

# Oracle Database 12c Release 2 Oracle Autonomous Health Framework

ORACLE WHITE PAPER | MARCH 2017





## Table of Contents

Introduction	1
What Issues are Addressed by Oracle Autonomous Health Framework?	2
Availability Issues	2
Server Availability Issues	2
Database Availability Issues	2
Performance Issues	2
Database Server Performance Issues	3
Database Client-Caused Performance Issues	3
How Does Oracle Autonomous Health Framework Address These Issues?	3
Generates Diagnostic Metric View of Cluster and Databases	3
Cluster Health Monitor Architecture	3
Using Cluster Health Monitor to Collect Metrics	4
Establishes Baseline and Maintains Best Practice Configurations	6
Cluster Verification Utility Architecture	6
Using Cluster Verification Utility to Perform Health Checks	7
Maintains Compliance with Best Practices and Alerts Vulnerabilities to Known Issues	8
ORAchk Architecture	8
Using ORAchk to Maintain Compliance	8
Autonomously Monitors Performance and Manages Resources to Meet SLAs	10
Quality of Service Management Architecture	10
Using Quality of Service Management to Manage Resources and Maintain SLAs	11



Autonomously Preserves Database Availability and Performance During Hangs	15
Hang Manager Architecture	15
Using Hang Manager to Resolve Hangs	15
Autonomously Preserves Server Availability By Relieving Memory Stress	16
Memory Guard Architecture	16
Using Memory Guard to Relieve Memory Stress	17
Discovers Potential Cluster & Database Problems - Notifies with Corrective Actions	18
Cluster Health Advisor Architecture	18
Using Cluster Health Advisor for Prognosis of Potential Threats	19
Speeds Issue Diagnosis, Triage and Resolution	19
Trace File Analyzer Architecture	19
Using Trace File Analyzer to Collect Relevant Information for an Issue	20
Oracle Autonomous Health Framework in Oracle Cluster Domain	21
Conclusion	22



## Introduction

Businesses today are becoming global. They have customers across the world using their applications and performing transactions 24x7. These applications are powered by databases that provide relevant data to applications through various database services. Therefore, in order to provide customers a continuous and consistent application experience, businesses need to ensure that their underlying databases are running smoothly 24x7. This means that databases not only need continuous availability, but also provide consistent performance. Therefore, any issues affecting this availability and performance needs to be addressed and resolved quickly to bring these databases back fully online.

Currently, these issues are resolved manually where human reaction time causes a delay in identification, diagnosis, and resolution. This delay can prove to be costly by adversely affecting on-going business transactions and user experience.

Oracle Autonomous Health Framework (AHF) presents the next generation of tools as components, which autonomously work 24x7 to keep database systems healthy and running while minimizing human reaction time. These components include both existing tools as components in ORAchk, Cluster Verification Utility, Trace File Analyzer, Cluster Health Monitor, Quality of Service Management, Memory Guard, and new components in Cluster Health Advisor and Hang Manager as shown in Figure 1.

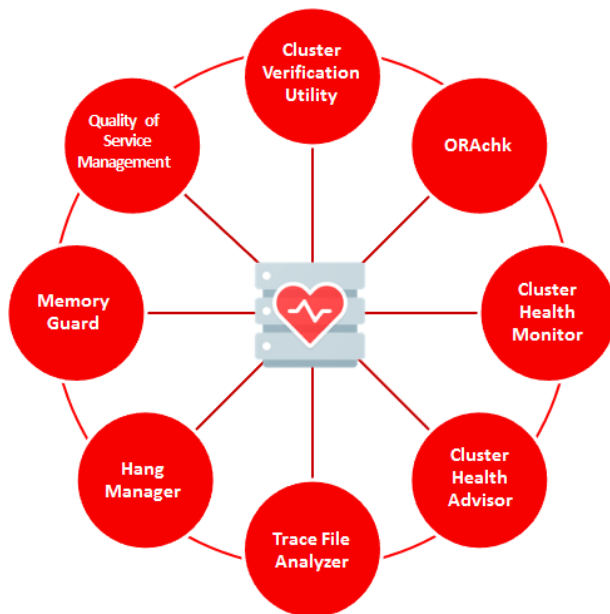


Figure 1: Oracle Autonomous Health Framework with its components

Oracle Autonomous Health Framework components work together in daemon mode to address issues faced by Database administrators and System administrators in areas of availability and performance.



## What Issues are Addressed by Oracle Autonomous Health Framework?

Oracle Autonomous Health Framework addresses availability and performance issues in system administrator and database administrator spaces. The responsibilities of system administrators include managing hardware resources - servers, OS, network, storage, and Oracle Grid Infrastructure (GI) stack. They are operationally responsible for installation, patching, upgrades and resource availability of these hardware resources. On the other hand, database administrators manage the database stack and the associated services. They are operationally responsible for installation, patching, upgrades, resource allocations, and SLAs of these database resources. Oracle AHF assists in fulfilling both these responsibilities by autonomously monitoring and managing the hardware resources as well as the database stack.

While many of Oracle Autonomous Health Framework components can be used interactively during installation, patching, and upgrading, their use within AHF is focused on operational runtime issues and either preventing their occurrence or mitigating their impact. These include the following availability and performance issues.

### Availability Issues

Availability issues are runtime issues that can threaten availability of the software stack either through a software issue (DB, GI, O/S) or underlying hardware resources (CPU, memory, network, storage). The specific availability issues addressed by Oracle Autonomous Health Framework can be grouped into server and database issues.

#### Server Availability Issues

Server availability issues can cause a server to be evicted from its cluster and shut down all database instances running there. Specific issues addressed by Oracle Autonomous Health Framework are:

- » Memory Stress caused by a node running out of free physical memory. This results in the O/S Swapper process running for extended periods moving memory to and from disk and preventing time critical cluster processes from running thereby causing the node to be evicted.
- » Network issues, for example, network congestion on private interconnect caused by a change in configuration. This can result in excessive latency in time-critical internode or storage I/O or dropped packets causing database instances to be non-responsive or ultimately node eviction.
- » Hardware issues that are not possible to anticipate. For example, network failures on private interconnect due to a network card failure or cable pull. This will immediately result in an evicted node.

#### Database Availability Issues

Database availability issues can cause a database or one of its instances to become unresponsive and thus unavailable. Specific issues addressed by Oracle Autonomous Framework are:

- » Runaway Queries or Hangs that can deny critical database resources in locks, latches, CPU to other sessions. This can result in a database instance or the entire database being non-responsive to applications.
- » Denial-of-Service attacks, rogue workloads or software bugs. These can cause a database or instance to be unresponsive.
- » Software configuration or permission changes, for example, incorrect permissions on oracle.bin. This can also cause database outages due to the inability to create sessions and can be very difficult to troubleshoot.

