 266***
- ***Wait 60 seconds***
- ***HangAnalyze Level 3***
- ***Systemstate Level 266***

For a non-clustered environment verify that new trace files were written to the user_dump_destination. Examine the trace files to ensure that they contain 2 hanganalyze dumps and 2 systemstate dumps.

For a clustered environment verify that the "diag" background process trace file has been updated in the background_dump_destination on each node. Examine each trace file to ensure that it contains 2 hanganalyze dumps and 2 systemstate dumps.

2. ***HANGFG - Simplified Data Collection***

HANGFG (Hang file generator) is a series of unix shell scripts used to automate the generation and collection of hanganalyze and systemstate trace files. HANGFG generates and collects hang trace files based on the impact of taking diagnostic traces on a system which is already in a degraded state. The overall decision on what level of impact the user can afford is left up to the user when he runs HANGFG, as the level of impact is passed in as an argument to the tool. HANGFG is also capable of making this decision for the user if the user selects light impact (option 1) as an argument to the tool. HANGFG is RAC aware and can run in either a RAC or non RAC environment.

Please read the HANGFG: THE HANG FILE GENERATOR USER GUIDE for more information. In most cases the default collection is recommended.

The default collection is as follows:
- ***2 HangAnalyze Level 3***
- ***1 Systemstate Level 10***

For a non-clustered environment verify that new trace files were written to the user_dump_destination. Examine the trace files to ensure that they contain 2 hanganalyze dumps and a systemstate dump.

For a clustered environment verify that the "diag" background process trace file has been updated in the background_dump_destination on each node. Examine each trace file to ensure that it contains 2 hanganalyze dumps and a systemstate dump.

Non-Clustered Environment: Manual Steps for Gathering Hanganalyze and Systemstate dumps

This sections provides detailed steps for manually collecting the hanganalyze and systemstate dumps in a non-cluster environment.

1. ***Database Version 8.1.7 - 9.2.x***

The ideal manner to collect the hanganalyze and systemstate dumps is to collect 2 sets. However, it is best to have the 2 hanganalyzes in one trace file and the 2 systemstate dumps in 2 separate trace files. In order to do this, you will need 3 separate sqlplus sessions. We will identify these as SQL1, SQL2, and SQL3.

A. ***Using SQL*Plus connect as SYSDBA using the following command:***
```
sqlplus " / as sysdba"
```
Do this 3 times in 3 separate windows, creating 3 sqlplus sessions (SQL1, SQL2, and SQL3)

B. ***In SQL1 execute the following:***
```
SQL1> oradebug setmypid
SQL1> oradebug unlimit;
SQL1> oradebug hanganalyze 3
```

C. ***In SQL2 execute the following:***
```
SQL2> oradebug setmypid
SQL2> oradebug unlimit;
SQL2> oradebug dump systemstate 266
```

D. ***Wait at least 2 minutes to give time to identify process state changes.***

E. ***In SQL1 execute the following:***
```
SQL1> oradebug hanganalyze 3
```

F. ***In SQL3 execute the following:***
```
SQL3> oradebug setmypid
SQL3> oradebug unlimit;
SQL3> oradebug dump systemstate 266
```

NOTE:
- ***Systemstate level 258 and 266 are only available in patchsets 9.2.0.6, 10.1.0.4 and base release 10.2.0.1 and higher. If you are not on one of these releases, use level 10 in place of the level suggested above.***
- ***If you are using systemstate level 266 and it is taking much longer than expected to generate the dump file, then end this systemstate dump and try level 258.***

Verify that 3 new trace files were created in the user_dump_destination. Examine the files to ensure that one file contains 2 hanganalyze dumps and each of the other two files contains a systemstate dump.

2. ***Database Version 10.1 and higher***

The ideal manner to collect the hanganalyze and systemstate dumps is to collect 2 sets. However, it is best to have the 2 hanganalyzes in one trace file and the 2 systemstate dumps is 2 separate trace files. In order to do this, you will need 3 separate sqlplus sessions. We will identify these as SQL1, SQL2, and SQL3.

A. ***export ORACLE_SID=PROD ## Replace PROD with the SID you want to trace***

B. ***Using SQL*Plus connect as SYSDBA using the following command:***
```
sqlplus -prelim / as sysdba
```
Do this 3 times in 3 separate windows, creating 3 sqlplus sessions (SQL1, SQL2, and SQL3)

C. ***In SQL1 execute the following:***
```
SQL1> oradebug setmypid
SQL1> oradebug unlimit
SQL1> oradebug hanganalyze 3
```

D. ***In SQL2 execute the following:***
```
SQL2> oradebug setmypid
SQL2> oradebug unlimit
SQL2> oradebug dump systemstate 266
```

E. ***Wait at least 2 minutes to give time to identify process state changes.***

F. ***In SQL1 execute the following:***
```

SQL1> oradebug hanganalyze 3
```

G. ***In SQL3 execute the following:***
```
SQL3> oradebug setmypid
SQL3> oradebug unlimit
SQL3> oradebug dump systemstate 266
```

NOTE:
- ***Systemstate level 258 and 266 are only available in patchsets 9.2.0.6, 10.1.0.4 and base release 10.2.0.1 and higher. If you are not on one of these releases, use level 10 in place of the level suggested above.***
- ***If you are using systemstate level 266 and it is taking much longer than expected to generate the dump file, then end this systemstate dump and try level 258.***

Verify that 3 new trace files were created in the user_dump_destination. Examine the files to ensure that one file contains 2 hanganalyze dumps and each of the other two files contains a systemstate dump.

NOTE: It is advisable not to use levels higher than 3 due to the potentially large number of trace files that may be produced (and could overwhelm the I/O subsystem). Since HANGANALYZE will be mostly used to diagnose "true hangs", a level 3 will dump the processes that are involved in a the hang condition - this usually involves fewer than 4 processes.
NOTE: Ensure that the database has a very large MAX_DUMP_FILE_SIZE or trace files may get truncated. Also ensure that the DUMP_DEST parameters point to where there is plenty of disk space.

Clustered Environment: Manual Steps for Gathering Hanganalyze and Systemstate dumps

This sections provides detailed steps for manually collecting the hanganalyze and systemstate dumps in a cluster environment.

1. ***Database Version 9.0.1 - 9.2.x***

The trace files will be written to the "diag" background process trace file in the background_dump_destination on each node.

A. ***Using SQL*Plus connect as "/ AS SYSDBA" using the following command:***

```
sqlplus "/ as sysdba"
```

B. ***Execute the following:***

```
SQL> oradebug setmypid
SQL> oradebug unlimit
SQL> oradebug -g all hanganalyze 3
SQL> oradebug -g all dump systemstate 266
```

C. ***Wait at least 2 minutes to give time to identify process state changes.***

D. ***Execute the following:***

```
SQL> oradebug -g all hanganalyze 3
SQL> oradebug -g all dump systemstate 266
```

Verify that the the "diag" background process trace file has been updated in the background_dump_destination on each node. Examine each "diag" file to ensure that it contains 2 hanganalyze dumps and a systemstate dump.

2. ***Database Version 10.1 and higher***

The trace files will be written to the "diag" background process trace file in the background_dump_destination on each node.

A. ***Identify the SID you want to trace***

```
export ORACLE_SID=PROD
```

Replace PROD with the SID you want to trace

B. ***Using SQL*Plus connect as "/ AS SYSDBA" using the following command:***

```
sqlplus -prelim / as sysdba
```

C. ***Execute the following:***

```
SQL> oradebug setmypid
SQL> oradebug unlimit
SQL> oradebug -g all hanganalyze 3
SQL> oradebug -g all dump systemstate 266
```

D. ***Wait at least 2 minutes to give time to identify process state changes.***

E. ***Execute the following:***

```
SQL> oradebug -g all hanganalyze 3
SQL> oradebug -g all dump systemstate 266
```

Verify that the the "diag" background process trace file has been updated in the background_dump_destination on each node. Examine each "diag" file to ensure that it contains 2 hanganalyze dumps and a systemstate dump.

NOTE: It is advisable not to use levels higher than 3 due to the potentially large number of trace files that may be produced (and could overwhelm the I/O subsystem). Since HANGANALYZE will be mostly used to diagnose "true hangs", a level 3 will dump the processes that are involved in a the hang condition - this usually involves fewer than 4 processes.
NOTE: Ensure that the database has a very large MAX_DUMP_FILE_SIZE or trace files may get truncated. Also ensure that the DUMP_DEST parameters point to where there is plenty of disk space.

Gather V$ View Data

This section provides a set of queries that should be executed at the time of the database hang.

Notes

- Diagnosing Database Hanging Issues

```
SPOOL v_views.log;

set linesize 130
col "Parameter" form a50
col "Session Value" form a30
col "Instance Value" form a30
select a.ksppinm "Parameter",
b.ksppstvl "Session Value",
c.ksppstvl "Instance Value"
from x$ksppi a, x$ksppcv b, x$ksppsv c
where a.indx = b.indx and a.indx = c.indx
order by 1;

SELECT class, value, name
FROM v$sysstat;

SELECT sid, id1, id2, type, lmode, request
FROM v$lock;

SELECT l.latch#, n.name, h.pid, l.gets, l.misses, l.immediate_gets, l.immediate_misses, l.sleeps
FROM v$latchname n, v$latchholder h, v$latch l
WHERE l.latch# = n.latch#
AND l.addr = h.laddr(+);

SELECT *
FROM v$session_wait
ORDER BY sid;

/* repeat last query 3 times - we want to see who's repeatedly waiting*/

SPOOL OFF;
```

Next Step - Analyze

When you have done the above, click "NEXT" to get some guidance on collecting data that will help to validate that you are looking at the right problem and that will help in diagnosing the cause of the problem.

