|  |  |  |
| --- | --- | --- |
| **Oracle Grid Infrastructure: How to Troubleshoot Node Reboots/Evictions or Abrupt Recycle of CRS due to the Escalation of a Member Kill Request to a Node Kill Request (Doc ID 1549977.1)** | [To Bottom](https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=503991598304626&id=1549977.1&_adf.ctrl-state=2vjdv6fkr_72%20\o%20To%20Bottom) |  |





|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **In this Document**   |  |  | | --- | --- | |  | [Purpose](https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=503991598304626&id=1549977.1&_adf.ctrl-state=2vjdv6fkr_72%20\l%20PURPOSE) | |  | [Troubleshooting Steps](https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=503991598304626&id=1549977.1&_adf.ctrl-state=2vjdv6fkr_72%20\l%20TRBLSHOOT) |  |  |  | | --- | --- | |  | [References](https://support.oracle.com/epmos/faces/DocumentDisplay?_afrLoop=503991598304626&id=1549977.1&_adf.ctrl-state=2vjdv6fkr_72%20\l%20REF) |     **APPLIES TO:**  Oracle Database - Enterprise Edition - Version 11.2.0.1 and later  Information in this document applies to any platform.  **PURPOSE**  When an evicted instance does not abort, the member kill request is sent to the ocssd.bin daemon of CRS of the node with the problem instance.  If the ocssd.bin daemon can not abort the problem instance in 30 seconds, the member kill request is escalated to the node kill request.  The result of the node kill request is that the node is rebooted unless CRS shuts down gracefully.  **TROUBLESHOOTING STEPS**  **To find out if the node reboot or CRS recycle is due to the node kill request as a result of the escalation of member kill request, please look for one of the following three in the log files:**  1) The alert.log of the instance that is evicting the problem instance shows  Remote instance kill is issued    2) The ocssd.log of the node with the instance that is evicting the problem instance shows  clssgmMbrKillEsc: Escalating node <node number of the problem node> Member request 0x00000001 Member success  clssnmMarkNodeForRemoval: node <node number of the problem node>, <the problem node name> marked for removal  clssnmKillNode: node <node number of the problem node> (<the problem node name>) kill initiated    3) The cluster alert.log (alert<node name>.log) in GI\_HOME/log/<node name> directory shows  CRS-1663:Member kill issued    **The following is the sequence of events:**  1) When an instance does not respond, then another instance evicts the problem instance.  2) If the problem instance does not abort itself, then the evicting instance issues a "member kill" request to the CRS (specifically to the ocssd.bin daemon of CRS) of the local node.  3) The ocssd.bin daemon of the local node then sends a "member kill" request to the ocssd.bin of the problem node.  4) The ocssd.bin of the problem node then issues "kill -9" to the instance background processes and start terminating the instance.  5) If one or more processes of the problem instance do not die within 30 seconds, the member kill request times out.  6) The original node that send the member kill request then escalates the member kill request to a node kill request and send the node kill request to the ocssd.bin daemon of problem node.  7) After receiving the node kill request via network or via voting disk, the ocssd.bin of the problem node brings itself down.  8) If graceful shutdown of ocssd.bin daemon is successful (this requires other resources dependent on ocssd.bin like database, asm, and crs must also successfully come down), then CRS restarts al the daemons again without rebooting the node.  9) If any of the resource or ocssd.bin can not shutdown gracefully, then the node reboots.  **The most common causes of the member kill request escalation to the node kill request are:**  1) On Unix/Linux, one or more processes for the instance are in a state that can not be terminated a "kill -9 <OS pid>" command.  An example of such process state is "D" which is " Uninterruptible sleep (usually IO)".  Issuing "ps -efl" will display the process state in the second column, and OSWatcher captures this output, so having OSWatcher set up and running all the time is important.  The reason that the process goes into state D is because the process is issued an IO request to OS and is waiting for OS to complete the IO.  Until the IO request is completed, the process can not be killed.  If ocssd.bin is trying to kill a process that is waiting for IO to complete and if that IO request is not completed in 30 seconds or less, the member kill request will time out and get escalated to the node kill request.  Because the process in state D is waiting for IO request to OS to get completed, any hang or delay coming out of state D is a result of OS taking too long for IO.  This is OS or Storage problem, so please consult the system admin or storage admin.  2) On Unix/Linux, one or more Oracle database/asm processes that are killed do not get reaped by the parent process and become a defunct or zombie process.  The parent process for all Oracle database/ASM processes are the OS init daemon (OS pid 1), so the init daemon needs to clean up and reap when the Oracle database/ASM processes get killed.  The killed process becomes a defunct or zombie process (state Z) until this happens.  As in case 1 above, OSWatcher output will show the process state.  The init daemon is OS daemon, so if the init daemon can not reap and clean the killed processes quickly, then the problem is with OS.  Please consult the system admin to find out the reason for this.  3) The load on the problem node is heavy, so either CPU is almost 100% used or memory starvation is causing heavy paging and swapping.  As a result, ocssd.bin can not kill all processes of evicted database or asm instance in 30 seconds or less.  Please consult with the system admin to find out the reason that the CPU or memory starvation is happening.  OSWatcher output will show the CPU and memory usage as well as paging and swapping statistics.  **Note:**  The member kill request is issued because an evicted instance is not aborting, so an instance eviction is the starting event that led to the node eviction. Therefore, finding out the reason that an instance is evicted needs to be done separately.  Please refer to the note 1549135.1 - Troubleshooting Instance Evictions (Instance terminates with ORA-29740, Instance Abort, Instance Kill)    **Examples**  Note 1493110.1  Database Instance Hung and One Of RAC Nodes Rebooted as Membership Kill Escalated  **REFERENCES**  [NOTE:1549135.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1549977.1&id=1549135.1) - Troubleshooting Instance Evictions (Instance terminates with ORA-29740, Instance Abort, Instance Kill)  [NOTE:1050693.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=1549977.1&id=1050693.1) - Troubleshooting 11.2 Clusterware Node Evictions (Reboots) |