### Performance Issues

Performance issues are runtime issues that threaten performance of the system as seen by database clients or applications either through software issues (bugs, configuration, contention, etc.) or client issues (demand, query types, connection management, etc.). The specific performance issues addressed by Oracle Autonomous Health Framework can be grouped into database server and client-caused issues.



## Database Server Performance Issues

Database server performance issues can result in a lower than optimum performance of database servers. Specific issues addressed by Oracle Autonomous Health Framework are:

- » Performance issues that can be caused by deviations from best practices in configuration.
- » Issues that can be caused by bottlenecked resources such as insufficient storage disks, high block contention in global cache, poorly constructed SQL, or a session that may be causing others to slow down waiting for it to release its resources or complete.
- » Issues or bugs that are already known and can be fixed with upgrades, patches, or workarounds.

## Database Client-Caused Performance Issues

Database clients can impact the performance of individual database instances or the entire database system. Specific issues addressed by Oracle Autonomous Framework are:

- » When a server hosts more databases instances than its resources and client load can handle, performance suffers due to waiting for CPU, I/O, or memory. This misconfiguration or oversubscription of CPUs, I/O or memory can prevent critical or background processes from running in a timely manner.
- » Degraded performance due to misconfigured parameters in SGA versus PGA allocation, number of sessions/processes, CPU counts, etc. based upon type of workload and level of concurrency required.
- » Client demand exceeds server or database capacity.

Thus, Oracle Autonomous Health Framework addresses a wide variety of operational runtime issues in areas of availability and performance for both hardware and software resources of the database system.

## How Does Oracle Autonomous Health Framework Address These Issues?

Oracle Autonomous Health Framework components work 24x7 in daemon mode to address availability and performance issues, and ensure high availability and consistent performance for the database system. They collaborate with each other to provide a framework that:

- » Continuously monitors database systems, collects OS metrics and generates diagnostic views of clusters and their hosted databases
- » Establishes baseline and maintains best practice configurations
- » Maintains compliance with best practices and alerts vulnerabilities to known issues
- » Monitors performance and manages resources to meet SLAs
- » Preserves database availability and performance by resolving hangs
- » Preserves server availability by detecting and relieving memory stress
- » Discovers potential cluster and database problems, and notifies with corrective actions to prevent the issues altogether
- » Speeds issue diagnosis, triage and resolution for the problems that do occur

### Generates Diagnostic Metric View of Cluster and Databases

Oracle Autonomous Health Framework continuously monitors and stores metrics associated with Clusterware and operating system resources through its Cluster Health Monitor (CHM) component. CHM collects information in real-time that serves as a data feed for other Oracle Autonomous Health Framework components. It also helps system admins to analyze issues and identify its cause. When Grid Infrastructure (GI) is installed for RAC or RAC One Node database, Cluster Health Monitor is automatically enabled by default.

### Cluster Health Monitor Architecture

CHM has two services to collect diagnostic metrics – System Monitor Service (osysmond) and Cluster Logger Service (ologgerd) as shown in Figure 2. System monitor service is a real-time monitoring and operating system metric collection service that runs on each cluster node and is managed as a High Availability Services (HAS) resource. The collected metrics are then forwarded to cluster logger service that stores data in Oracle Grid Infrastructure Management Repository database.

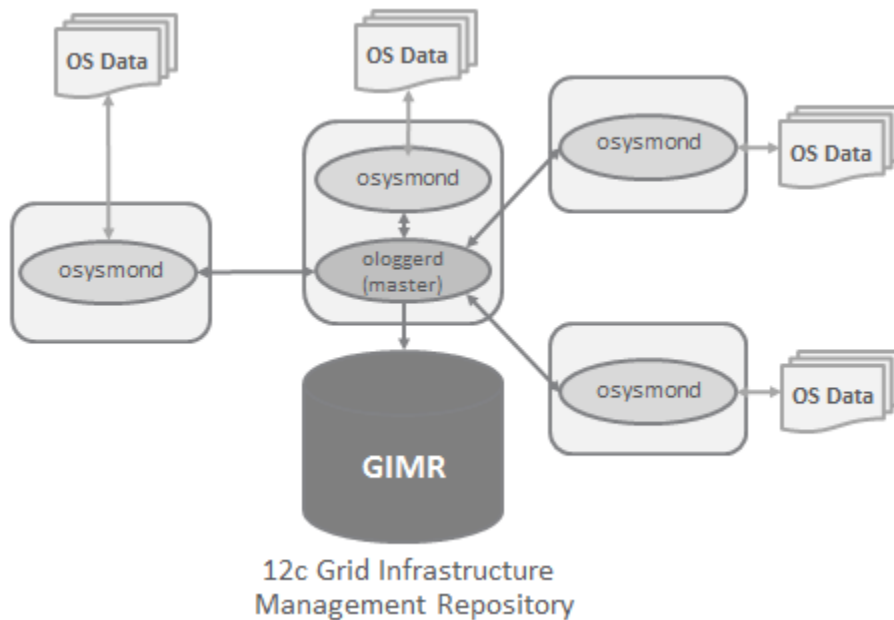


Figure 2: Architecture of Cluster Health Monitor

In a cluster, there is one cluster logger service per 32 nodes. Additional logger services are spawned for every additional 32 nodes. If logger service fails and is not able to come up after a fixed number of retries, all osysmond processes locally log and one respawns the ologgerd process.

#### Using Cluster Health Monitor to Collect Metrics

Cluster Health Monitor helps analyze issues and identify their cause by collecting the historic metric data including CPU utilization, memory utilization and total transfer rate as shown in Figure 3. This metric data from Cluster Health Monitor is available in graphical display within Enterprise Manager Cloud Control. Complete cluster views of this data are accessible from the cluster target page.



Figure 3: Metrics collected by Cluster Health Monitor for multiple nodes in cluster as seen in Enterprise Manager

Cluster Health Monitor also provides the historical review capability to examine trends to diagnose cross cluster issues that occur for example, over a weekend as shown in Figure 4.



Figure 4: Historical review of metrics collected by Cluster Health Monitor for multiple nodes in cluster as seen in Enterprise Manager



The metrics are broken down for further analysis as shown in Figure 5. For example, CPU utilization is broken down into CPU usage, CPU system usage and CPU user usage. In addition, CPU utilization metric can be drilled down to see CPU system usage, CPU user usage and CPU queue length.



Figure 5: CPU utilization metric broken down further into CPU usage, CPU system usage, and CPU user usage in Cluster Health Monitor

CHM by default monitors the top 127 processes to collect significant system metrics while keeping its resource consumption at acceptable levels. These processes include critical cluster processes, for example, crsd.bin, ocspd.bin, gipcd.bin, etc. CHM also allows user-specified critical processes to be monitored.

CHM supports plug-in collectors, for example, traceroute, netstat ping, etc. to provide enhanced network insight in 12.2. It listens to CSS and GIPC events where CSS and GIPC are protocols that involve node-to-node communication. CSS maintains membership for each node in the cluster. GIPC is used when blocks are moved between instances.