This step will analyze the data collected in the previous step to verify if the database is hung, has stuck sessions, is slow, or if the problem lies outside of the database.

Verify OS Resource Usage

This step will verify that:

- There are enough CPU and memory resources for Oracle processes, or if not, then at least Oracle is using those resources and more detailed analysis of the database is required
- OR, non-Oracle processes are using most of the CPU or memory; this is not an Oracle hang issue

Check CPU Consumption

A system needs sufficient CPU for good, solid performance. Analyze CPU utilization by answering the following questions:

1. ***Are CPU resources scarce?***

We will check CPU usage by looking for:

- ***Total CPU utilization (USER + SYS) should be less than 90%***
  - ***Batch or reporting applications can use higher CPU utilization to maximize throughput, but generally OLTP must have some headroom to permit a stable response time.***
- ***Run queue size per CPU less than 4***
  - ***The run queue size is a very good indicator of CPU utilization problems. In general, when the run queue per CPU is 4 or higher, the system may experience performance problems (of course, the higher the run queue size the worse the problem). This is an indicator of how many processes must wait for a CPU on average and can be used as a gauge on the scarcity of CPU resources.***

Check CPU usage depending on how you collected OS data:

Enterprise Manager

1. ***In the General section of the database tab, click on the name of your host***
2. ***On the host page, examine the chart for CPU utilization. Select the "CPU Details" view from the pull-down menu. This will show you charts of CPU utilization and CPU load.***

OS Watcher

- ***Graph the data using OSWg and inspect the CPU utilization and run queue charts.***

LTOM

- ***Use the LTOM profiler to generate the CPU charts***

If CPU is not saturated, then proceed to check memory utilization below; otherwise check if Oracle processes are using most of the CPU in the next step.

Notes
- Link 1 title

How-To
- Link 1 title

SR Help
- Link 1 title

Case Studies
- Link 1 title

2. ***What processes are using most of the CPU?***

Examine the data you collected to find out what kind of process is using most of the CPU. The analysis method depends on which type of data you collected - see below:

OSWatcher

1. ***Navigate to the OSW archive directory and then to the directory with "top" reports, oswtop***
2. ***Examine the files there which correspond to the time of the performance problem; look at the top processes using CPU***

   For example:

```
        zzz ***Thu Feb 8 15:03:14 PST 2007
load averages:  3.57,  4.23,  3.32    15:03:15
105 processes: 98 sleeping, 6 running, 1 on cpu

Memory: 2048M real, 29M free, 4316M swap in use, 1847M swap free


   PID USERNAME THR PRI NICE  SIZE   RES STATE    TIME    CPU COMMAND
 19003 oracle     1  32    0    0K    0K run      6:37 25.13% oracle
  6446 oracle     1   0    0    0K    0K run      0:05 21.31% oracle
  1980 oracle    30  29   10  200M   53M sleep    1:23  4.83% java
  6408 oracle     1  59    0   26M   12M sleep    0:01  2.68% perl
 26471 oracle     1  59    0    0K    0K sleep    0:01  1.48% oracle
  6424 oracle     1  59    0    0K    0K sleep    0:00  0.81% oracle
   697 oracle    14  59   -5    0K    0K sleep  340:59  0.67% ocssd.bin
  6455 oracle     1  59    0    0K    0K sleep    0:00  0.56% oracle
 28744 oracle     1  59    0    0K    0K sleep    3:50  0.25% oracle
```

   The top two Oracle processes use around 46% of the CPU. However, because top doesn't give complete information on the process, you'll still need to determine which instance the oracle processes belong to (if more than one on the machine).

3. ***If most of them are Oracle processes, then check which instance they belong to (if you have more than one instance on the machine)***
4. ***Determine if they are all part of the same instance or not; you may have more than once instance that is affecting performance***

If most of the CPU is used by this instance, then you have verified that Oracle is responsible for the CPU consumption.

LTOM Profiler

Under construction

Enterprise Manager

1. ***In the General section of the database tab, click on the name of your host***
2. ***On the host page, click on the Performance tab***
3. ***Review the list of processes in the Process section at the bottom of the page***
4. ***If most of them are Oracle processes, then check which instance they belong to (if you have more than one instance on the machine)***
5. ***Determine if they are all part of the same instance or not; you may have more than once instance that is affecting performance***

If most of the CPU is used by this instance, then you have verified that Oracle is responsible for the CPU consumption.

Are Oracle processes using most of the CPU?

- ***If Oracle processes use most of the CPU, you have identified an Oracle performance problem instead of hang/locking problem. Click on the Slow Database tab for assistance with performance tuning.***
- ***If non-Oracle processes are using most of the CPU, the problem appears to be outside of Oracle; this tuning effort should be aborted and the non-Oracle processes should be investigated or more CPUs should be added to the system.***

Check Memory Consumption

Database performance can be very slow and unpredictable, sometimes resembling a database hang, when the system is short on memory. Answer the two questions below to determine if there is a memory problem on your system.

1. ***Is there a memory shortage?***

Regardless of the tool used to collect the data, you will need to consider the following metrics when looking for a memory shortage:

- ***Memory Utilization (% or free KB): measures how much physical memory has been allocated to processes. When this is around 100% the system will utilize more and more swap; the severity of the shortage is evident by the following two metrics.***

- ***Memory Page Scan Rate (pages/s): a measure of how hard the page scanner is working to reclaim memory. When this is in the hundreds/sec, its likely that there is a memory shortage.***

- ***Swap Utilization (% or free KB): how much of the swap device is being used. As physical memory becomes scarce this percentage goes up. Compare this value to a baseline to see if swap usage increased beyond a normal amount for the system. As swap approaches 100% utilization, the memory crunch gets worse and the system becomes unstable (and could crash).***

You are seeing a memory shortage If you see memory utilization close to 100%, the scan rate in the hundreds / sec, and a large percentage of the swap device utilized.

These metrics can be seen using the following tools or data collection:

Enterprise Manager

1. ***In the General section of the database tab, click on the name of your host***
2. ***On the host page, examine the chart for memory utilization. Select the "Memory Details" view from the pull-down menu. This will show you charts of memory utilization, swap utilization, and page scanner activity.***

OS Watcher

1. ***Use OSWg to create graphs of the OS data, specifically the memory graphs***
2. ***Examine the graphs for "Memory: Available Swap" and "Memory: Scan Rate"***

LTOM Profiler

1. ***Create an LTOM Profiler output directory***
2. ***Click on the link for Operating System Memory***
3. ***Examine the graphs for "Memory: Available Swap" and "Memory: Scan Rate"***

If there is NOT a memory shortage, then proceed to verify if the database is hung. Otherwise, find the processes using memory with the following step.

2. ***What is using most of the memory?***

Determine which processes are using most of the memory (Oracle or non-Oracle) using the following methods/data collection.

Enterprise Manager

1. ***In the General section of the database tab, click on the name of your host***
2. ***On the host page, click on the Performance tab***
3. ***Sort the list of processes in the Process section at the bottom of the page by "Memory Utilization %" (selectable in the "View by" dropdown list)***
4. ***If most of them are Oracle processes, then check which instance they belong to (if you have more than one instance on the machine)***
5. ***Determine if they are all part of the same instance or not; you may have more than once instance that is affecting performance***

OSWatcher

1. ***Go to the OSWatcher "archive/oswps" directory; this directory has output of the "ps" command taken at every sample interval***
2. ***Choose a ps output file during the performance problem***
3. ***Look at one of the ps outputs that was obtained during the problem***

```
        zzz ***Thu Feb 8 15:00:08 PST 2007
 F S      UID   PID  PPID  C PRI NI     ADDR     SZ    WCHAN    STIME TTY      TIME
CMD
 8 R   oracle 21150     1  0  79 20        ?    241          14:43:19 pts/5    0:00 /
usr/bin/ksh ./OSWatcher.sh
 8 S   oracle 14258     1  0  40 20        ?  44958        ?   Jan 25 ?        0:01
ora_q001_DB10gR2
 8 O   oracle  4057  4055  0  69 20        ?    151          15:00:08 pts/5    0:00
ps -elf
 8 S   oracle 28748     1  0  40 20        ?  34961        ? 16:19:40 ?        0:00
ora_d000_DB9iR2
 8 R   oracle 19003 19002 27  77 20        ? 439930          14:41:33 ?        5:49
oracleDB10gR2 (DESCRIPTION=(LOCAL=Y
 8 S   oracle  8842     1  0  40 20        ?   4017        ?   Dec 26 ?        0:41 /
u01/app/oracle/product/DB10gR2/bin
 8 S   oracle  4048  4045  0  59 20        ?    173        ? 15:00:08 pts/5    0:00
iostat -xn 1 3
```

4. ***Sort the processes by the "SZ" column to see which ones are using the most memory (this includes mapped shared memory so care must be taken to subtract the size of the SGA from each value)***
5. ***Determine if they are all part of the same instance or not; you may have more than once instance that is affecting performance***

Are Oracle processes using most of the memory?

