|  |  |  |
| --- | --- | --- |
| **OS Watcher For Windows (OSWFW) User Guide (文档 ID 433472.1)** | [ת���ײ�](https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=76572835234685&parent=SrDetailText&sourceId=3-13290188391&id=433472.1&_afrWindowMode=0&_adf.ctrl-state=g2ehgj2ux_228%20\o%20×ªµ½µ×²¿)  [转到底部](https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=76572835234685&parent=SrDetailText&sourceId=3-13290188391&id=433472.1&_afrWindowMode=0&_adf.ctrl-state=g2ehgj2ux_228%20\o%20×ªµ½µ×²¿) |  |

graphic



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Applies to:**  Oracle Database Products > Oracle Database Suite > Platform specific utilities  Windows  Microsoft Windows (32-bit) - OS Version: 7  Microsoft Windows x64 (64-bit) - OS Version: 7  Microsoft Windows x64 (64-bit)  Microsoft Windows (32-bit)  Generic Windows  Microsoft Windows x64 (64-bit) - Version: 2008 R2  **Abstract**  **Note:** This tool can still be downloaded and used, but no further enhancement or problem fixes will be provided. If having issues with this tool, please use the tool Cluster Health Monitor instead.  Please see Document ID: 736752.1 for more information, and how to download, the Cluster Health Monitor.  OS Watcher for Windows (OSWFW) is several batch files that run the Windows utility logman and schtasks. The logman utility collects various Operating System counters. It archives these metrics to aid diagnosing performance and Operating System issues. OSWFW has segmented these counter collections into various categories. The schtasks utility is used to run a batch file that cleans up the archive files to keep only 24 hours of data. It is also used if Oracle's Real Application Cluster (RAC) is involved, to run a batch file that checks the RAC Interconnect. OSWFW can be downloaded from this note. Installation instructions for OSWFW are provided in this User Guide.  **History**  **Author :** Kevin Reardon Create Date 05-23-2007 Update Date 05-13-2013 Expire Date  Version: OSWFW 2.5.1  **Details**  The OS Watcher For Windows (OSWFW) User Guide  Kevin Reardon, Center of Expertise  **Introduction**  OS Watcher for Windows (OSWFW) is several batch files that run the Windows utility logman and schtasks. The logman utility collects various Operating System counters. It archives these metrics to aid diagnosing performance and Operating System issues. OSWFW has segmented these counter collections into various categories. The schtasks utility is used to run a batch file that cleans up the archive files to keep only 24 hours of data. It is also used if Oracle's Real Application Cluster (RAC) is involved, to run a batch file that checks the RAC Interconnect. OSWFW can be downloaded from this note. Installation instructions for OSWFW are provided in this User Guide.  **Overview**  OSWFW consists of a batch file and a series logman configuration files that contain the counter paths to be captured. The main controlling batch file is the file "OSWATCHER.BAT," which creates and schedules individual counter collections to collect specific kinds of data, using the Windows logman utility. Each counter collection will have its own output file. This version of OSWFW has been made aware of Oracle's Real Application Clusters. When it runs, it will detect if Oracle Clusterware is installed, install itself on all nodes in the Cluster, and schedule a batch file that checks the RAC Interconnect.  Data collection intervals are configurable by the user, and all counter collections run on this interval. For example, if OSWFW is configured to collect data once per minute, each counter collection will collect its data, append it to its output file, sleep for one minute and repeat the data collection. Each output file will contain, at most, one hour of data. At the end of each hour, logman creates a new file. This file creation interval is not command line modifiable.  The Operation System utility schtasks is used to remove older data collection files. This is done to prevent these collection files to fill up the disk system they reside on. OSWFW will keep twenty four hours of data on disk and will delete the older files. If these files need to be saved, view the help files on schtasks to set up a different task to archive these files.  OSWFW will terminate after the Run Time Interval. With the proper command line options, OSWFW can be stopped on all nodes, or on individual nodes.  **Supported Platforms**  OSWFW is certified to run on the following platforms:   * Windows XP (x86 & x64) * Windows 7 (x86 & x64) * Windows 8 (x86 & x64) * Windows 2003 R1 & R2 (x86 and x64) * Windows 2008 R1 & R2 (x86 & x64) * Windows 2012   OSWFW needs to be run in an Administrator account (Run as Administrator). Exactly which OS permissions are required to run the logman or schtasks is not covered in this document. See the proper Microsoft documentation on this topic. OSWFW was tested in a default installation of the Operating System (kept to the most current patch set available during the testing period) which has all permissions set to their default settings.  