### Establishes Baseline and Maintains Best Practice Configurations

Configuration changes such as changes in a file or directory permissions during deployment lifecycle can cause a database outage. For example, incorrect permissions on the oracle.bin file can prevent session processes from being created. Such issues are detected by Oracle Autonomous Health Framework component, Cluster Verification Utility (CVU). When Oracle Grid Infrastructure (GI) is installed for RAC or RAC One Node database, CVU is automatically enabled by default.

### Cluster Verification Utility Architecture

Cluster Verification Utility daemon runs every 6 hours. to verify components including free disk space, memory, processes, and other Clusterware and database components. For each of these components, as shown in Figure 6, the checks/verifications to be performed are controlled through XML files. These files are processed to generate XML data which in turn generates a list of verification task Java objects which are processed by Verification engine. Finally, verification results and summary are displayed. CVU generates a baseline from the XML files, XML data about the pre-requisites and data on implicit Java tasks. This baseline is stored in a separate XML file to be available for future comparisons.

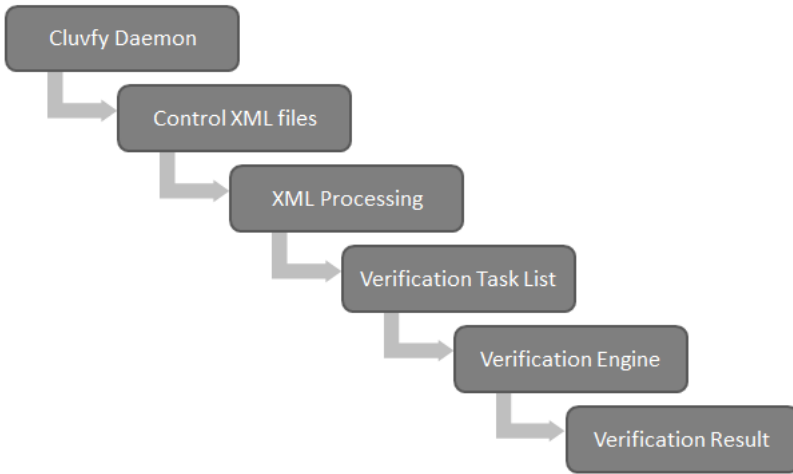


Figure 6: Cluster Verification Utility Architecture

### Using Cluster Verification Utility to Perform Health Checks

Cluster Verification Utility runs in daemon mode to maintain system health before and after any new installations, patches or upgrades. It allows administrators to establish a baseline for a healthy system, and performs checks against this baseline for O/S, Grid Infrastructure and Database compliance and best practices in the event of a configuration change. Users can access the results of CVU checks through its generated report in text or HTML file format. Figure 7 displays an example HTML report. These reports can be saved for later reference. CVU can be extended to include user-defined checks. Users can choose to run the CVU daemon for either the entire cluster or specific databases.

**Detailed report for Best Practices checks**

**Summary of environment**

Date (mm/dd/yyyy) 11/09/2016  
 Time (hh:mm:ss) 14:13:40  
 Cluster name cluster1  
 Clusterware version 12.2.0.1.0  
 Grid home /scratch/app/12.2/grid  
 Grid User grid  
 Operating system Linux3.8.13-68.3.4.el6uek.x86\_64

**Following components are checked as part of this report (Click on each component listed below to navigate)**

1. [System requirements](#)
2. [System recommendations](#)
3. [Clusterware requirements](#)
4. [Clusterware recommendations](#)

**System requirements**

Verification Check	Verification Result	Verification Description
Swap Size	WARNING	This is a prerequisite condition to test whether sufficient total swap space is available on the system.
Physical Memory	PASSED	This is a prerequisite condition to test whether the system has at least 8GB (8388608.0KB) of total physical memory.
Available Physical Memory	PASSED	This is a prerequisite condition to test whether the system has at least 50MB (51200.0KB) of available physical memory.
Free Space: node1 /usr,node1 /var,node1 /etc,node1 /sbin,node1 /tmp	PASSED	This is a prerequisite condition to test whether sufficient free space is available in the file system.
Free Space: node2 /usr,node2 /var,node2 /etc,node2 /sbin,node2 /tmp	PASSED	This is a prerequisite condition to test whether sufficient free space is available in the file system.

Figure 7: Cluster Verification Utility report

## Maintains Compliance with Best Practices and Alerts Vulnerabilities to Known Issues

DOS attacks, exploited vulnerabilities, software bugs, etc. can cause a database or instance to be unresponsive. Oracle Autonomous Health Framework component ORAchk is a lightweight and non-intrusive health check for Oracle stack of software and hardware components. It proactively scans database systems for known issues, analyzes them and recommends resolutions. When Oracle Grid Infrastructure (GI) is installed for RAC or RAC One Node database, ORAchk is automatically enabled by default.

### ORAchk Architecture

ORAchk works in three steps – Scheduling, Identification and Action. During scheduling, users set the frequency to run ORAchk’s data collection for a cluster’s nodes and databases. Users then start the ORAchk daemon. During its identification step, as shown in Figure 8, the ORAchk daemon:

- » Checks if version is out of date, if so either downloads or recommends download of latest version
- » Discovers all Oracle RAC stack components (both hardware and software) for servers within same database cluster
- » Executes health check scripts which compare node data against the baseline that ORAchk creates for healthy system
- » Compare results of health checks to best practice and generate compliance results

These compliance results are then sent to Collection Manager, when configured, where users can view them. Finally, during the Action step, ORAchk provides recommendations for resolving these issues within Collection Manager.