- ***If Oracle processes use most of the memory, you have identified an Oracle performance problem instead of hang/locking problem. Click on the Slow Database tab for assistance with performance tuning.***
- ***If non-Oracle processes are using most of the memory, the problem appears to be outside of Oracle; this hang/lock effort should be aborted and the non-Oracle processes should be investigated or more memory should be added to the system.***

Verify the Database is Hung

A database may appear to be hung, but may actually be slow. Therefore database tuning is necessary. Also, there are cases where the database may appear to be hung, but really only a few sessions are hung. This step will help you verify if the database is experiencing a "true" hang condition, a "stuck" or "locking" condition, or if the database is simply slow.

Notes

- Link 1 title

SR Help

- Link 1 title
- Link 2 title

"True" Hang

A "true" hang in the Oracle database can be defined as an internal deadlock or a cyclical dependency between two or more processes. When dealing with DML locks (i.e., TM), Oracle is able to detect this dependency and rollback one of the processes to break the cyclical condition. On the other hand, when this situation happens with internal kernel-level resources (e.g latches or pins), Oracle is usually unable to automatically detect and resolve the deadlock.

1. ***Compare Hanganalyze Outputs***

If you gathered the hanganalyze as previously requested, there should be 2 hanganalyze outputs in one trace file for a non-clustered environment. For a clustered environment the "diag" background process trace file should have been updated in the background_dump_destination on each node with 2 hanganalyze outputs.

In the hanganalyze trace file, the "CYCLE" section reports the process dependencies between sessions that are in a deadlock condition. Cycles are considered "true" hangs. Check for a "CYCLE" section in the hanganalyze trace files previously generated. Below is an example:

```
Cycle 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
   <17/472/0x80057dd8/6674/library cache lock> <--- sid 17, ospid 6674: blocker/
waiter
-- <13/7/0x800580f4/6676/library cache lock>   <--- sid 13, ospid 6676: blocker/
waiter
-- <21/204/0x8005a644/5813/library cache lock> <--- sid 21, ospid 5813: blocker/
waiter
```

If your hanganalyze output has a "CYCLE" section, the database is in a hang state.

- ***Note down the SID and OSPID for the blocker and waiter sessions to use in the Determine a Cause>Data Collection and Determine a Cause>Analysis steps.***
- ***Go to the next step to collect additional data.***

If your hanganalyze output does not have a "CYCLE" section, go to the "Stuck or Locked Sessions" step below.

2. ***Additional Data***

Additional data is automatically collected by the hanganalyze command if a "true" hang is detected. A systemstate and processstate dump will be performed on the blocking session and processstate dumps will be performed on the waiting sessions that are listed in the "CYCLE" section. These dump files are written to the user_dump_destination. The hanganalyze trace file will list the process information collected. It will look similar to the example below. Locate these files for later use.

```
Dumping System_State and Fixed_SGA in process with ospid 6676
Dumping Process information for process with ospid 6676  <--- blocker/waiter
Dumping Process information for process with ospid 6674  <--- blocker/waiter
Dumping Process information for process with ospid 5813  <--- blocker/waiter
```

If your database is hung

- ***Note down the SID and OSPID for the blocker and waiter sessions to use in the Determine a Cause>Data Collection and Determine a Cause>Analysis steps.***
- ***Locate the new trace files listed in hanganalyze for later use***
- ***Continue to "Next Step - Determine a Cause" below***

If your database is NOT hung go to the "Stuck or Locked Sessions" step.

"Stuck" or "Locked" Sessions

Oracle use enqueues as a locking mechanism for managing access to shared resources. A shared resource can be a table definition, a transaction, or any type of structure that represent something sharable between sessions. Each type of action performed by Oracle sessions on those shared resources will require a certain type of lock or lock mode (e.g. a 'select on a table' action will require that the executing session has a shared lock on the resource 'table definition' of the selected table). When conflicting actions are occurring, Oracle will serialize the processing by putting a number of sessions in waiting mode until the work of the blocking session has been completed.

1. ***Compare Hanganalyze Outputs***

If you gathered the hanganalyze as previously requested, there should be 2 hanganalyze outputs in one trace file for a non-clustered environment. For a clustered environment the "diag" background process trace file should have been updated in the background_dump_destination on each node with 2 hanganalyze outputs.

In the hanganalyze trace file, the "OPEN CHAIN" section reports sessions involved in a wait chain. A wait chain means that one session is blocking one or more other sessions. Open chains represents "stuck" or "locked" sessions. Below is an example of an "OPEN CHAIN" section:

Hanganalyze #1

```
Open chains found:
Chain 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
   <16/44773/0x265f15c/1948/SQL*Net message from client> <--sid 16, ospid 1948:
blocker
-- <12/5/0x265fad4/2112/enqueue>                        <--sid 12, ospid 2112:
waiter
-- <13/14/0x265fdfc/2076/enqueue>                       <--sid 13, ospid 2076:
waiter
Chain 2 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
          <19/3/0x2660124/2392/No Wait>
```

Hanganalyze #2

```
Open chains found:
Chain 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
   <16/44773/0x265f15c/1948/SQL*Net message from client> <--sid 16, ospid 1948:
blocker
-- <12/5/0x265fad4/2112/enqueue>                        <--sid 12, ospid 2112:
waiter
-- <13/14/0x265fdfc/2076/enqueue>                       <--sid 13, ospid 2076:
waiter
Chain 2 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
   <19/3/0x2660124/2392/No Wait>
```

Hanganalyze #3

```
Open chains found:
Chain 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
   <16/44773/0x265f15c/1948/SQL*Net message from client> <--sid 16, ospid 1948:
blocker
-- <12/5/0x265fad4/2112/enqueue>                        <--sid 12, ospid 2112:
waiter
-- <13/14/0x265fdfc/2076/enqueue>                       <--sid 13, ospid 2076:
waiter
-- <19/3/0x2660124/2392/enqueue>                        <--sid 19, ospid 2392:
waiter
```

At any given time, there will most likely be blockers and waiters if work is being done on the database. The key is to determine if these blockers and waiters are stuck for an unacceptable amount of time. Compare the chains in the "OPEN CHAIN" sections between the hanganalyze dumps previously generated. If there are a large number of chains in the "OPEN CHAIN" section, focus on the top 3 or 4 longest chains first.

The database is experiencing a locking condition if both of the following are true:

A. ***One or more chains in the "OPEN CHAIN" section have more than one session. At least one blocker and one waiter is needed.***

EXAMPLE: "Chain 1" in all three hanganalyze examples above meet this criteria.

B. ***The blocking session remains the same between hanganalyze outputs in one or more of the chains from "A". A waiter can be added to the chain between outputs and the chain would still be considered to in a wait condition.***

EXAMPLE: "Chain 1" in Hanganalyze #1,Hanganalyze #2, and Hanganalyze #3 meet this criteria. The blocking session of "Chain 1" in Hanganalyze #1 and Hanganalyze #2 does not change. In "Chain 1", sid 16 is the blocker and sid 12, 13, and 19 are waiters. The chain is the same in each hanganalyze output with the exception of sid 19. Sid 19 was added to the chain in Hanganalyze #3.

If your hanganalyze output is showing a locking condition:

- ***Note down the SID and OSPID for the blocker and waiter sessions to use in the Determine a Cause>Data Collection and Determine a Cause>Analysis steps.***
- ***Go to the "Collect Additional Data" step to collect more data for the sessions that are stuck.***

2. ***Collect Additional Data***

In this step you will perform errorstack dumps on some sessions in the "OPEN CHAIN" section. The session's ospid will be used in the errorstack dump syntax below.

```
Open chains found:
Chain 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
   <16/44773/0x265f15c/1948/SQL*Net message from client> <--sid 16, ospid 1948:
blocker
-- <12/5/0x265fad4/2112/enqueue>                         <--sid 12, ospid 2112:
waiter
-- <13/14/0x265fdfc/2076/enqueue>
-- <19/3/0x2660124/2392/enqueue>
```

Before collecting the errorstack, determine which chains to focus on. If there are many chains in the "OPEN CHAIN" section, apply the below steps on 3 or 4 of the longest chains. You will need to dump the errorstack for the blocker session and the first waiter session on each chain you chose to focus on. In the example above ospid 1948 is the blocker and ospid 2112 is the first waiter. So, you will dump the errorstack for 1948 and 2112.

If a process is blocking a lot of other sessions (usually more than 10), a "FOUND" section usually appears in the hanganalyze trace file prior to the "OPEN CHAINS" section. See the example below. If the "FOUND" section appears in your hanganalyze trace file, compare each session between the hanganalyze outputs. If a blocker remains in the "FOUND" section between outputs, collect an errorstack trace for it using the ospid.

```
Found 34 objects waiting for <sid/sess_srno/proc_ptr/ospid/wait_event>
    <131/754/0x9fc1e8/576293/No Wait>
Found 17 objects waiting for <sid/sess_srno/proc_ptr/ospid/wait_event>
    <245/2343/0xa19f48/575938/latch free>
Found 24 objects waiting for <sid/sess_srno/proc_ptr/ospid/wait_event>
    <144/2183/0xa0c9b8/575457/latch free>
```

Collect the Errorstack trace files using the steps appropriate for your database version.