OSWFW can not run in OS installations that use a language other than English.  Various commands used in the batch file do not return results in English reliably enough.  **Installing and Removing OSWFW**  ***Installing OSWFW***  OSWFW should be installed manually by using the following procedure.  OSWFW is available through My Oracle Support and is downloaded as a zip file. The user then copies the file oswfw.zip to the directory where OSWFW is to be installed and issues the following command:   |  | | --- | | C:\> unzip oswfw.zip |   This installs all the files associated with OSWFW into this directory. OSWFW is now installed.  Note: The logman utility will not write its counter output to a file on a shared drive. As such it must be installed on a local drive. This is a restriction of the logman utility and not of OSWFW.  **Real Application Cluster and OSWPrivNet.bat**  OSWatcher runs in a Real Application Cluster environment and will deploy itself on all nodes that are a cluster member and are up.  Prior to running OSW the first time, one should rename the file OSWPrivNet.config.template to OSWPrivNet.config and modify it to contain the IP addresses of all the Interconnect IP addresses.  These addresses are the initial IP addresses of the interfaces and not the HAIP addresses.  The HAIP addresses can change between system reboots.  An example of the OSWPrivNet.config file is as follows:  # Start of OSWPrivNet.config file  # Put the IP addresses for all Interconnect interfaces of all nodes on a single line  # Remove the "#" character from the address line.  The following are examples only:  192.168.2.1  192.168.2.2  192.168.2.3  192.168.2.4  # End of OSWPrivNet.config file  In this case each node in the cluster has two interfaces for a total of four IP addresses.  The OSWPrivNet.bat file runs as a scheduled task.  Its purpose is to check the viability of the interconnect network.  It does this by both pinging and running tracert (traceroute).  ***Removing OSWFW***  Removing OSWFW is quite simple.   |  | | --- | | C:\> oswatcher remove |   This will first stop and then remove all the OSWFW counters and tasks from a single node, or all RAC nodes.  To complete the removal task, on the host OSWFW was installed on, and on each node in the RAC cluster it was installed on, issue the following command   |  | | --- | | C:\> del /s osw |   This last step must be manual in order to prevent accidental deletion of the captured data.  **OSWFW Command Line Options**   OSWFW has had a few more command line options added in order to work in a RAC environment. These are detailed in the following section.  **Initially configure OSWFW**  To initially configure OSWFW, you specify the interval where logman will collect the counter data, the number of hours OSWFW will run, and if it is to be run on RAC or not. The following is the syntax to configure OSWFW:   |  | | --- | | OSWatcher {ARG1} {ARG2} {ARG3}  ARG1 = Snapshot interval in seconds  ARG2 = Runtime Interval - hours OSWatcher will run  ARG3 = RAC - detect Real Application Cluster |   When OSWFW is started for the first time it creates the Archive sub-directory and several sub-directories (one for each data collection). OSWFW will automatically start after this command is given.  OSWFW can be reconfigured at any time, running or not, using the same syntax above.  ***OSWatcher start***  OSWFW will start after the first time the command is issued. It can also be stopped from the command line. To start the OSWFW utility execute the OSWATCHER.BAT batch script from the directory where OSWFW was installed. If not run from this directory, OSWFW will not find its configuration files. If it is installed on RAC, this command starts OSWFW on all nodes or an individual node.  The start command line syntax is:   |  | | --- | | OSWatcher start {node name} |   If the node name is left off, and OSWFW was installed on a RAC system, it will start all the counters on all the nodes. It does not matter if they have already been started as no change occurs to an already started counter.  OSWFW is configured to create a new log file every hour and this interval is not configurable (there should be no need to configure it). If no arguments are entered, the script runs with default values of collecting data every 30 seconds and will run for 48 hours.  OSWFW is configured to create a new log file every hour and this interval is not configurable (there should be no need to configure it). If no arguments are entered, the script runs with default values of collecting data every 30 seconds and will run for 48 hours.   |  | | --- | | C:\> oswatcher 60 10 RAC |   This would start the tool, collect data at 60-second intervals, and run for 10 hours. With the last argument, OSWFW will detect it is on RAC, configure all the nodes, and start on all nodes.  ***OSWatcher stop***  OSWatcher stop {node name}  To stop the OSWFW utility execute the OSWatcher stop command from the directory where OSWFW was installed. This will stop all the counters. If OSWFW is installed on a RAC system, an optional node name can be given to stop OSWFW on that node. To stop OSWFW on all nodes, no node name is given.   |  | | --- | | C:\> oswatcher stop |   This will stop OSWFW on the system it is installed on, or all nodes in a RAC system.   |  | | --- | | C:\> oswatcher stop curiousgeorge1 |   This will stop OSWFW on the RAC node named curiousgeorge1.  ***Getting the Status of OSWFW***  To find out the status of all of the counters, use the command line option of "status". If installed on a RAC system, the status of a specific node can be found. The status command line option is used to provide a quick check of the status. If more detail is needed, use the query command line option.   |  | | --- | | OSWatcher status {node name} |   It will list all the counters and show if they are running or not:   |  | | --- | | Start of Operating System Watcher for Windows  The status on curiourgeorge1 is:   Data Collector Set Type Status  -------------------------------------------------------------------------------  OSWCache Counter Running  OSWLogicalDisk Counter Running  OSWMemory Counter Running  OSWNetstat Counter Running  OSWPagingFile Counter Running  OSWPhysicalDisk Counter Running  OSWProcess Counter Running  OSWProcessor Counter Running  OSWServer\_Work\_Queue Counter Running  OSWSystem Counter Running  OSWThread Counter Running  The command completed successfully.   The status of OSWCleanup on curiourgeorge1 is:  Folder: \  HostName: curiourgeorge1  TaskName: \OSWCleanup  Next Run Time: 2/16/2010 7:26:00 PM  Status: Ready  Logon Mode: Interactive/Background   The status of OSWPrivNet on curiourgeorge1 is:  Folder: \  HostName: curiourgeorge1  TaskName: \OSWPrivNet  Next Run Time: 2/16/2010 2:41:00 PM  Status: Ready  Logon Mode: Interactive/Background |   In this example, all OSWatcher counters are running on the node curiousgeorge1. For this example, OSWFW was installed on a RAC system, and the status for one node was requested. This is why the task OSWPrivNet was included. The task OSWCleanup is also included, and would be even for a stand-alone system.  ***Querying details of a specific counter or task***  This command line option is to display more detailed information about the counters. The syntax is:   |  | | --- | | OSWatcher query {node name} {counter / OSWCleanup / OSWPrivNet} |   To query more extensive details of a specific Counter or task on a node, use the query command line option. Counter names are case sensitive. A special counter name "all" is used to specify all nodes or all counters (which includes the tasks OSWCleanup and, if on RAC, OSWPrivNet).  As an example, to query the counter OSWThread on the node curiousgeorge1:   |  | | --- | | C:\> oswatcher query curiousgeorge1 OSWThread  Status on node curiousgeorge1 of counter OSWThread:   Name: OSWThread  Status: Running  Root Path: C:\Users\Administrator\run\Archive\OSWThread\  Duration: 172800 second(s)  Segment: On  Schedules: On  Segment Max Duration: 3600 second(s)  Run as: SYSTEM   Schedule  Start Date: 2/15/2010   Name: OSWThread\OSWThread  Type: Counter  Output Location: C:\Users\Administrator\run\Archive\OSWThread\CURIOUSGEORGE1 \_OSWThread\_02161437.csv  Append: Off  Circular: Off  Overwrite: Off  Sample Interval: 15 second(s)   Counters:  \Thread(\*)\% Privileged Time  \Thread(\*)\% Processor Time  \Thread(\*)\% User Time  \Thread(\*)\Context Switches/sec  \Thread(\*)\ID Process  \Thread(\*)\ID Thread  \Thread(\*)\Thread State  \Thread(\*)\Thread Wait Reason   The command completed successfully. |   To display details for all the counters, and if on RAC, all the nodes, use the option "all". This option will display all the details for each counter, on all nodes, one at a time..   |  | | --- | | C:\> oswatcher query all |   **Parsing the Output Files**  The files that OSWFW creates can contain more counter outputs than can be easily managed. To break these files down into more manageable sizes, the Windows utility "relog" is used.  Each entry in the OSWFW represents a unique Operating System entity and as such its name can vary from system to system. Other OSWFW capture files are capturing different counters, so follow this procedure to find the names of those objects. The utility "relog" allows you to see all the names of the captured objects. The following is a list of the possible formats of these captured objects:   |  | | --- | | [\\machine\object(parent/instance#index)\counter](file:///\\machine\object(parent\instance%23index)\counter)  [\\machine\object(parent/instance)\counter](file:///\\machine\object(parent\instance)\counter)  [\\machine\object(instance#index)\counter](file:///\\machine\object(instance%23index)\counter)  [\\machine\object(instance)\counter](file:///\\machine\object(instance)\counter)  [\\machine\object\counter](file:///\\machine\object\counter)  \object(parent/instance#index)\counter  \object(parent/instance)\counter  \object(instance#index)\counter  \object(instance)\counter  \object\counter |   Even though the use of the wild card "\*" is possible, it is not a very robust option in this version of the Operating System, and many times does not produce reliable results. As such, a different method is outlined in this document. This method is to put the unique names of the objects of interest into a configuration file and have relog use that. The relog command line syntax can be retrieved from the command line: "relog" This explains, quite well, the syntax of the command and can be referred to if need be.  ***Extracting the Names of the Counters in a Capture File***  To extract the names of all the captured objects in a trace file, and save it off so it can be used to create the configuration file, use this command:   |  | | --- | | relog {trace\_file\_name} -q > {trace\_file\_name}.counter.txt |   This will extract the counters as they are in the log file. Typically, these counters are listed in the order they were created, by Performance Object Counter. If you are only after a specific Counter type for all Threads or Objects, then you can use this file to parse out the specific data.  If you wish to group the counters of a specific type, another technique is to sort the file:   |  | | --- | | relog {trace\_file\_name} -q | sort /+1 > {trace\_file\_name}.sorted.counter.txt |   This output file, {trace\_file\_name}.sorted.counter.txt, now contains just the names of the captured objects and has them sorted. The sorting will group the various counters for a specific OS object. For example, from the entire capture file, once these names are extracted and sorted, the following can be extracted:  [\\GEORGE\Thread(svchost/0)\](file:///\\GEORGE\Thread(svchost\0)\)% Privileged Time  [\\GEORGE\Thread(svchost/0)\](file:///\\GEORGE\Thread(svchost\0)\)% Processor Time  [\\GEORGE\Thread(svchost/0)\](file:///\\GEORGE\Thread(svchost\0)\)% User Time  [\\GEORGE\Thread(svchost/0)\Elapsed](file:///\\GEORGE\Thread(svchost\0)\Elapsed) Time  [\\GEORGE\Thread(svchost/0)\ID](file:///\\GEORGE\Thread(svchost\0)\ID) Process  [\\GEORGE\Thread(svchost/0)\ID](file:///\\GEORGE\Thread(svchost\0)\ID) Thread  [\\GEORGE\Thread(svchost/0)\Priority](file:///\\GEORGE\Thread(svchost\0)\Priority) Base  [\\GEORGE\Thread(svchost/0)\Priority](file:///\\GEORGE\Thread(svchost\0)\Priority) Current  [\\GEORGE\Thread(svchost/0)\Thread](file:///\\GEORGE\Thread(svchost\0)\Thread) State  [\\GEORGE\Thread(svchost/0)\Thread](file:///\\GEORGE\Thread(svchost\0)\Thread) Wait Reason  [\\GEORGE\Thread(svchost/0#1)\](file:///\\GEORGE\Thread(svchost\0%231)\)% Privileged Time  [\\GEORGE\Thread(svchost/0#1)\](file:///\\GEORGE\Thread(svchost\0%231)\)% Processor Time  [\\GEORGE\Thread(svchost/0#1)\](file:///\\GEORGE\Thread(svchost\0%231)\)% User Time  [\\GEORGE\Thread(svchost/0#1)\Elapsed](file:///\\GEORGE\Thread(svchost\0%231)\Elapsed) Time  [\\GEORGE\Thread(svchost/0#1)\ID](file:///\\GEORGE\Thread(svchost\0%231)\ID) Process  [\\GEORGE\Thread(svchost/0#1)\ID](file:///\\GEORGE\Thread(svchost\0%231)\ID) Thread  [\\GEORGE\Thread(svchost/0#1)\Priority](file:///\\GEORGE\Thread(svchost\0%231)\Priority) Base  [\\GEORGE\Thread(svchost/0#1)\Priority](file:///\\GEORGE\Thread(svchost\0%231)\Priority) Current  [\\GEORGE\Thread(svchost/0#1)\Thread](file:///\\GEORGE\Thread(svchost\0%231)\Thread) State  [\\GEORGE\Thread(svchost/0#1)\Thread](file:///\\GEORGE\Thread(svchost\0%231)\Thread) Wait Reason  [\\GEORGE\Thread(svchost/0#2)\](file:///\\GEORGE\Thread(svchost\0%232)\)% Privileged Time  [\\GEORGE\Thread(svchost/0#2)\](file:///\\GEORGE\Thread(svchost\0%232)\)% Processor Time  [\\GEORGE\Thread(svchost/0#2)\](file:///\\GEORGE\Thread(svchost\0%232)\)% User Time  [\\GEORGE\Thread(svchost/0#2)\Elapsed](file:///\\GEORGE\Thread(svchost\0%232)\Elapsed) Time  [\\GEORGE\Thread(svchost/0#2)\ID](file:///\\GEORGE\Thread(svchost\0%232)\ID) Process  [\\GEORGE\Thread(svchost/0#2)\ID](file:///\\GEORGE\Thread(svchost\0%232)\ID) Thread  [\\GEORGE\Thread(svchost/0#2)\Priority](file:///\\GEORGE\Thread(svchost\0%232)\Priority) Base  [\\GEORGE\Thread(svchost/0#2)\Priority](file:///\\GEORGE\Thread(svchost\0%232)\Priority) Current  [\\GEORGE\Thread(svchost/0#2)\Thread](file:///\\GEORGE\Thread(svchost\0%232)\Thread) State  [\\GEORGE\Thread(svchost/0#2)\Thread](file:///\\GEORGE\Thread(svchost\0%232)\Thread) Wait Reason  From above, we see that the Machine name is "GEORGE," while the object is "Thread" and the parent executable is "svchost." In this case, the parent executable, svchost/0 (the base instance) is listed along with three of its indexes. Each index is a separate thread. Even though each thread has an Index ID, this number is not the ID Thread. Finding the ID Thread for a particular thread is a little more complex and is outlined later in this document. The last part of the captured object name is the actual counter, for instance "Thread Wait Reason" or "Thread State."  Other techniques can also be used. If there are Unix utilities installed on your Windows system, you can use the utility "grep" to extract just the "Thread State" counters, or any other combination of strings.  ***Extracting Specific Counters from a Capture File***  Since this part of this guide concerns reducing the amount of information in one of the capture files, we are going to extract all of the counters for the base executable and just one of its child Threads. To do this we copy the file we created above to a file we are to modify. We do this just in case there will be a different combination of objects we wish to extract later.   |  | | --- | | copy {trace\_file\_name}.counter.txt thread\_svchost\_0.txt |   Edit the thread\_svchost\_0.txt file to contain only the counters that refer to svchost/0 and svchost/0#1.  [\\GEORGE\Thread(svchost/0)\](file:///\\GEORGE\Thread(svchost\0)\)% Privileged Time  [\\GEORGE\Thread(svchost/0)\](file:///\\GEORGE\Thread(svchost\0)\)% Processor Time  [\\GEORGE\Thread(svchost/0)\](file:///\\GEORGE\Thread(svchost\0)\)% User Time  [\\GEORGE\Thread(svchost/0)\Elapsed](file:///\\GEORGE\Thread(svchost\0)\Elapsed) Time  [\\GEORGE\Thread(svchost/0)\ID](file:///\\GEORGE\Thread(svchost\0)\ID) Proces  [\\GEORGE\Thread(svchost/0)\ID](file:///\\GEORGE\Thread(svchost\0)\ID) Thread  [\\GEORGE\Thread(svchost/0)\Priority](file:///\\GEORGE\Thread(svchost\0)\Priority) Base  [\\GEORGE\Thread(svchost/0)\Priority](file:///\\GEORGE\Thread(svchost\0)\Priority) Current  [\\GEORGE\Thread(svchost/0)\Thread](file:///\\GEORGE\Thread(svchost\0)\Thread) State  [\\GEORGE\Thread(svchost/0)\Thread](file:///\\GEORGE\Thread(svchost\0)\Thread) Wait Reason  [\\GEORGE\Thread(svchost/0#1)\](file:///\\GEORGE\Thread(svchost\0%231)\)% Privileged Time  [\\GEORGE\Thread(svchost/0#1)\](file:///\\GEORGE\Thread(svchost\0%231)\)% Processor Time  [\\GEORGE\Thread(svchost/0#1)\](file:///\\GEORGE\Thread(svchost\0%231)\)% User Time  [\\GEORGE\Thread(svchost/0#1)\Elapsed](file:///\\GEORGE\Thread(svchost\0%231)\Elapsed) Time  [\\GEORGE\Thread(svchost/0#1)\ID](file:///\\GEORGE\Thread(svchost\0%231)\ID) Process  [\\GEORGE\Thread(svchost/0#1)\ID](file:///\\GEORGE\Thread(svchost\0%231)\ID) Thread  [\\GEORGE\Thread(svchost/0#1)\Priority](file:///\\GEORGE\Thread(svchost\0%231)\Priority) Base  [\\GEORGE\Thread(svchost/0#1)\Priority](file:///\\GEORGE\Thread(svchost\0%231)\Priority) Current  [\\GEORGE\Thread(svchost/0#1)\Thread](file:///\\GEORGE\Thread(svchost\0%231)\Thread) State  [\\GEORGE\Thread(svchost/0#1)\Thread](file:///\\GEORGE\Thread(svchost\0%231)\Thread) Wait Reason  Save this file. We now run relog to extract the values of these counters from the original log file:   |  | | --- | | relog {trace\_file\_name} -cf thread\_svchost\_0.txt -f csv -o thread\_svchost\_0.csv |   This command will create a comma-delimited file that can be brought up in Excel or other spread-sheet-like application. This "csv" can now be imported into Excel to use its graphing capabilities, or to further examine the file.  Keep in mind that if Excel is to be used, some versions have a limit as to the number of columns one spreadsheet can have (256 columns in Excel 2000 so check your version's limits). Each counter will be a column in Excel. Each row will be the counter's value. The number of rows this will resolve to will depend on the command line options issued when OSWFW was started that created these log files.  Depending on the size of the file and number of counters listed, this extraction could take some time. It was found that the smaller number of counters in the configuration file, the quicker this extract takes. It might be faster to perform various small extracts and concatenate the output files together in the end. This determination is left to the reader.  ***Finding ID Thread from Thread Instance Number (Finding a Thread in a Haystack)***  OSWFW, by default, is configured to capture the ID Tread counter. All Performance Counters, on the other hand, use the "Thread Instance Number" to delineate a thread spawned by a particular process. This Thread Instance Number is a monotonically increasing number, starting from zero, which identifies a thread in a particular process. In conjunction with the Process Name and thread Instance Number, there is also the ID Thread, which is a globally unique number assigned to each Thread. Unfortunately, logman does not put the ID Thread as part of counter name, but only the Process Name and the Thread Instance Number so one has to capture the ID Thread as a separate counter. This counter does not change during the lifetime of the Thread. Depending on how often the Parent process creates and destroys threads, this number can be reused. The global ID Thread, on the other hand, might repeat, but that case is exceptional and today computers are not manufactured with that much memory to accommodate that many threads.  When the Oracle Database views V$PROCESS.SPID or V$SESSION.PROCESS are queried for the Process ID of a particular process, both the Process ID and ID Thread are returned.   |  | | --- | | SQL> SELECT PROGRAM, SPID, ADDR FROM V$PROCESS; |   Since the Windows Operating System is thread based, the Process ID alone will not give enough information to trace down the information that OSWatcher delivers, so the ID Thread is needed. Unfortunately, the Operating System logs that can be used (the Counters) do not use the Process ID or the ID Thread but use the Process Name and the Thread Instance Number. This section describes how to find the ID Thread in the logs and relate them to the Process Name and Thread Instance Number so the information in the logs for the ID Thread of interest can be extracted from the connection log files.  OSWFW, by default, is configured to capture the ID Tread counter by using the "\Thread(\*)\ID Thread" counter. This counter will log the ID Thread for all threads in the system (because of the use of the wildcard "\*"). This static counter does not change for the life of the thread. All Performance Counters, use the  Thread Instance Number  to delineate a thread spawned by a particular process. This Thread Instance Number is a monotonically increasing number, starting from zero, which identifies a thread in a particular process, while the ID Thread is a globally unique number assigned to the thread when it is created.  If you wish to find the performance counter that corresponds to the ID Thread of interest, you will have to find the Process Name and Thread Instance Number for that ID Thread. This counter does not change during the lifetime of the Thread.  To extract the ID Thread for a particular thread, first all the ID Threads must be extracted from the log file. This can be done using the wildcard "\*". The syntax of relog is a little touchy, so if the following format does not work, use the method outlined above to create a configuration file from the exact counter names. To extract the ID Thread counters and their values, issue the following command:   |  | | --- | | relog {trace\_file\_name} -q > {trace\_file\_name}.counter.txt |   Sorting at this point will not assist as the log file puts all the ID Thread counters together. This extract does not include the values of the counters, just the counter's names. Once this file is created, copy it to another file that will be edited to leave only the ID Thread counter names.   |  | | --- | | copy {trace\_file\_name}.counter.txt IDthread.txt |   Edit this file to leave only the entries that are of this format:  [\\Machine\Thread({Parent](file:///\\Machine\Thread(%7bParent) /Instance#Index})\ID Thread  Since it is expected that the reader will be only interested in only one process parent, those that are associated with Oracle, leave only those with the process parent "oracle," "TNSLSNR," and "oradim." As example the list will take on this appearance:  [\\GEORGE\Thread(TNSLSNR/0)\ID](file:///\\GEORGE\Thread(TNSLSNR\0)\ID) Thread  [\\GEORGE\Thread(TNSLSNR/1)\ID](file:///\\GEORGE\Thread(TNSLSNR\1)\ID) Thread  [\\GEORGE\Thread(TNSLSNR/2)\ID](file:///\\GEORGE\Thread(TNSLSNR\2)\ID) Thread  [\\GEORGE\Thread(oracle/0)\ID](file:///\\GEORGE\Thread(oracle\0)\ID) Thread  [\\GEORGE\Thread(oracle/1)\ID](file:///\\GEORGE\Thread(oracle\1)\ID) Thread  [\\GEORGE\Thread(oracle/2)\ID](file:///\\GEORGE\Thread(oracle\2)\ID) Thread  [\\GEORGE\Thread(oracle/3)\ID](file:///\\GEORGE\Thread(oracle\3)\ID) Thread  [\\GEORGE\Thread(oracle/4)\ID](file:///\\GEORGE\Thread(oracle\4)\ID) Thread  [\\GEORGE\Thread(oracle/5)\ID](file:///\\GEORGE\Thread(oracle\5)\ID) Thread  [\\GEORGE\Thread(oracle/6)\ID](file:///\\GEORGE\Thread(oracle\6)\ID) Thread  [\\GEORGE\Thread(oracle/7)\ID](file:///\\GEORGE\Thread(oracle\7)\ID) Thread  [\\GEORGE\Thread(oracle/8)\ID](file:///\\GEORGE\Thread(oracle\8)\ID) Thread  [\\GEORGE\Thread(oracle/9)\ID](file:///\\GEORGE\Thread(oracle\9)\ID) Thread  [\\GEORGE\Thread(oracle/10)\ID](file:///\\GEORGE\Thread(oracle\10)\ID) Thread  [\\GEORGE\Thread(oracle/11)\ID](file:///\\GEORGE\Thread(oracle\11)\ID) Thread  [\\GEORGE\Thread(oracle/12)\ID](file:///\\GEORGE\Thread(oracle\12)\ID) Thread  [\\GEORGE\Thread(oracle/13)\ID](file:///\\GEORGE\Thread(oracle\13)\ID) Thread  [\\GEORGE\Thread(oracle/14)\ID](file:///\\GEORGE\Thread(oracle\14)\ID) Thread  [\\GEORGE\Thread(oracle/15)\ID](file:///\\GEORGE\Thread(oracle\15)\ID) Thread  [\\GEORGE\Thread(oracle/16)\ID](file:///\\GEORGE\Thread(oracle\16)\ID) Thread  [\\GEORGE\Thread(oracle/17)\ID](file:///\\GEORGE\Thread(oracle\17)\ID) Thread  [\\GEORGE\Thread(oradim/0)\ID](file:///\\GEORGE\Thread(oradim\0)\ID) Thread  This list contains the process parents of the Oracle Listener (TNSLSNR), the Oracle executable (oracle) and the Database Configuration Assistant (oradim). This file will be used to extract just the ID Threads.   |  | | --- | | relog {trace\_file\_name} -cf IDThread.txt -f csv -o IDThread.csv |   The output file, IDThread.csv, now contains all the ID Threads for the Oracle Threads. The simplest method to use at this point is to bring up the file in Excel to find the ID Thread. It will be the number that was found from V$PROCESS or V$SESSION. When selecting the process ID from V$SESSION, remote sessions will have the Process ID of the Client process also. The format will be:   |  | | --- | | Client ID Thread:Server ID Thread |   The select statement to use to find the ID Thread is:   |  | | --- | | SQL> SELECT SID, PROCESS, PADDR FROM V$SESSION;         SID PROCESS      PADDR  ---------- ------------ --------         145 3444         3425231C         147 2948         34251D2C         151 2572:480     3425290C         154 4092         3425056C         159 2232         34250B5C         160 2596         3424EDAC         161 2324         3424E7BC         162 2308         3424E1CC         163 1112         3424DBDC         164 3300         3424D5EC         165 4068         3424CFFC         166 2380         3424CA0C         167 920          3424C41C         168 3456         3424BE2C         169 2916         3424B83C         170 3220         3424B24C   16 rows selected. |   In this example, the SQLPLUS.EXE ID Thread is 480 while the SQL\*Plus Process ID is 2572. If the Oracle background threads are under scrutiny, use the V$PROCESS view to find the ID Thread:   |  | | --- | | SQL> SELECT PROGRAM, SPID, ADDR FROM V$PROCESS;   PROGRAM             SPID  ADDR  ------------------- ----- --------  PSEUDO                    3424AC5C  ORACLE.EXE (PMON)   3220  3424B24C  ORACLE.EXE (PSP0)   2916  3424B83C  ORACLE.EXE (MMAN)   3456  3424BE2C  ORACLE.EXE (DBW0)   920   3424C41C  ORACLE.EXE (LGWR)   2380  3424CA0C  ORACLE.EXE (CKPT)   4068  3424CFFC  ORACLE.EXE (SMON)   3300  3424D5EC  ORACLE.EXE (RECO)   1112  3424DBDC  ORACLE.EXE (CJQ0)   2308  3424E1CC  ORACLE.EXE (MMON)   2324  3424E7BC  ORACLE.EXE (MMNL)   2596  3424EDAC  ORACLE.EXE (D000)   940   3424F39C  ORACLE.EXE (S000)   3180  3424F98C  ORACLE.EXE (QMNC)   4092  3425056C  ORACLE.EXE (J001)   2232  34250B5C  ORACLE.EXE (q000)   2948  34251D2C  ORACLE.EXE (q001)   3444  3425231C  ORACLE.EXE (SHAD)   3344  3425290C   19 rows selected. |   In the case where the intent is to isolate which thread the SQL\*Plus session is part of, take the PADDR from V$SESSION (3425290C) and find it in V$PROCESS. This will result in the IDThread of 3344 (ORACLE.EXE (SHAD) 3344 3425290C).  Once the ID Thread of in interest is found in the IDThread.csv file, the name of the counter will be the header for that column. In the case where the PMON thread is to be examined, search for the ID Thread 3220. In this case it will have the counter name of:  [\\GEORGE\Thread\oracle(3)\ID](file:///\\GEORGE\Thread\oracle(3)\ID) Thread = 3220  NOTE: The Thread Instance Number does not have any special meaning. For instance, PMON may not always have a Thread Instance Number of three. During the startup of the Oracle Database Service, the ORACLE.EXE will spawn threads and then close them, thus the next thread that is created might get the Thread Instance Number of the recently closed thread. Thread Instance Numbers are recycled.  After all of this work, the ID Thread can now be related to the parent Process Name and the Thread Instance Number. From this information, all the counters for this particular thread can be extracted from the log file. In the case mentioned above, where the interest lies in PMON, the counter "[\\GEORGE\Thread\oracle(3)\ID](file:///\\GEORGE\Thread\oracle(3)\ID) Thread" is extracted.  But wait, there's more. The use of wild cards would come in quite handy at this point in the process, but lacking that, the counters for the particular thread have to be pulled from the list of all counters created earlier.   |  | | --- | | type {trace\_file\_name}.counter.txt | sort /+1 > Thread\_oracle\_3.txt |   This sort will combine all the counters based on their name, rather than the order they were gathered in. From this new file it should be easy to get the counters for [\\GEORGE\Thread\oracle(3](file:///\\GEORGE\Thread\oracle(3)). Once all the extraneous counters are removed, the file should contain something like:  [\\GEORGE\Thread(oracle/3)\](file:///\\GEORGE\Thread(oracle\3)\)% Privileged Time  [\\GEORGE\Thread(oracle/3)\](file:///\\GEORGE\Thread(oracle\3)\)% Processor Time  [\\GEORGE\Thread(oracle/3)\](file:///\\GEORGE\Thread(oracle\3)\)% User Time  [\\GEORGE\Thread(oracle/3)\Elapsed](file:///\\GEORGE\Thread(oracle\3)\Elapsed) Time  [\\GEORGE\Thread(oracle/3)\ID](file:///\\GEORGE\Thread(oracle\3)\ID) Process  [\\GEORGE\Thread(oracle/3)\ID](file:///\\GEORGE\Thread(oracle\3)\ID) Thread  [\\GEORGE\Thread(oracle/3)\Priority](file:///\\GEORGE\Thread(oracle\3)\Priority) Base  [\\GEORGE\Thread(oracle/3)\Priority](file:///\\GEORGE\Thread(oracle\3)\Priority) Current  [\\GEORGE\Thread(oracle/3)\Thread](file:///\\GEORGE\Thread(oracle\3)\Thread) State  [\\GEORGE\Thread(oracle/3)\Thread](file:///\\GEORGE\Thread(oracle\3)\Thread) Wait Reason  Now you can extract the counters for the thread of interest:   |  | | --- | | relog {trace\_file\_name} -cf Thread\_oracle\_3.txt -f csv -o Thread\_oracle\_3.csv |   The file Thread\_oracle\_3.csv can now be viewed in Excel, or some other editor.  ***References from Microsoft***  Windows NT 4.0 Resource Kit  Chapter 10 - About Performance Monitor  <http://www.microsoft.com/technet/archive/ntwrkstn/reskit/02perfmn.mspx?mfr=true>  How To Troubleshoot High CPU Utilization of an MTS or COM+ Process  <http://support.microsoft.com/kb/258833>  **Diagnostic Data Output**  As stated above, when OSWFW is started for the first time it creates the archive subdirectory under the OSWFW installation directory. The archive directory contains several subdirectories, one for each data collection. These directories are named OSWMemory, OSWNetstat, OSWPhysicalDisk, OSWProcess, OSWProcessor, OSWServer\_Work\_Queue, OSWSystem, and OSWThread. One file per hour will be generated in each of the subdirectories. A new file is created after each hour that OSWFW is running. The file will be in the following format:   |  | | --- | | %COMPUTERNAME%\_OSW<Performance Object>\_MMDDHHMM\_nnn.csv |   The format of MMDDHHMM is Month, Day, Hour, and Minute. The nnn is a numerical value, which starts at 001 and increases by one, but typically will not is this configuration.  The descriptions of these Counters can be found by bringing up the Windows Performance monitor. First open the Taskbar, Start, Run. In the Run prompt screen, type in "perfmon.msc", without the quotes. In the Performance Microsoft Management Console, the lower right section will list various Counters. Right click this part of the window and select Add Counters. In the Add Counters window the Counter of interest can be brought up and the Explain button can be pressed to bring up the description.  At the end of this document are links to attachments which are text files listing all the Counters and their descriptions for the verions of Windows.  They were acquired using Microsoft's PowerShell v2.0 which is installed either by default or through patching the Windows Operating System.  The format of a Counter's name is:    [\\Computer](file:///\\Computer) name\Performance object\Counter\instance  For example:  [\\GEORGE\Logicaldisk\](file:///\\GEORGE\Logicaldisk\)% Disk Time\C:  This is the percentage of the elapsed time that the logical C: disk drive was busy servicing read or write requests.  **Known Issues**  OSWFW does not run in a directory with spaces in it.  This is planned to be fixed in the next release.  If OSWFW is not run as Administrator, it may faslely report it can't run on a remote drive when it is a local drive.  This is due to the OS utilities being called can't be run except by the Administrator.  **Download**  Current OSWatcher for Windows is Version 2.5.1 May 13, 2013  Click [here](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:OSWFW&clickstream=yes%20\o%20Click%20here%20to%20download%20OSWatcher%20for%20Windows%20\t%20_self) to download the zip file containing OSWFW.  The list of counters can be downloaded via the following links:  [Windows2003R2x64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS2003R2X86&clickstream=yes%20\o%20Counter%20list%20for%20Windows%202003%20R2%20x86) [Windows2003R2x86Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS2003R2X86&clickstream=yes%20\o%20Counter%20list%20for%20Window%20s2003%20R2%20x86) [Windows2003x64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS2003X64&clickstream=yes%20\o%20Counter%20list%20for%20Windows%202003%20x64) [Windows2008R2x64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS2008R2X64&clickstream=yes%20\o%20Counter%20list%20for%20Windows%202008%20R2%20x64) [Windows2008x86Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS2008X86&clickstream=yes%20\o%20Counter%20list%20for%20Windows%202008%20x86) [Windows7x64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS7X64&clickstream=yes%20\o%20Counter%20list%20for%20Windows%207%20x64) [Windows7x86Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS7X86&clickstream=yes%20\o%20Counter%20list%20for%20Windows%207%20x86) [Windows8x64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS8X64&clickstream=yes%20\o%20Counter%20list%20for%20Windows%208%20x64) [Windows8x86Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS8X86&clickstream=yes%20\o%20Counter%20list%20for%20Windows%208%20x86) [WindowsXPx64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWSXPX64&clickstream=yes%20\o%20Counter%20list%20for%20Windows%20XP%20x64) WindowsXPx86Counters [Wubdiws2012x64Counters](https://support.oracle.com/epmos/main/downloadattachmentprocessor?parent=DOCUMENT&sourceId=433472.1&attachid=433472.1:WINDOWS2012X64&clickstream=yes%20\o%20Windows%202012%20x64%20Counters)  If a file download dialog box does not appear when clicking on the above link, you may need to clear your web browser's cache and/or restart your web browser. If you are still unable to download the file, you may request that we email you a copy: [Kevin.Reardon@oracle.com](mailto:Kevin.Reardon@oracle.com)  **Reporting Feedback**  If you encounter problems running OSWFW, which is not listed under the Known Issue section, or would like to provide comments or feedback about OSWFW (including enhancement requests) please send email to [Kevin.Reardon@oracle.com](mailto:Kevin.Reardon@oracle.com).  **Sending Files To Support**  Create a Zip file of the archive directory and upload it to MyOracleSupport for Oracle Support. |