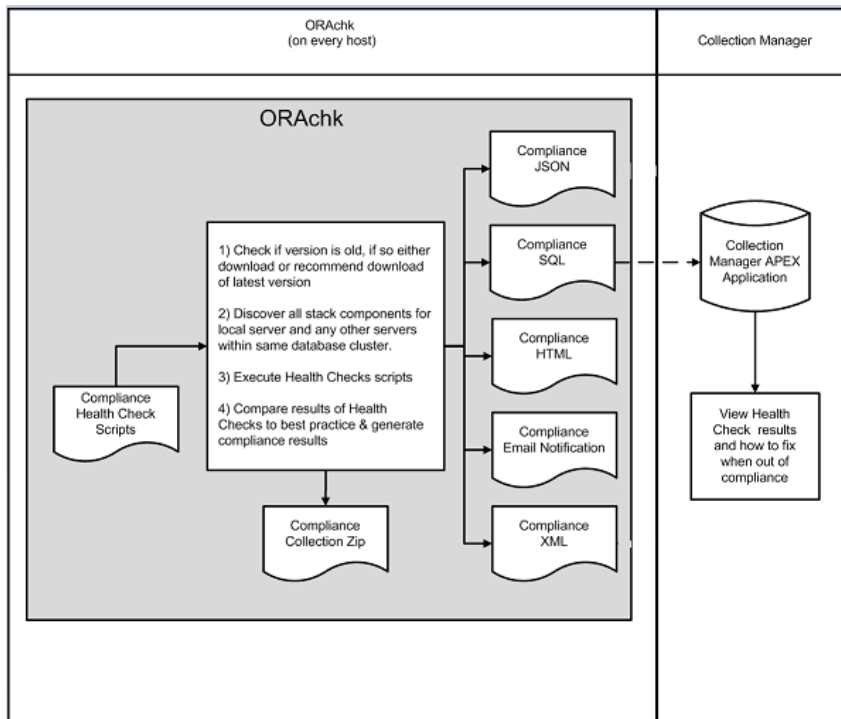


Figure 8: ORAchk Architecture

### Using ORAchk to Maintain Compliance

ORAchk stores the results of the checks it performs in files called collections and in the user-specified database configured to run its Apex-based application, Collection Manager. Collection Manager is sent the data by ORAchk and uses it to conveniently display health of entire database system and can be extended to multiple clusters as shown in Figure 9. Each bar on the cluster health chart denotes health of a cluster. The green section of the bar indicates healthy cluster checks, yellow indicates warnings, while red section indicates problems on the cluster.

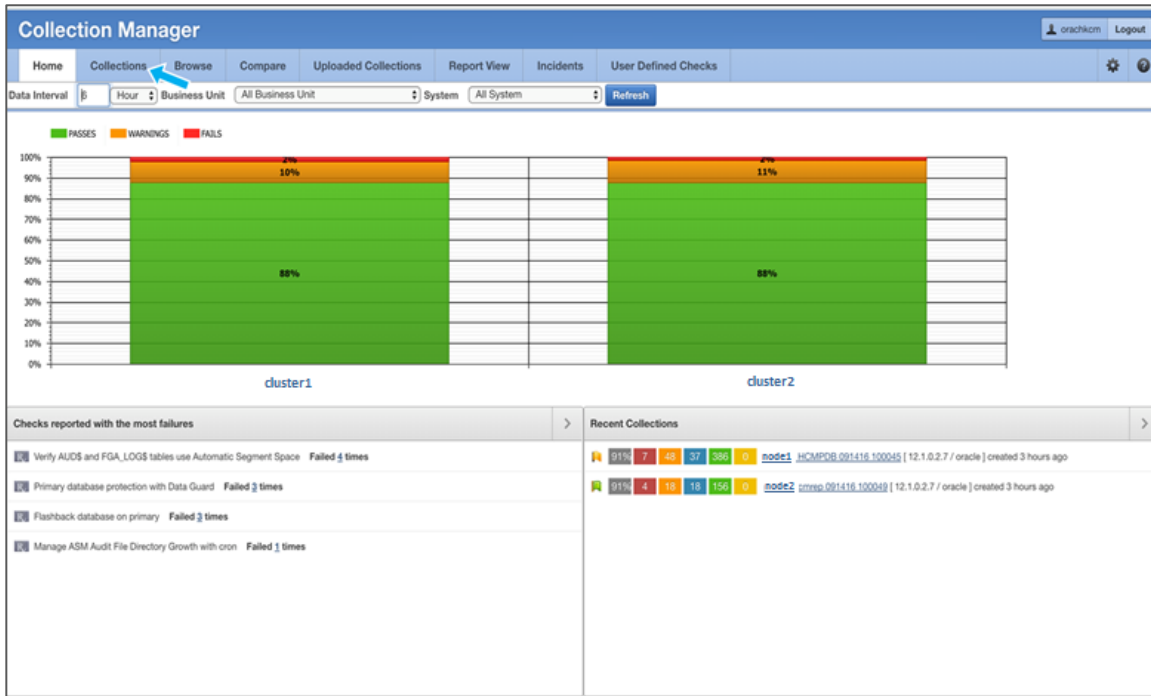


Figure 9: Collection Manager Dashboard

Collection Manager also allows users to dive deeper to assess health of individual clusters, to browse collections, to get details of individual collections including checks performed, status of the checks, etc., to compare two different collections as shown in Figure 10, to generate reports, and to add user-defined checks.

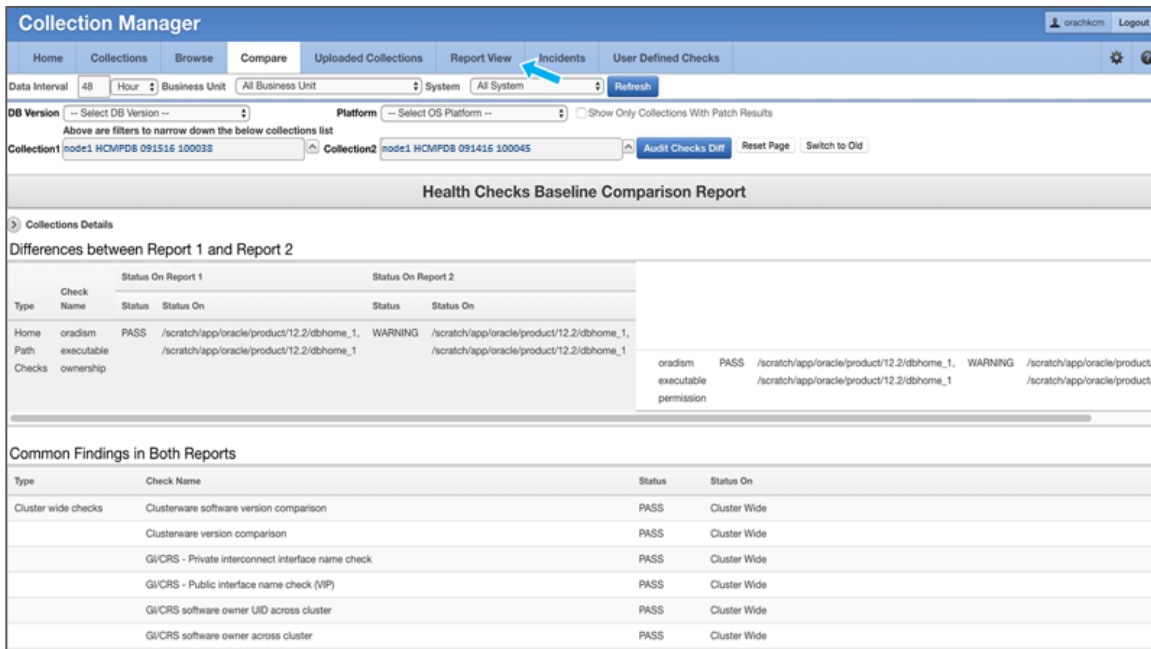


Figure 10: View of a collection in Collection Manager

## Autonomously Monitors Performance and Manages Resources to Meet SLAs

Oracle Autonomous Health Framework component Quality of Service Management (QoSM) addresses database server performance issues caused by bottlenecked resources. Quality of Service Management identifies these issues, generates notifications when they put SLAs at risk, and provides recommendations to manage resources to resolve issues and meet SLAs. QoSM allocates server resources where they are required the most based upon performance requirements in terms of performance objectives and business criticality rankings, in order to manage workloads to their service level agreements (SLAs).