Database Version 8.1.7 - 9.2.x:

A. ***Using SQL*Plus connect as "/ AS SYSDBA" using the following command:***
sqlplus "/ as sysdba"

B. ***Using the ospid execute the following commands:***
- ***Once for the blocking session on each chain in the "OPEN CHAIN" section that you decided to focus on.***
- ***Once for the first waiting session on each chain in the "OPEN CHAIN" section that you decided to focus on.***
- ***Once for each blocker in the "FOUND" section that remains the same between hanganalyze outputs.***

```
SQL> oradebug setospid <ospid>
SQL> oradebug unlimit
SQL> oradebug dump errorstack 3
```

Database Version 10.1:

A. ***Identify the SID you want to trace***

   ```
   export ORACLE_SID=PROD
   ```

   Replace PROD with the SID you want to trace

B. ***Using SQL*Plus connect as "/ AS SYSDBA" using the following command:***

   ```
   sqlplus -prelim / as sysdba
   ```

C. ***Using the ospid execute the following commands***
   - ***Once for the blocking session on each chain in the "OPEN CHAIN" section that you decided to focus on.***
   - ***Once for the first waiting session on each chain in the "OPEN CHAIN" section that you decided to focus on.***
   - ***Once for each blocker in the "FOUND" section that remains the same between hanganalyze outputs.***

   ```
   SQL> oradebug setospid <ospid>
   SQL> oradebug unlimit
   SQL> oradebug dump errorstack 3
   ```

Multiple trace files will be generated in the user_dump_destination. There will be a trace file for each session that an errorstack was executed for.

If the database has "stuck" or "locked" sessions,
- ***Note down the SID and OSPID for the blocker and waiter sessions to use in the Determine a Cause>Data Collection and Determine a Cause>Analysis steps.***
- ***For a non-clustered environment verify that the new trace files were written to the user_dump_destination. Examine the trace files to ensure that they contain an errorstack dump.***
- ***For a clustered environment verify that the "diag" background process trace file has been updated in the background_dump_destination on each node. Examine each "diag" trace file to ensure that it contains an errorstack dump.***
- ***Continue to "Next Step - Determine a Cause" below***

If the database does not have "stuck" or "locked" sessions, go to the "Slow Database" step.

Slow Database

A severe performance problem may easily be mistaken for a hang. This usually happens when contention is so bad that it seems like the database is completely hung.

In the hanganalyze trace file, the "OPEN CHAIN" section reports sessions involved in a wait chain. A wait chain means that one session is blocking one or more other sessions. Open chains represents "stuck" or "locked" sessions. Below is an example of an "OPEN CHAIN" section:

Hanganalyze #1

```
Open chains found:
  Chain 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
     <13/23/0x2660124/1776/No Wait>
  Chain 2 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
     <16/8/0x265f7ac/1888/SQL*Net message from client>
  -- <15/20/0x265fad4/296/enqueue>
  -- <12/5/0x265fdfc/1804/enqueue>
```

Hanganalyze #2

```
Open chains found:
  Chain 1 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
     <18/3839/0x265ee34/2004/SQL*Net message from client>
  -- <19/23/0x265f15c/1876/enqueue>
  Chain 2 : <sid/sess_srno/proc_ptr/ospid/wait_event> :
     <12/5/0x265fdfc/1804/No Wait>
```

At any given time, there will most likely be blockers and waiters if work is being done on the database. The key is to determine if these blockers and waiters are stuck for an unacceptable amount of time. Compare the "OPEN CHAIN" sections in each of the hanganalyze trace files previously generated. In most cases the chains to focus on will be chains that have a blocking session and one or more waiting sessions. A chain that has a blocking session and no waiting session and the wait_event is "No Wait" can be ignored at this step.

The database is slow if there are no chains in the "OPEN CHAIN" section that are stuck. A chain is not stuck if the blocker session changes in the chain between the hanganalyze outputs.

> EXAMPLE: The blocking session changes in "Chain 2" between Hanganalyze #1 and Hanganalyze #2. In Hanganalyze #1, the blocking session is sid 16. In Hanganalyze #2, the blocking session is sid 12.

If your hanganalyze output is showing slow database conditions, click on the Slow Database tab for assistance with database tuning.

Next Step - Determine a Cause

If the analysis above has confirmed the database is hung or has stuck sessions, click "NEXT" to move to the next phase of this process where you will receive guidance to determine a cause for the hung database or stuck sessions.

Before continuing to *Determine a Cause* > *Overview,*, verify the following:

Non-Clustered Environment

- a new trace file was written to the user_dump_destination containing 2 hanganalyze dumps and a process state dump.
- one or more new trace files were written to the user_dump_destination containing an errorstack dump
- v_views.log file was created

Clustered Environment

- the "diag" background process trace file has been updated with 2 hanganalyze dumps and a process state dump.
- the "diag" background process trace file has been updated with one or more errorstack dumps
- v_views.log file was created

Would You Like to Stop and Log a Service Request?

We would encourage you to continue until at least the "Determine a Cause", "Data Collection" step, but If you would like to stop at this point and receive assistance from Oracle Support Services, please do the following:

- In the SR Creation Template, Question "Last Diagnostic Step Completed?", Please copy and paste the following:

  Last Diagnostic Step = Performance_Diagnostic_Guide.Hang/Locking.
  Issue_Identification.Analysis

- Enter the problem statement and how the issue has been verified (if performed)
- Gather the following items already collected and prepare to upload them to the service request
  - Hanganalyze and Errorstack trace files
  - v_views.log file with the v$ data.
  - OS data
- Optionally, gather an RDA
- Gather other relevant information you may have

The more data you collect ahead of time and upload to Oracle, the fewer round trips will be required for this data and the quicker the problem will be resolved.

Click here to log your service request

Hang/Locking > Determine a Cause >Overview

At this point you should have identified if the issue you are experiencing is a "True" database hang, "Stuck" or "Locked" sessions, OS resource problem, or slow database. The "Determine a Cause" portion of the guide will assist you in troubleshooting "True" database hang, or "Stuck" or "Locked" session related issues.

Data Collection

To further troubleshoot "Stuck" sessions, you will have several choices. You can troubleshoot by taking a closer look at the traces previously generated (hanganalyze, systemstate, and errorstack dumps), by using OEM, by using utllockt.sql, or by performing queries on some V$ views. Hanganalyze, systemstate, and errorstack dumps are great for collecting hang related data quickly, but understanding how to read these files can be complex. It is not really possible to present all methods for interpreting these files, but examples will be provided in the Determine a Cause > Analysis portion of the guide.

The hanganalyze, systemstate, and errorstack dumps are very useful (if collected as previously requested) when logging a service request because they generally provide a good picture of what the database is doing when the sessions are hung.

Troubleshooting "Stuck" session issues by collecting data from the V$ views is less complex than reading the dump files. However, in order for this to be effective, the sessions must remain in a "stuck" state until all the data is collected. In some cases this is not possible due to time constraints.

Analysis Approach

Further troubleshooting will:

1. Determine which wait events are involved
2. Determine the active SQL involved in the blocking and waiting sessions
3. Look for common causes for "Stuck" sessions
4. Review possible solutions for the likely cause
5. Implement the best solution
6. Verify that the solution solved the problem or if more work is needed

Click Next to continue.

This phase is very critical to resolving the hang or locking problem because accurate data about the sessions involved is essential for us to determine a cause for the hang or lock.

"True" Database Hang

To further troubleshoot a "True" database hang, you will examine the hanganalyze, systemstate, and errorstack trace files more closely.

No additional data needs to be collected. If you have not already done so, verify that all requested files from the *Identify the Issue > Data Collection and Analysis* steps exist. See *Identify the Issue > Analysis > Next Step - Determine a Cause* for a list of the files. Continue to the *Determine a Cause > Analysis* step.

"Stuck" or "Locked" Sessions

To further troubleshoot "Stuck" sessions, you will have several options. Choose one.

Use previously generated trace files

Troubleshooting "Stuck" sessions can be done by taking a closer look at the data (hanganalyze, systemstate, and errorstack dumps) generated in the *Identify the Issue > Data Collection and Analysis* steps. Hanganalyze, systemstate, and errorstack dumps are great for collecting hang related data quickly, but understanding how to read these files can be complex. It is not really possible to present all methods for interpreting these files, but examples will be provided in the *Determine a Cause > Analysis* portion of the guide.

In order to use this approach, you will need to have collected all the hanganalyze, systemstate, and errorstack dumps as requested in the *Identify the Issue > Data Collection and Analysis* steps. These dumps also must have been collected at the time that the sessions were stuck.

No additional data needs to be collected. Continue to the *Determine a Cause > Analysis* step.

Continue to the *Determine a Cause > Analysis*

Execute Queries on V$ Views

Troubleshooting "Stuck" sessions can be done by using hanganalyze dumps generated in the *Identify the Issue > Data Collection and Analysis* steps in conjunction with data from V$ views. Some of the data gathered from V$ views is based on the blocker and waiter sessions. These blockers and waiter sessions were identified in the *Identify the Issue > Analysis* step. The stuck session must still be stuck when the data from the V$ views is collected.