Today, multiple and varied workloads are now being handled by a single server, each with their own set of performance objectives in terms of their response time. Some workloads may be highly critical from the business perspective and may need to be catered to more quickly than other workloads and therefore have a very low response time as their performance objective. Quality of Service Management provides a single dashboard to monitor and manage all workloads on the database system and helps to organize workloads just-in-time, based on their ranking, performance objectives and other criteria and allocates resources to them accordingly in order to optimize performance. When Grid Infrastructure (GI) is installed for RAC or RAC One Node database, Quality of Service Management is automatically ready to be enabled on a database-by-database basis.

### Quality of Service Management Architecture

Oracle Database QoS Management Server, as diagramed in Figure 11, retrieves database and OS metrics as well as topology from data sources including Oracle RAC and RAC One Node databases, Oracle Clusterware and Cluster Health Monitor. QoSM displays the results on a single dashboard in Enterprise Manager. These metrics include database request arrival rate, CPU use, CPU wait time, I/O use, I/O wait time, Global Cache use and Global Cache wait times from each database instance. The data is correlated by Performance Class every five seconds. Information about the current topology of cluster and health of servers is added to the data. The Policy and Performance Management engine of Oracle Database QoS Management analyzes the data to determine overall performance and resource profile of the system with regard to the current Performance Objectives established by the active Performance Policy.

The performance evaluation occurs once a minute and results in a recommendation and corresponding notification if any Performance Class does not meet its objectives. The recommendation specifies the target workload represented as a Performance Class, its bottlenecked resource and if possible specific corrective actions. The recommendation also includes its projected impact on all Performance Classes in the system.

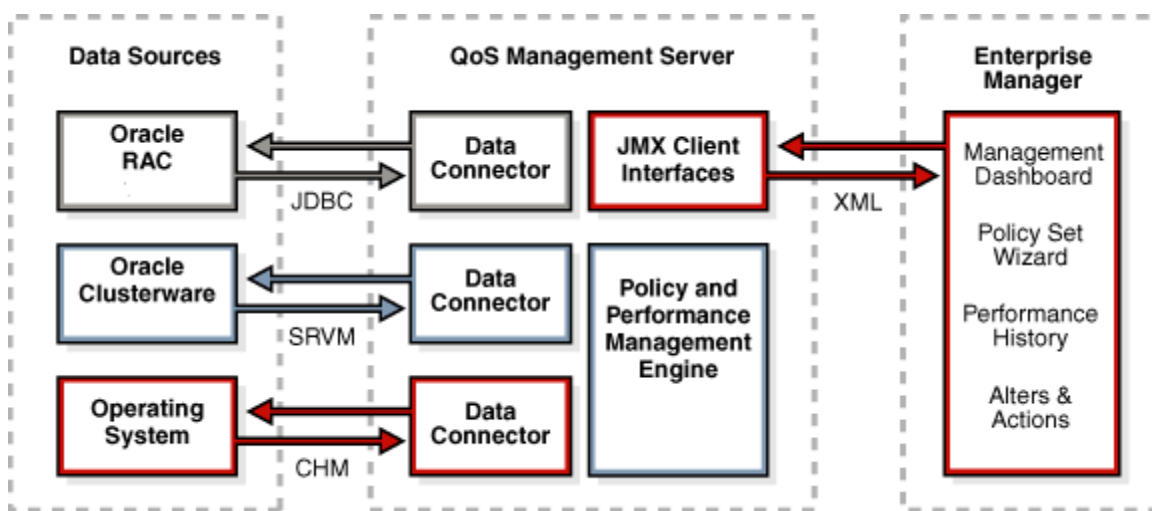


Figure 11: Quality of Service Management Architecture

## Using Quality of Service Management to Manage Resources and Maintain SLAs

Users can classify workloads through QoS into different performance classes by setting parameters and creating policies to filter workloads. QoS uses these policies for autonomous resource management to trade-off resources between competing workloads to maintain SLAs.

QoS can be used in three phases or in combination: Measurement phase, Monitoring phase and Management phase. In measurement phase, QoS helps to analyze current performance of workloads in terms of average response time categorized into resource usage time (blue bar) and resource wait time (grey bar) as shown in Figure 12. This helps to determine realistic performance objectives (in terms of average response time) for workloads.

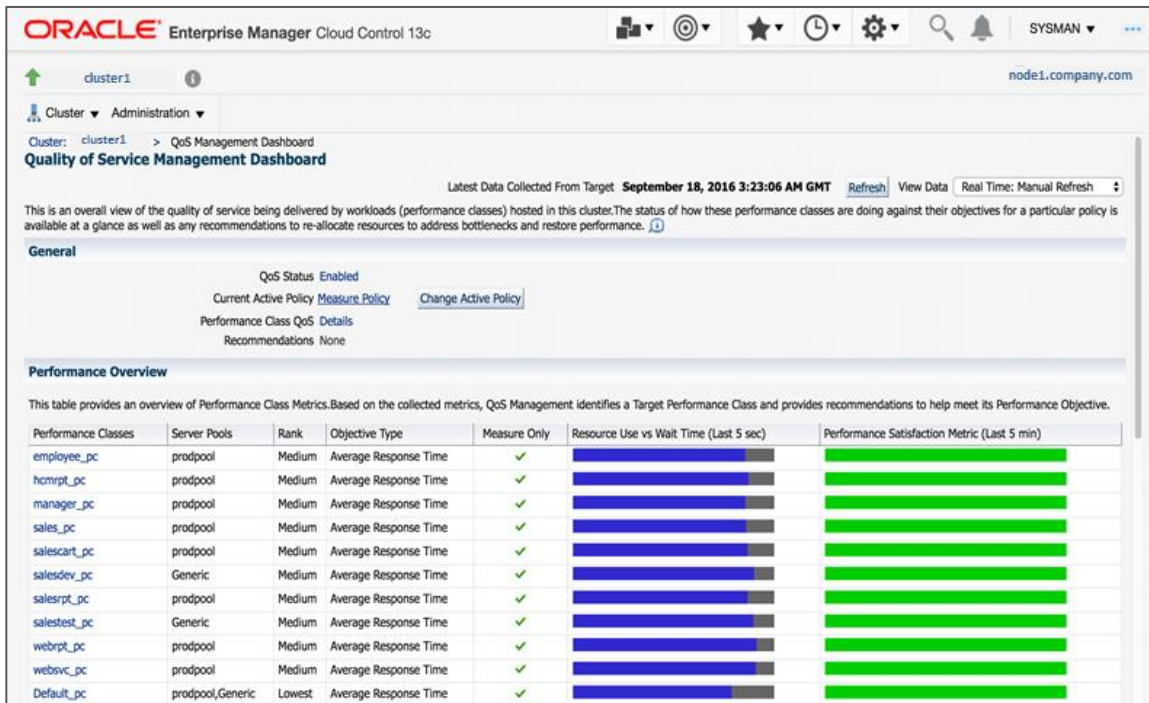


Figure 12: Quality of Service Management dashboard in the measurement phase

Quality of Service Management also identifies bottlenecked resources that degrade performance of a workload. QoS classifies resource wait time for a workload into CPU, I/O, Global cache and Other wait time as shown in Figure 13 where the highest values category of resource wait time is the bottlenecked resource.