The queries below will prompt you for a SIDLIST. Use the blocker and waiter SIDs identified in the *Identify the Issue > Analysis* step when prompted. See the queries below for an example of how to enter this information when prompted.

If you plan to continue with this approach do the following:

```
SPOOL v_lock.log;

set linesize 200
column SID format 9999
column EVENT format a30
column P1 format a30
column P2 format a30
column P3 format a30
column WAIT_T format 9999
select SID,EVENT,P1TEXT||' '||P1 P1,P2TEXT||' '||P2 P2,P3TEXT||' '||P3 P3,SECONDS_IN_WAIT
SEC_IN_WAIT
from v$session_wait
where sid in (&SIDLIST);
--example sidlist entry: 100,118,125
```

```
set linesize 80
select s.username, s.sid, s.module, t.sql_text
from v$session s, v$sql t
where s.sql_address =t.address and s.sql_hash_value =t.hash_value
and s.sid in (&SIDLIST);
--example sidlist entry: 100,118,125

set linesize 200
SELECT DECODE(request,0,'Holder: ','Waiter: ')||sid sess,
id1, id2, lmode, request, type
FROM V$LOCK
WHERE (id1, id2, type) IN
(SELECT id1, id2, type FROM V$LOCK WHERE request>0)
ORDER BY id1, request;

SPOOL OFF;
```

Verify that a file was creating in the current directory called v_lock.log. Examine the file to ensure that there is output from each query.

Continue to the *Determine a Cause > Analysis*

Use OEM (10gR2)

Troubleshooting "Stuck" sessions can be done by using the several OEM features. The stuck sessions must still be stuck when viewing the data via OEM. The stuck sessions must remain stuck until troubleshooting is complete, or screen shots should be saved to reference later in the *Determine a Cause > Analysis* step. Choose one of the features below as a means for locating specific information related to "Stuck" sessions.

1. ***Instance Locks***

Use the Instance Locks page to display a list of all sessions currently blocking other sessions. The Instance Locks page displays a table of sessions listed by Username, Sessions Blocked, Session ID, and Session Serial Number.

Use the Instance Locks page to:
- ***View blocking locks, user locks, or all database locks.***
- ***Kill the current session.***
- ***Access the Session Details page.***
- ***Access the SQL Details page.***
- ***View object details.***

Do the following to access the the Instance Locks:
- ***click the "Performance" tab from the home page***
- ***from the "Performance" page, click the "Instance Locks" link under "Additional Monitoring Links"***

Use the "Help" feature in OEM for more details about each page.

The Instance Locks page shows:
- ***SID***
- ***PID***
- ***SQL Hash Value which links to the "SQL Details" page, Lock Type***
- ***Mode Held***
- ***Mode Requested***
- ***Object Type***
- ***Object Owner***
- ***Object Name***

You can click on the SID to navigate to the "Session Detail" page. The "Session Detail" page shows:
- ***OSPID***
- ***Current/Previous SQL Hash Value which links to the "SQL Details" page***
- ***Blocking Session***
- ***Current Wait Event***
- ***P1,P2,P3***
- ***Object owner.name***

Later in the Determine a Cause > Analysis step we will focus on the "Session Detail" and "SQL Details" pages.

Continue to the Determine a Cause > Analysis

2. ***Blocking Sessions***

Use the Blocking Sessions page to display a list of all sessions currently blocking other sessions. The Blocking Sessions page displays a table of sessions listed by Sessions Blocked, Session ID, and Session Serial Number.

Do the following to access the the Instance Locks:

- ***click the "Performance" tab from the home page***
- ***from the "Performance" page, click the "Blocking Sessions" link under "Additional Monitoring Links"***

Use the "Help" feature in OEM for more details.

The "Blocking Sessions" page shows:

- ***SID***
- ***SQL Hash Value which links to the "SQL Details" page***
- ***Wait Event***
- ***P1, P2, P3***

You can click on the SID to navigate to the "Session Detail" page. The "Session Detail" page shows:

- ***OSPID***
- ***Current/Previous SQL Hash Value which links to the "SQL Details" page***
- ***Blocking Session***
- ***Current Wait Event***
- ***P1,P2,P3***
- ***Object owner.name***

Later in the Determine a Cause > Analysis step we will focus on the "Session Detail" and "SQL Details" pages.

Continue to the Determine a Cause > Analysis

Use utllockt.sql Script

The utllockt.sql script displays, in a tree fashion, the sessions in the system that are waiting for locks and the locks that they are waiting for. The location of this script file is operating system dependent. (You must have run the CATBLOCK. SQL script before using UTLLOCKT.SQL.) .

See My Oracle Support Note 166534.1 for more information.

utllockt.sql will display output to the screen. It is a good idea to spool the output to the file "utllockt.out" prior to running utllockt.sql. This produces a file that can be referred to in later steps. If you choose to spool the output, verify that a file is created and examine the file to ensure that the output is similar to the example.

```
        WAITING_SESSION  LOCK_TYPE  MODE_REQUESTED MODE_HELD  LOCK_ID1  LOCK_ID2
        ---------------- ---------- -------------- ---------- --------- --------
blocker-> 100            None
waiter ---> 118          DML        Exclusive      Exclusive  51148     0
waiter ---> 125          DML        Row-X (SX)     Exclusive  51148     0
```

Once you have identified the blocking and waiting sessions using utllockt.sql, run the following queries. The queries below will prompt you for a SIDLIST. Use the blocker and waiter SIDs identified in the *Identify the Issue > Analysis* step when prompted. See the queries below for an example of how to enter this information when prompted.

```
SPOOL v_lock.log;

set linesize 200
column SID format 9999
column EVENT format a30
column P1 format a30
column P2 format a30
column P3 format a30
column WAIT_T format 9999
select SID,EVENT,P1TEXT||' '||P1 P1,P2TEXT||' '||P2 P2,P3TEXT||' '||P3 P3,SECONDS_IN_WAIT SEC_IN_WAIT
from v$session_wait
where sid in (&SIDLIST);
--example sidlist entry: 100,118,125

set linesize 80
select s.username, s.sid, s.module, t.sql_text
from v$session s, v$sql t
where s.sql_address =t.address and s.sql_hash_value =t.hash_value
and s.sid in (&SIDLIST);
--example sidlist entry: 100,118,125

SPOOL OFF;
```

Verify that a file was created in the current directory called v_lock.log. Examine the file to ensure that there is output from each query.

Continue to the *Determine a Cause > Analysis*

Next Step - Analyze

In the following step, you will receive guidance on interpreting the data you collected to determine the cause for the hang or locking problem; click "NEXT" to continue.

Hang/Locking > Determine a Cause >Analysis

The data collected in the previous step will be analyzed in this step to determine a cause for the hang or locking issue.

By this point you should know if you are troubleshooting a "True" Database Hang, or a "Stuck" or "Locked" Session problem. If you do not know the type of issue that you are troubleshooting, please refer back to the *Identify the Issue > Analysis* step.

Identify Blocker and Waiter Data

This section will guide you on finding important pieces of data needed for troubleshooting hang and locking issues. The data that will be identified is:

OSPID	The OS Process ID is needed to kill sessions once troubleshooting is complete.
Active SQL	The Active SQL will be the current SQL for the SQL. This will be used to identify what the blocker or waiter session is doing at the time or the problem
Wait Event	The wait event is the resource or event that the session is waiting for. This will be used to narrow the problem down to a specific type of hang or lock.
CallStack	The call stack is the list of functions that have been called by the session. This will be used when searching My Oracle Support for already identified issues.

You will not use all the data for every issue, but it is good to be familiar with all of it. There are also several methods to locating the data. This section will discuss these methods. Choose one method for troubleshooting your hang or locking issue.

Before using one of these methods the blocker and waiter sessions need to be identified. This should have been done during the *Identify the Issue>Analysis>Verify the Database is Hung* step. Please refer back to the *Identify the Issue>Analysis>Verify the Database is Hung* step if you do not know the session IDs for the blocker and waiters.

Notes
- Link 1 title
- Link 2 title

How-To
- Link 1 title

SR Help
- Link 1 title

Case Studies
- Link 1 title

Identify Blocker and Waiter Data Using Errorstack

Locate the below information for the Blocker and Waiters. Some or all of the information will be used in the "Determine the Type of Lock" step under the "True Database Hang" step or the "Stuck or Locked Sessions" step below. The information needed for further troubleshooting will depend on the wait event.

If you collected an Errorstack for the blocking session and at least 1 waiting sessions in the Identify the Issue > Analysis step then you can follow these steps. The Errorstack dump contains a "call stack" and a "process state" dump. The process state information is what we will be interested in for this step. The process state is broken down into sections called state objects which are identified with "SO:". The SO type is located at the beginning of the second line of the state object under "SO:". You will only need to examine some of these state objects.

1. ***OSPID***

The OSPID can be found in the "process" state object. See the example below. The sections in blue represent the data to look at in regard to identifying the OSPID.

The OSPID will be needed to kill sessions once troubleshooting is complete.

(process)	***shows that the state object is of type process***
ospid: 25765	***shows that the ospid is equal to 25765***

```
  SO: 0x3661bd0c, type: 2, owner: (nil), flag: INIT/-/-/0x00
  (process) Oracle pid=16, calls cur/top: 0x3672bb94/0x3672bb94, flag: (0) -
            int error: 0, call error: 0, sess error: 0, txn error 0
  (post info) last post received: 0 0 0
              last post received-location: No post
              last process to post me: none
              last post sent: 0 0 0
              last post sent-location: No post
              last process posted by me: none
    (latch info) wait_event=0 bits=0
    Process Group: DEFAULT, pseudo proc: 0x3664d528
    O/S info: user: oracle, term: pts/0, ospid: 25765
    OSD pid info: Unix process pid: 25765, image: username@machine.name.com (TNS V1-
V3)
```

2. *Wait Event*

The Wait Event can be found in the "session" state object. Other helpful information found here is the SID and the SQL address. See the example below. The sections in blue represent the data to look at in regards to identifying the SID, SQL address, and Wait Event.

(session)	***shows that the state object is of type process***
sid: 118	***shows that the sid for this session is 118. This is used to ensure that you are looking at the correct session.***
sql: 0x3076f3f8	***shows that the address of the SQL for this session is 3076f3f8. This will be used to find the Active SQL in the next step.***
waiting for 'enq: TM - contention'	***shows that the wait event for this session is enq: TM - contention. This will be used when determining the type of lock associated with the hang or lock situation that you are troubleshooting.***
name|mode=544d0006	***shows the value for P1 if the wait event is an enqueue. P1 is a hexadecimal value that represents the lock type and lock mode (lmode). See My Oracle Support Note 34566.1 for details on obtaining the lock type and lock mode from P1. My Oracle Support Note 29787.1 provides details on the lock types and lock modes.***

```
  SO: 0x366dd1b4, type: 4, owner: 0x3661bd0c, flag: INIT/-/-/0x00
  (session) sid: 118 trans: 0x354e1fb0, creator: 0x3661bd0c, flag: (41) USR/-
BSY/-/-/-/-/-
            DID: 0001-0010-000010E9, short-term DID: 0000-0000-00000000
            txn branch: (nil)
            oct: 26, prv: 0, sql: 0x3076f3f8, psql: 0x32a2c520, user: 54/SCOTT
  O/S info: user: oracle, term: pts/0, ospid: 25764, machine: machine.name.com
            program: sqlplus@machine.name.com (TNS V1-V3)
  application name: SQL*Plus, hash value=3669949024
  waiting for 'enq: TM - contention' blocking sess=0x0x366c7f14 seq=39 wait_time=0
seconds since wait started=11900
              name|mode=544d0006, object #=c7cc, table/partition=0
```

3. *Active SQL*

The Active SQL can be found in the "Library Object Lock" state object. The Active SQL is helpful for identifying the object involved in the hang or lock.

When examining the SQL try to break it down. This will help during the "Determine the Type of Lock" and "Search My Oracle Support" steps under "True Database Hang" and "Stuck or Locked Sessions". Some examples of questions to ask yourself about the active SQL are:

- ***What objects are involved?***
- ***What are the object types?***
- ***Is the SQL DDL or DML?***
- ***Do these objects have any dependencies? If so, what are they?***
- ***Are there triggers involved?***
- ***Is the SQL part of a package or procedure?***
- ***What part of the application is involved?***
- ***etc...***

See the Active SQL example below. The sections in blue represent the data to look at in regards to identifying the Active SQL.

To find the proper "Library Object Lock" state object containing the active SQL, search for the SQL address. The SQL address to search for is located in the "session" state object shown in the previous "Wait Event" step. In the previous example the SQL address is 0x3076f3f8. When performing the search do not include "0x". Use the digits following "0x". In this example you will search for "3076f3f8". Once you find the Library Object Handle with this address, you should see a line beginning with "name=". This will be the active SQL.

LIBRARY OBJECT LOCK	***shows that the state object is of type Library Object Lock***
LIBRARY OBJECT HANDLE: handle=3076f3f8	***shows that the Library Object Handle address is 3076f3f8. This address is the same as the SQL address found in the "session" state object. This is use to verify that this is the state object containing the active SQL.***

name=lock table emp in exclusive mode	show that the active SQL is "lock table emp in exclusive mode".

```
  SO: 0x2f041b48, type: 53, owner: 0x366dd1b4, flag: INIT/-/-/0x00
  LIBRARY OBJECT LOCK: lock=2f041b48 handle=3076f3f8 mode=N
  call pin=(nil) session pin=(nil) hpc=0000 hlc=0000
  htl=0x2f041b94[0x30c3e358,0x324497a4] htb=0x30c3e358 ssga=0x30c3de24
  user=366dd1b4 session=366dd1b4 count=1 flags=[0000] savepoint=0x460964fc
  LIBRARY OBJECT HANDLE: handle=3076f3f8 mtx=0x3076f4ac(1) cdp=1
  name=lock table emp in exclusive mode
  hash=01e8abf8cabca30a0249f91737e32c8f timestamp=03-27-2007 14:39:28
  namespace=CRSR flags=RON/KGHP/TIM/PN0/SML/KST/DBN/MTX/[120100d0]
  kkkk-dddd-llll=0000-0001-0001 lock=N pin=0 latch#=1 hpc=0002 hlc=0002
  lwt=0x3076f454[0x3076f454,0x3076f454] ltm=0x3076f45c[0x3076f45c,0x3076f45c]
  pwt=0x3076f438[0x3076f438,0x3076f438] ptm=0x3076f440[0x3076f440,0x3076f440]
  ref=0x3076f474[0x3076f474,0x3076f474] lnd=0x3076f480[0x3076f480,0x3076f480]
```

4. CallStack

The call stack trace can be found in the errorstack or processstate output. Search for "Call Stack Trace" in the trace file to find the call stack. An example of a call stack is below.

The call stack may be helpful when searching My Oracle Support for previously identified issues. Searching My Oracle Support is discussed under the "True Database Hang" and the "Stuck or Locked Sessions" steps below.

You do not need to use all functions when performing the search. Only enter some of the function names as part of the search criteria. As a general rule, ignore the functions beginning with 'kse' and 'ksd'. Start by using the first 5 functions following the 'kse' and 'ksd' functions. You may need to try several combinations of search criteria before finding any previously identified issues that match the issue you are attempting to troubleshoot.

```
----- Call Stack Trace -----
calling                  call     entry                argument values in hex
location                 type     point                (? means dubious value)
-------------------- -------- -------------------- ----------------------------
ksedst()+27              call     ksedst1()            1 ? 1 ?
ksedmp()+557             call     ksedst()             1 ? 8409404 ? AC007B ? 18 ?
                                                       C0246C4 ? 6 ?
ksdxfdmp()+1382          call     0C969FDE             3 ? 1 ? BBA5021 ? 6F64736E ?
                                                       312B2928 ? 373134 ?
ksdxcb()+1321            call     00000000             BFFFC3B0 ? 11 ? 3 ?
                                                       BFFFC310 ? BFFFC360 ?
sspuser()+102            call     00000000             1 ? 2000 ? 0 ? 0 ? 0 ? 0 ?
00A767A0                 signal   00000000             C ? BFFFC838 ? BFFFC8B8 ?
ntprd()+150              call     sntpread()           CCAAE88 ? CCAB4A0 ? CCAA030 ?
                                                       CCADB9E ? CCAA09C ? 0 ?
nsprecv()+416            call     00000000             CCA9F08 ? CCADB9E ? CCAA09C ?
                                                       0 ? 0 ? A ?
nsrdr()+155              call     nsprecv()            CCA9BF8 ? BFFFD540 ?
                                                       CCA1AC4 ?
nsdo()+1417              call     nsrdr()              CCA9BF8 ? CCA1AC4 ?
nsbrecv()+51             call     nsdo()               CCA19A8 ? 55 ? CCA1AC4 ? 0 ?
                                                       BFFFD7FC ? 0 ? 3 ?
nioqrc()+373             call     nsbrecv()            CCA19A8 ? CCA1AC4 ?
                                                       BFFFD7FC ? 0 ?
__PGOSF86_opikndf2(      call     00000000             CC6B5C4 ? 0 ? BFFFED90 ? 1 ?
)+740                                                  0 ? 0 ?
opitsk()+511             call     00000000             CC6B5C4 ? 2 ? BFFFED90 ? 1 ?
                                                       0 ? 0 ?
opiino()+821             call     opitsk()             0 ? 0 ?
opiodr()+835             call     00000000             3C ? 4 ? BFFFF830 ?
opidrv()+466             call     opiodr()             3C ? 4 ? BFFFF830 ? 0 ?
sou2o()+91               call     opidrv()             3C ? 4 ? BFFFF830 ?
```

```
opimai_real()+117    call     sou2o()              BFFFF814 ? 3C ? 4 ?
                                                   BFFFF830 ?
main()+111           call     opimai_real()        2 ? BFFFF860 ?
__libc_start_main()  call     00000000             2 ? BFFFF924 ? BFFFF930 ?
+211                                               A81C66 ? BBAFF4 ? 0 ?
```

Continue to the "True" Database Hang step or the "Stuck" or "Locked" Session step based on the type of issue you are troubleshooting. You do not need to do both steps.

Identify Blocker and Waiter Data using V$view output

Locate the below information for the Blocker and Waiters. Some or all of the information will be used in the "Determine the Type of Lock" step under the "True Database Hang" step or the "Stuck or Locked Sessions" step below. The information needed for further troubleshooting will depend on the wait event.