For example, high CPU contention would cause high CPU wait time, high block contention would cause high Global Cache wait time, high I/O contention due to fewer disks would cause high I/O wait time and a SQL issue in latch or lock that could require an AWR report analysis would cause high Other wait time.



### Resource Wait Times Breakdown

This table provides breakdown of resource wait times by Performance Class. For each performance class, the bottlenecked resource is the Recommendations. The data can also be used to make manual adjustments to the system.

[Expand All](#) | [Collapse All](#)

Performance Class/Server Pool	CPU (sec)	Global Cache (sec)	IO (sec)	Other (sec)
xyzcluster				
salescart_pc	0.003557	0.000000	0.000000	0.000029
manager_pc	0.003518	0.000000	0.000000	0.000025
sales_pc	0.003596	0.000000	0.000002	0.000025
websvc_pc	0.002047	0.000000	0.000091	0.000032
employee_pc	0.003595	0.000000	0.000004	0.000031
hcmrpt_pc	0.004288	0.000000	0.000002	0.000025
salesrpt_pc	0.004066	0.000000	0.000000	0.000008
webrpt_pc	0.002024	0.000000	0.000021	0.000080
salesdev_pc	0.002528	0.000000	0.000000	0.000019
salestest_pc	0.002738	0.000000	0.000000	0.000047
Default_pc	0.000189	0.000000	0.000000	0.000242

Figure 13: Resource wait time breakdown by Quality of Service Management showing a high CPU contention in most of the workloads implying CPU as a bottlenecked resource

As shown in Figure 14, Quality of Service Management also provides a historical view of workload performance in terms of resource use time, resource wait time, demand, etc. for further analysis to identify causes of problems like fluctuations or sudden surge in the workload performance.



Figure 14: Quality of Service Management display of the performance history of the workloads

By default, workloads are classified based on service names. However, in monitoring phase, users can set additional parameters to classify workloads more granularly and set performance objectives and priority ranking for workloads through performance policy. QoS uses this policy to compare current workload performance with set performance objectives. If performance objectives are violated, additional workload resource wait time is represented by red bar under the Resource Use vs Wait Time column as shown in Figure 15. If performance objectives are met, extra headroom available is represented by green bar. QoS displays workload performance relative to its performance objective for last 5 mins under Performance Satisfaction Metric column. The red bar represents the amount of time its response time exceeds a performance class exceeds its performance objective. QoS also allows users to set the threshold time within EMCC's notification framework to receive warnings or alert notifications due to performance classes continuously violating their objectives.

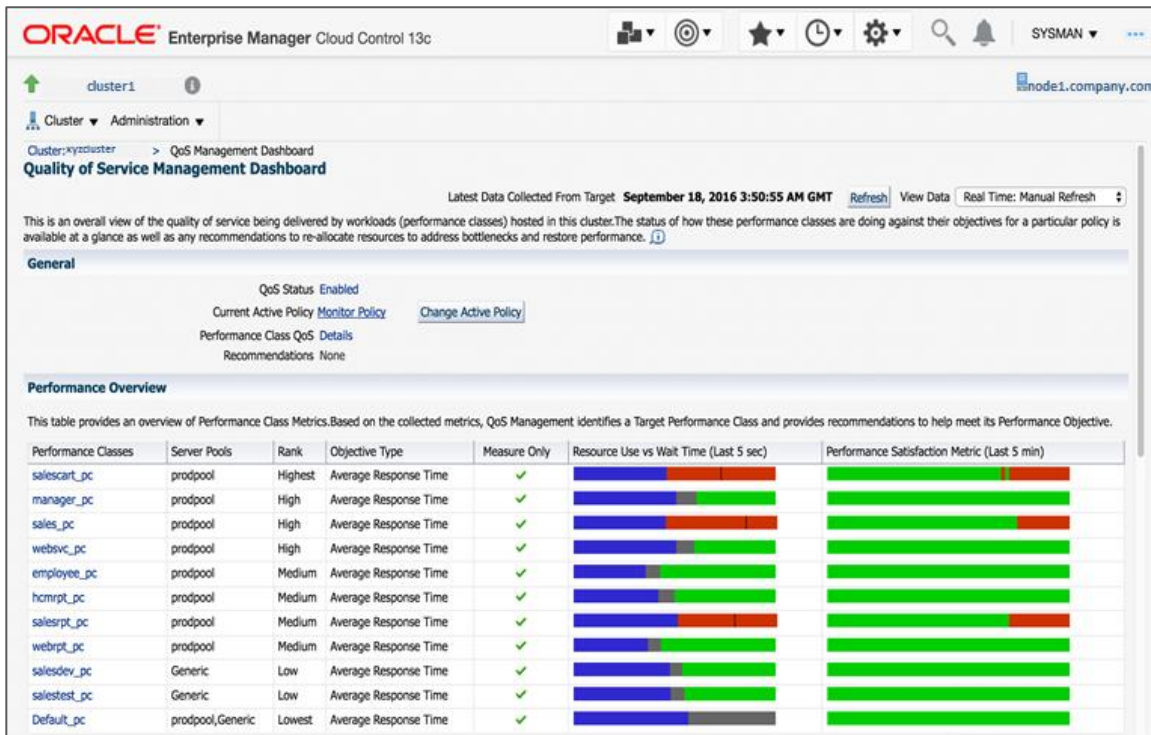


Figure 15: Quality of Service Management dashboard in the monitoring phase

In management phase, users can set a new policy to actively manage workloads. In this phase, the user defines server pool resource parameters along with performance objectives and ranking for workloads. Based on this policy, QoS recommends resource reallocation to fulfill performance objectives for business critical workloads and optimize performance for other workloads as shown in Figure 16. Note that QoS manages reallocation of CPU resources only to manage to workload SLAs.