1. ***Wait Event***

The wait event can be identified in the output of v$lock.log. Look for the data generated from the v$session_wait query. An example is below.

SID	***shows that the sid. This is used to ensure that you are looking at the correct session.***
EVENT	***shows the wait event for each session. This will be used when determining the type of lock associated with the hang or lock situation that you are troubleshooting.***
P1	***shows additional details for the wait event. If the wait event is an enqueue, you will need to use the data from P1. P1 shows the name| mode value for the lock if the wait event is an enqueue.***

- ***10g: P1 is a decimal value representing the lock mode. Convert this number to hexadecimal to obtain the lock mode. The lock type is already included in the wait event name. My Oracle Support Note 29787.1 provides details on the lock types and lock modes.***
- ***Pre 10g: P1 is a hexadecimal value that represents the lock type and lock mode (lmode). See My Oracle Support Note 34566.1 for details on obtaining the lock type and lock mode from P1. My Oracle Support Note 29787.1 provides details on the lock types and lock modes.***

```
SID EVENT                        P1                    P2               P3                 ...
--- ---------------------------- --------------------- ---------------- ------------------ ...
100 SQL*Net message from client  driver id 1650815232  #bytes 1                          0  ...
118 enq: TM - contention         name|mode 1414332422  object # 51148   table/partition 0  ...
125 enq: TM - contention         name|mode 1414332419  object # 51148   table/partition 0  ...
```

2. ***Active SQL***

The active SQL can be identified in the output of v$lock.log. Look for the data generated from the v$lock query. An example is below.

When examining the SQL try to break it down. This will help during the "Determine the Type of Lock" and "Search My Oracle Support" steps under "True Database Hang" and "Stuck or Locked Sessions". Some examples of questions to ask yourself about the active SQL are:

- ***What objects are involved?***
- ***What are the object types?***
- ***Is the SQL DDL or DML?***
- ***Do these objects have any dependencies? If so, what are they?***
- ***Are there triggers involved?***
- ***Is the SQL part of a package or procedure?***
- ***What part of the application is involved?***
- ***etc...***

```
USERNAME                         SID
------------------------------ -----
MODULE
------------------------------------------------
SQL_TEXT
---------------------------------------------------
SCOTT                            118
SQL*Plus
lock table emp in exclusive mode
SCOTT                            125
SQL*Plus
update emp set sal = sal * 1
```

Continue to the "True" Database Hang step or the "Stuck" or "Locked" Session step based on the type of issue you are troubleshooting. You do not need to do both steps.

Identify Blocker and Waiter Data using OEM (10gR2)

Regardless of the feature used in OEM to view the blocked sessions, the most valuable information for troubleshooting "Stuck" session will be on the "Session Detail" page. Navigate to the "Session Detail" page for the blocker session and each waiter sessions by clicking on the SID from the initial page of one of the three options discussed in the *Determine a Cause > Data Collection* step. A sample section of the general information on the "Session Detail" page is below.

Note: This sample is from version 10gR2. This page may not look the same for other versions of OEM.

Server

Current Status	ACTIVE
Serial Number	55561
DB User Name	SCOTT
OS Process ID	26566
Logged On Since	Mar 27, 2007 2:44:45 PM
Logged On For	1 Hours, 30 Minutes, 53 Seconds
Connection Type	DEDICATED
Type	USER
Resource Consumer Group	Unavailable

Client

OS User Name	oracle
OS Process ID	26565
Host	machine.name.com
Terminal	pts/1
Current Client ID	Unavailable
Current Client Info	Unavailable

Application

Current SQL	aw0dmbzz2aqrh
Current SQL Command	UPDATE
Last Call Elapsed Time	1 Hours, 30 Minutes, 27 Seconds
SQL Trace	DISABLED
Open Cursors	10
Program	sqlplus@machine.name.com (TNS V1-V3)>
Service	SYS$USERS
Current Module	SQL*Plus
Current Action	Unavailable

Contention

Blocking Session ID	100
File	None
Block Number	0
Row Number	0

Wait

Current Wait Event	enq: TM - contention
Current Wait Class	Application
Waiting for	1 Hours, 30 Minutes, 27 Seconds
P1 name|mode	1414332419
P2 object #	51148
P3 table/partition	0
Object	SCOTT.EMP

Advanced

1. ***OSPID***

The OSPID can be found in "Client" section of the "Session Detail" page. See the example below. The sections in blue represent the data to look at in regard to identifying the OSPID.

The OSPID will be needed to kill the blocking sessions from the OS command prompt once troubleshooting is complete. OEM also provides a button for killing a session on the "Session Detail" page.

Client

OS User Name	***oracle***
OS Process ID	***26565***
Host	***machine.name.com***
Terminal	***pts/1***
Current Client ID	***Unavailable***
Current Client Info	***Unavailable***

2. *Wait Event*

The Wait Event can be found in the "Wait" section of the "Session Detail" page. See the example below. The Wait Event is shown in blue.

P1 (also in blue) shows additional details for the wait event. If the wait event is an enqueue, you will need to use the data from P1. P1 shows the name|mode value for the lock if the wait event is an enqueue.

- ***10g:*** ***P1 is a decimal value representing the lock mode. Convert this number to hexadecimal to obtain the lock mode. The lock type is already included in the wait event name. My Oracle Support Note 29787.1 provides details on the lock types and lock modes.***
- ***Pre 10g:*** ***P1 is a hexadecimal value that represents the lock type and lock mode (lmode). See My Oracle Support Note 34566.1 for details on obtaining the lock type and lock mode from P1. My Oracle Support Note 29787.1 provides details on the lock types and lock modes.***

Wait
Current Wait Event enq: TM - contention
Current Wait Class Application
Waiting for 1 Hours, 30 Minutes, 27 Seconds
P1 name|mode 1414332419
P2 object #51148
P3 table/partition 0
Object SCOTT.EMP

3. *Active SQL*

A link to the Active SQL can be found in the "Application" section of the "Session Detail" page. See the example below. In this case the SQL is active which is reflected by "Current SQL". In some cases, the SQL may not be active and may be reflected as "Previous SQL". From the "Session Detail" page click on the SQL Hash Value link. This will direct you to the "SQL Details" page.

Application
Current SQL aw0dmbzz2aqrh
Current SQL Command UPDATE
Last Call Elapsed Time 1 Hours, 30 Minutes, 27 Seconds
SQL Trace DISABLED
Open Cursors 10
Program sqlplus@machine.name.com (TNS V1-V3)>
Service SYS$USERS
*Current Module SQL*Plus*
Current Action Unavailable

The "Text" section on the "SQL Details" page contains the SQL. An example of this section is below.

Text
```
lock table emp in exclusive mode
```

When examining the SQL try to break it down. This will help during the "Determine the Type of Lock" and "Search My Oracle Support" steps under "True Database Hang" and "Stuck or Locked Sessions". Some examples of questions to ask yourself about the active SQL are:

- ***What objects are involved?***
- ***What are the object types?***
- ***Is the SQL DDL or DML?***
- ***Do these objects have any dependencies? If so, what are they?***
- ***Are there triggers involved?***
- ***Is the SQL part of a package or procedure?***
- ***What part of the application is involved?***
- ***etc..***

Continue to the "True Database Hang" step or the "Stuck or Locked Session" step based on the type of issue you are troubleshooting. You do not need to do both steps.

"True" Database Hang

In this section, you will have to opportunity to use the lock type (wait event) to identify why your database is hung or you can search My Oracle Support for issues that have already been identified. Start with the "Determine the Type of Lock" step. If this does not lead you to a solution for the hung database, try the "Search My Oracle Support" step.

Notes
- Link 1 title
- Link 2 title

How-To
- Link 1 title

SR Help
- Link 1 title

Case Studies
- Link 1 title

Determine the Type of Lock

Below are some of the common wait events associated with database hangs. If the wait event identified for the blocker or waiter in the above step is the same as one in this list, refer to it for more troubleshooting. Start by focusing on the wait event of the waiters. Once you know what the sessions are waiting on, you can attempt to determine why the blocker is preventing the waiters from doing their work.

1. ***library cache pin***

Library cache pins are used to manage library cache concurrency. Pinning an object causes the heaps to be loaded into memory (if not already loaded). PINS can be acquired in NULL, SHARE or EXCLUSIVE modes and can be considered like a special form of lock. A wait for a "library cache pin" implies some other session holds that PIN in an incompatible mode.

For details on this wait event see My Oracle Support Note 34579.1.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving library cache pin. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

2. *library cache lock*

The library cache lock controls the concurrency between clients of the library cache by acquiring a lock on the object handle so that one client can prevent other clients from accessing the same object or the client can maintain a dependency for a long time (no other client can change the object). This lock is also gotten to locate an object in the library cache.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving library cache lock. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.

Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

3. *row cache lock*

This event is used to wait for a lock on a data dictionary cache specified by "cache id". If one is running in shared mode (Parallel Server), the LCK0 is signalled to get the row cache lock for the foreground waiting on this event. The LCK0 process will get the lock asynchronously. In exclusive mode the foreground process will try to get the lock.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving row cache lock. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.

Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M Effort Details

Work in Progress

L Risk Details

Low risk. Work in Progress.
Solution Implementation

Work in Progress

[TBD]

[TBD]

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M Effort Details

Work in Progress

L Risk Details

Low risk. Work in Progress.
Solution Implementation

Work in Progress

[TBD]

[TBD]

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Search My Oracle Support

If the "Determine the Lock Type" step above did not provide a solution to the database hang, then the next approach to finding a solution to a "True" database hang is to search My Oracle Support for issues that have already been documented by support. If you are unable to identify the issue after performing a search via My Oracle Support, log an SR and provide all data collected up to this point. See the "Open a Service Request with Oracle Support Services" section below for details on logging an SR.

To search My Oracle Support click the "Advanced" search link at the top right corner of any My Oracle Support page. The "Advanced Search" page contains several fields.

Step 1: Specify the sources to search. At a minimum choose "Bug Database", "Knowledge Base", and "Communities" from the list in the "In the source" field.
Step 2: Specify your query terms (at least one term is required). Enter the search criteria in the field "All these words".   Use the default settings for the other fields for the first few search attempts. At some point, you may find that it is better refine the search by filling in the other fields and it may be necessary to limit the key words used in the "All these words" field.

Start the search by listing the call stack and lock type as your query terms.

Example search criteria:
ntprd nsprecv nsrdr nsdo nsbrecv enq: TM connection
where the call stack is in blue and the lock type is in red

This search criteria actually returns no results, but it is a good start. For the next search, remove some of the keywords. It may take several attempts before getting results from the search. You may have to remove the call stack completely from the search criteria. However, it is beneficial to use it if possible because it will help narrow the search results. You may also choose to add keywords to the search to limit the search results. Below are some other suggestions for keywords.

Other keywords to consider for your search criteria:

- the action being performed on the object (update, insert, etc.)
- list the type of object (trigger, table, primary key, foreign key, etc.)
- if the object is a data dictionary object, list its name
- etc.

In most cases the database will need to be rebooted in order to recover from a "true" database hang. If you have identified the cause of the hang from the "Determine the Type of Lock" or "Search My Oracle Support" steps, reboot the database. If you have not identified the cause of the hang and are planning to open an SR it would be best to leave the database in the hung state until the engineer has had an opportunity to review the collected data. If you cannot leave the database in a hung state, verify that you have the data from the previous steps and then reboot the database.

If you need further assistance with your hung database, continue to the "Open a Service Request with Oracle Support Services " step. If you do not need any further assistance, continue to the "Give Us Your Feedback" step.

"Stuck" or "Locked" Sessions

In this section, you will have to opportunity to use the lock type (wait event) to identify why your sessions are stuck or you can search My Oracle Support for issues that have already been identified. Start with the "Determine the Type of Lock" step. If this does not lead you to a solution for the stuck sessions, try the "Search My Oracle Support" step.

Determine the Type of Lock

Below are some of the common wait events associated with stuck sessions. If the wait event identified force blocker or waiter in the above step is the same as one in this list, refer to it for more troubleshooting. Start by focusing on the wait event of the waiters. Once you know what the sessions are waiting on, you can attempt to determine why the blocker is preventing the waiters from doing their work.

1. ***enqueue***

Enqueues are local locks that serialize access to various resources. This wait event indicates a wait for a lock that is held by another session (or sessions) in an incompatible mode to the requested mode.

For details on this wait event see My Oracle Support Note 34566.1.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving enqueues. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Notes
- Link 1 title
- Link 2 title

How-To
- Link 1 title

SR Help
- Link 1 title

Case Studies
- Link 1 title

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

[*TBD*]

[*TBD*]

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

[*TBD*]

[*TBD*]

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

2. library cache lock

The library cache lock controls the concurrency between clients of the library cache by acquiring a lock on the object handle so that one client can prevent other clients from accessing the same object or the client can maintain a dependency for a long time (no other client can change the object). This lock is also gotten to locate an object in the library cache.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving library cache lock. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.

Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.

Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

3. library cache load lock

This event is used when a database object is loaded. The load lock is always gotten in Exclusive mode, so that no other process can load the same object. If the load lock is busy the session will wait on this event until the lock becomes available.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving library cache load lock. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

4. library cache pin

Library cache pins are used to manage library cache concurrency. Pinning an object causes the heaps to be loaded into memory (if not already loaded). PINS can be acquired in NULL, SHARE or EXCLUSIVE modes and can be considered like a special form of lock. A wait for a "library cache pin" implies some other session holds that PIN in an incompatible mode.

For details on this wait event see My Oracle Support Note 34579.1.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving library cache pin. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

5. row cache lock

This event is used to wait for a lock on a data dictionary cache specified by "cache id". If one is running in shared mode (Parallel Server), the LCK0 is signaled to get the row cache lock for the foreground waiting on this event. The LCK0 process will get the lock asynchronously. In exclusive mode the foreground process will try to get the lock.

The observation table below is still under construction. The completed table will include possible causes and solutions to issues involving row cache lock. For now, continue to the next step "Search My Oracle Support".

Note: This list shows some common observations and causes but is not a complete list. If you do not find a possible cause in this list, you can always open a service request with Oracle to investigate other possible causes. Please see the section below called, "Open a Service Request with Oracle Support Services".

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M *Effort Details*

Work in Progress

L *Risk Details*

Low risk. Work in Progress.

Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

  TBD

  TBD

Implementation Verification

NA

TBD

To Be Determined

What to look for

We are working to develop this section

Cause Identified: TBD

To Be Determined

Cause Justification
We are working to develop this area.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

  TBD

  TBD

Implementation Verification

NA

Cause Identified: Many concurrent inserts into the same table

Several sessions are inserting rows into the same table.

Cause Justification
HW enqueue waits are confined to sessions performing inserts into the same tables.

Solution Identified: TBD

To Be Determined

M ***Effort Details***

Work in Progress

L ***Risk Details***

Low risk. Work in Progress.
Solution Implementation

Work in Progress

TBD

TBD

Implementation Verification

NA

Search My Oracle Support

If the "Determine the Lock Type" step above did not provide a solution to the "stuck" sessions, then the next approach to finding a solution to a "True" database hang is to search My Oracle Support for issues that have already been documented by support. If you are unable to identify the issue after performing a search via My Oracle Support, log an SR and provide all data collected up to this point. See the "Open a Service Request with Oracle Support Services" section below for details on logging an SR.

To search My Oracle Support click the "Advanced" search link at the top right corner of any My Oracle Support page. The "Advanced Search" page contains several fields.

Step 1: Specify the sources to search. At a minimum choose "Bug Database", "Knowledge Base", and "Communities" from the list in the "In the source" field.
Step 2: Specify your query terms (at least one term is required). Enter the search criteria in the field "All these words". Use the default settings for the other fields for the first few search attempts. At some point, you may find that it is better refine the search by filling in the other fields and it may be necessary to limit the key words used in the "All these words" field.

Start the search by listing the call stack and lock type as your query terms.

Example search criteria:
ntprd nsprecv nsrdr nsdo nsbrecv enq: TM connection
where the call stack is in blue and the lock type is in red

This search criteria actually returns no results, but it is a good start. For the next search, remove some of the keywords. It may take several attempts before getting results from the search. You may have to remove the call stack completely from the search criteria. However, it is beneficial to use it if possible because it will help narrow the search results. You may also choose to add keywords to the search to limit the search results. Below are some other suggestions for keywords.

Other keywords to consider for your search criteria:

- the action being performed on the object (update, insert, etc.)
- list the type of object (trigger, table, primary key, foreign key, etc.)
- if the object is a data dictionary object, list its name
- etc.

In most cases the database will not need to be rebooted in order to recover from a "stuck" session. Instead, the blocking session can be killed from the OS and the waiting sessions will resume processing. If you have identified the cause of the "stuck" session from the "Determine the Type of Lock" or "Search My Oracle Support" steps, kill the blocking session using the OSPID identified in earlier steps. If you have not identified the cause of the "stuck" session and are planning to open an SR it would be best to leave the sessions in the "stuck" state until the engineer has had an opportunity to review the collected data. If you cannot leave the sessions in a "stuck" state, verify that you have the data from the previous steps and then kill the blocking session.

If you need further assistance with your the "stuck" sessions issue, continue to the "Open a Service Request with Oracle Support Services " step. If you do not need any further assistance, continue to the "Give Us Your Feedback" step.

Open a Service Request with Oracle Support Services

If you would like to stop at this point and receive assistance from Oracle Support Services, please do the following:

- In the SR Creation Template, Question "Last Diagnostic Step Completed?", Please copy and paste the following:

  Last Diagnostic Step = Performance_Diagnostic_Guide.Hang/Locking.
  Issue_Identification.Analysis

- Enter the problem statement and how the issue has been verified
- Upload into the SR:
  - Hanganalyze, Systemstate, and Errorstack trace files
  - v_views.log and v_lock.log files with the v$ data.
  - OS data
  - (optionally) RDA collection

The more data you collect ahead of time and upload to Oracle, the fewer round trips will be required for this data and the quicker the problem will be resolved.

Click here to log your service request

Give Us Your Feedback

Your feedback is very valuable to us - please take the Performance Tuning Guide survey or email your comments to: Vickie.Carbonneau@oracle.com