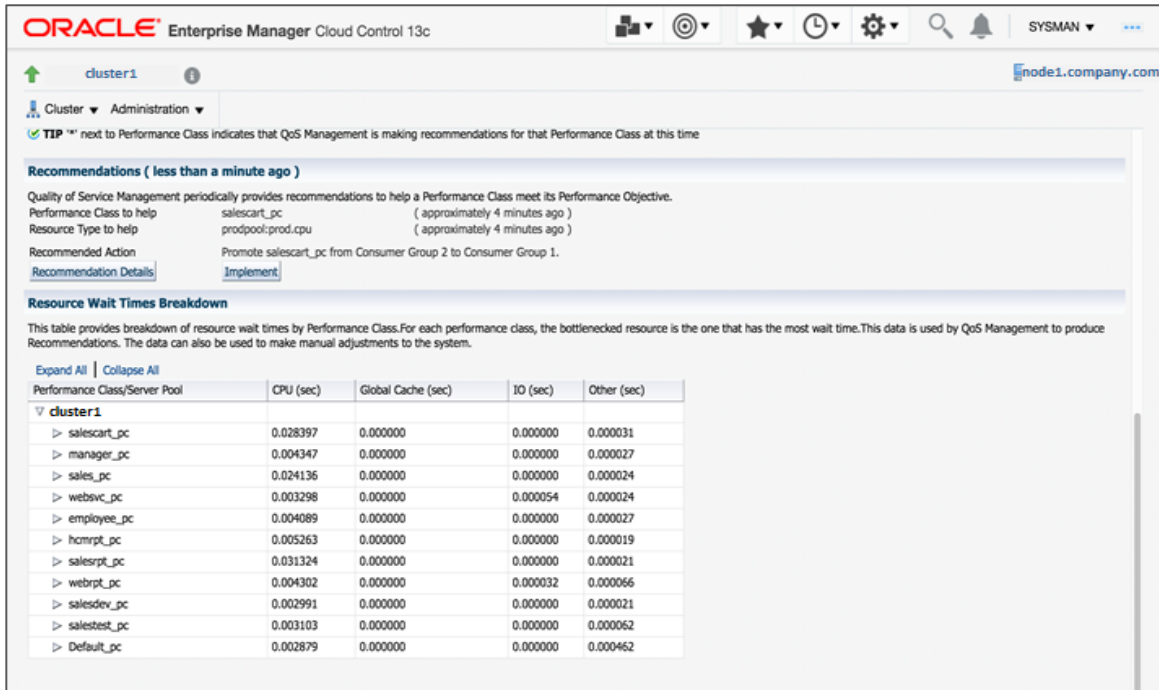


Figure 16: Quality of Service Management dashboard presenting recommendations in the Management phase

Through these three phases – measurement, monitoring and management, Quality of Service Management provides a continuous workload health view through a single cluster-wide real-time dashboard. It also helps to identify bottleneck resources, analyse the performance history of the workloads, and manage the resources with its targeted bottleneck resolution recommendations to meet SLAs.

## Autonomously Preserves Database Availability and Performance During Hangs

Database hangs occur when a chain of one or more sessions is blocked by another session and is not able to make any progress. These can make databases unresponsive to applications by denying critical database resources in locks, latches, and CPU to other sessions. Oracle Autonomous Health Framework component Hang Manager autonomously detects and resolves these hangs. Hang Manager is enabled when RAC or RAC One Node databases are created.

### Hang Manager Architecture

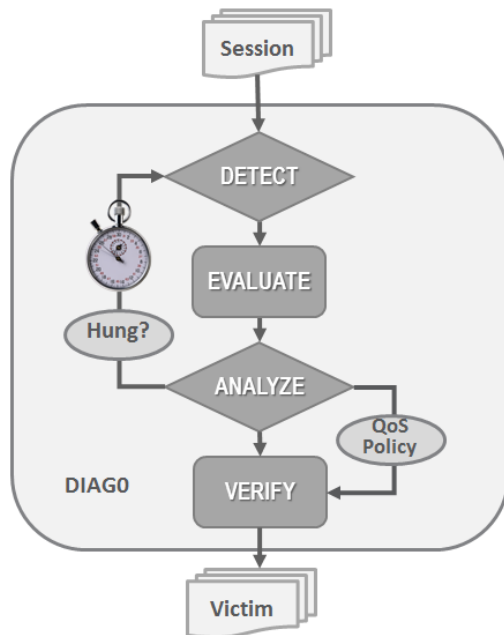


Figure 17: Hang Manager Architecture

Hang Manager autonomously runs as a `DIA0` background process within Oracle databases as shown in Figure 17. Hang Manager has three phases – Detect, Analyze and Verify. In its Detect phase, Hang Manager collects data on all the nodes from Cluster Health Monitor. It detects sessions waiting for resources held by another session for some time and monitors them. Hang Manager then analyzes these sessions in its Analyze phase to determine if they are part of potential hang. If so, Hang Manager waits to ensure that sessions are truly hung. After a set time, Hang Manager verifies these sessions as hangs in its Verify phase and selects a final blocker session as victim session. It applies hang resolution heuristics to victim session. In case the hang does not resolve, it terminates victim session and if that fails, Hang Manager terminates the session process.

### Using Hang Manager to Resolve Hangs

Hang Manager by default has its sensitivity parameter set to Normal and trace file size set to a default value. Users can change these parameters if required. For example, for faster hang resolution the sensitivity parameter can be set to High.

While resolving hangs, Hang Manager also considers the active Quality of Service Management policy. For example, if a hang includes a session associated with a highly ranked critical Performance Class in the QoSM policy, Hang Manager expedites the termination of victim session to maintain performance objectives of the critical session.

Hang Manager detects and resolves hangs autonomously. However, it continuously logs all detections and resolutions in DB Alert Logs as shown below.

```

2015-10-13T16:47:59.435039+17:00
Errors in file /oracle/log/diag/rdbms/hm6/hm6/trace/hm6_dia0_12433.trc (incident=7353):
ORA-32701: Possible hangs up to hang ID=1 detected
Incident details in: .../diag/rdbms/hm6/hm6/incident/incdir_7353/hm6_dia0_12433_i7353.trc
2015-10-13T16:47:59.506775+17:00
DIA0 requesting termination of session sid:40 with serial # 43179 (ospid:13031) on instance 2
due to a GLOBAL, HIGH confidence hang with ID=1.
Hang Resolution Reason: Automatic hang resolution was performed to free a
significant number of affected sessions.
DIA0: Examine the alert log on instance 2 for session termination status of hang with ID=1.
In the alert log on the instance local to the session (instance 2 in this case),
we see the following:
2015-10-13T16:47:59.538673+17:00
Errors in file .../diag/rdbms/hm6/hm62/trace/hm62_dia0_12656.trc (incident=5753):
ORA-32701: Possible hangs up to hang ID=1 detected
Incident details in: .../diag/rdbms/hm6/hm62/incident/incdir_5753/hm62_dia0_12656_i5753.trc
2015-10-13T16:48:04.222661+17:00
DIA0 terminating blocker (ospid: 13031 sid: 40 ser#: 43179) of hang with ID = 1
requested by master DIA0 process on instance 1
Hang Resolution Reason: Automatic hang resolution was performed to free a
significant number of affected sessions.
by terminating session sid:40 with serial # 43179 (ospid:13031)

```

The details of complete hang resolution are also available in dump trace files for later reference and troubleshooting.

## Autonomously Preserves Server Availability By Relieving Memory Stress

Enterprise database servers can use all available free memory due to too many open sessions or runaway workloads causing node eviction. This event where free memory falls below a safe threshold is called memory stress. Oracle Autonomous Health Framework component Memory Guard autonomously monitors nodes for memory stress and relieves it in order to prevent node eviction and maintain server availability. When Grid Infrastructure (GI) is installed for RAC or RAC One Node database, Memory Guard is automatically enabled by default.

### Memory Guard Architecture

Memory Guard as shown in Figure 18 runs as an MBean daemon in a J2EE container managed by Cluster Ready Services (CRS). Memory Guard is hosted on the qosmserver singleton resource that runs on any cluster node for high availability. Cluster Health Monitor sends a metrics stream to Memory Guard providing real-time memory resources information for cluster nodes including amount of available memory and amount of memory currently in use. Memory Guard also collects cluster topology from Oracle Clusterware. It uses cluster topology and memory metrics to identify database nodes that have memory stress.

Memory Guard then stops database services managed by Oracle Clusterware on the stressed node transactionally. It relieves memory stress without affecting already running sessions and their associated transactions. After completion, memory used by these processes starts freeing up and adding to pool of available memory on the node. When Memory Guard detects that amount of available memory is more than threshold, it restarts services on the affected node.

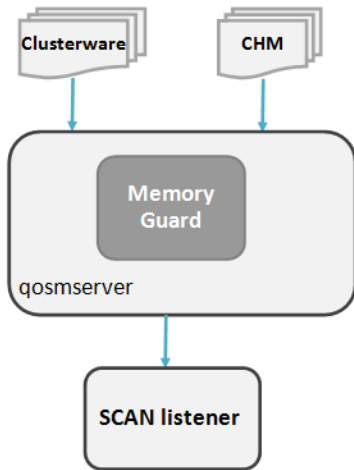


Figure 18: Memory Guard Architecture

While a service is stopped on a stressed node, new connections for that service are redirected by the listener to other nodes providing the same service for non-singleton database instances. However, for policy-managed databases, the last instance of a service is never stopped in order to maintain availability.

### Using Memory Guard to Relieve Memory Stress

Memory Guard autonomously detects and monitors Oracle Real Application Clusters (Oracle RAC) or Oracle RAC One Node databases when they are open. Memory Guard sends alert notifications when it detects memory stress on a database node. Memory Guard alerts can be found in audit logs under \$ORACLE\_BASE/crsdata/node name/qos/logs/dbwlm/auditing.

Memory Guard log file when the services are stopped due to memory stress is as shown below:

```

<MESSAGE>
<HEADER>
<TSTZ_ORIGINATING>2016-07-28T16:11:03.701Z</TSTZ_ORIGINATING>
<COMPONENT_ID>wlm</COMPONENT_ID>
<MSG_TYPE TYPE="NOTIFICATION"></MSG_TYPE>
<MSG_LEVEL>1</MSG_LEVEL>
<HOST_ID>hostABC</HOST_ID>
<HOST_NWADDR>11.111.1.111</HOST_NWADDR>
<MODULE_ID>gomlogger</MODULE_ID>
<THREAD_ID>26</THREAD_ID>
<USER_ID>userABC</USER_ID>
<SUPPL_ATTRS>
<ATTR NAME="DBWLM_OPERATION_USER_ID">userABC</ATTR>
<ATTR NAME="DBWLM_THREAD_NAME">MPA Task Thread 1469722257648</ATTR>
</SUPPL_ATTRS>
</HEADER>
<PAYLOAD>
<MSG_TEXT>Server Pool Generic has violation risk level RED.</MSG_TEXT>
</PAYLOAD>
</MESSAGE>
<MESSAGE>
<HEADER>
<TSTZ_ORIGINATING>2016-07-28T16:11:03.701Z</TSTZ_ORIGINATING>
<COMPONENT_ID>wlm</COMPONENT_ID>
<MSG_TYPE TYPE="NOTIFICATION"></MSG_TYPE>
<MSG_LEVEL>1</MSG_LEVEL>
<HOST_ID>hostABC</HOST_ID>
<HOST_NWADDR>11.111.1.111</HOST_NWADDR>
<MODULE_ID>gomlogger</MODULE_ID>
<THREAD_ID>26</THREAD_ID>
<USER_ID>userABC</USER_ID>
<SUPPL_ATTRS>
<ATTR NAME="DBWLM_OPERATION_USER_ID">userABC</ATTR>
<ATTR NAME="DBWLM_THREAD_NAME">MPA Task Thread 1469722257648</ATTR>
</SUPPL_ATTRS>
</HEADER>
<PAYLOAD>
MSG_TEXT>Server userABC-hostABC-0 has violation risk level RED. New connection requests will no longer be accepted.</MSG_TEXT>
</PAYLOAD>
</MESSAGE>
  
```

Memory Guard log file when the services were restarted after relieving the memory stress is as shown below:

```
<MESSAGE>
...
<MSG_TEXT>Memory pressure in Server Pool Generic has returned to normal.</MSG_TEXT>
...
<MSG_TEXT>Memory pressure in server userABC-hostABC-0 has returned to normal. New connection requests are now accepted.</MSG_TEXT>
...
</MESSAGE>
```

### Discovers Potential Cluster & Database Problems - Notifies with Corrective Actions

Oracle Autonomous Health Framework component Cluster Health Advisor (CHA) provides system and database administrators with early warning of pending performance issues, and root causes and corrective actions for Oracle RAC databases and cluster nodes. Oracle Cluster Health Advisor then performs anomaly detection for each input based on the difference between observed and expected values. If sufficient inputs associated with a specific problem are abnormal, then Oracle Cluster Health Advisor raises a warning and generates an immediate targeted diagnosis and corrective action.

Oracle Cluster Health Advisor stores the analysis results, along with diagnosis information, corrective action, and metric evidence for later triage, in the Grid Infrastructure Management Repository (GIMR). Oracle Cluster Health Advisor also sends warning messages to Enterprise Manager Cloud Control using the Oracle Clusterware event notification protocol.

When Grid Infrastructure (GI) is installed for RAC or RAC One Node database, Cluster Health Advisor is automatically enabled by default.

### Cluster Health Advisor Architecture

As shown in Figure 19, Oracle Cluster Health Advisor runs as a highly available cluster resource, ochad, on each node in the cluster. Each Oracle Cluster Health Advisor daemon (ochad) monitors the operating system on the cluster node and optionally, each Oracle Real Application Clusters (Oracle RAC) database instance on the node.

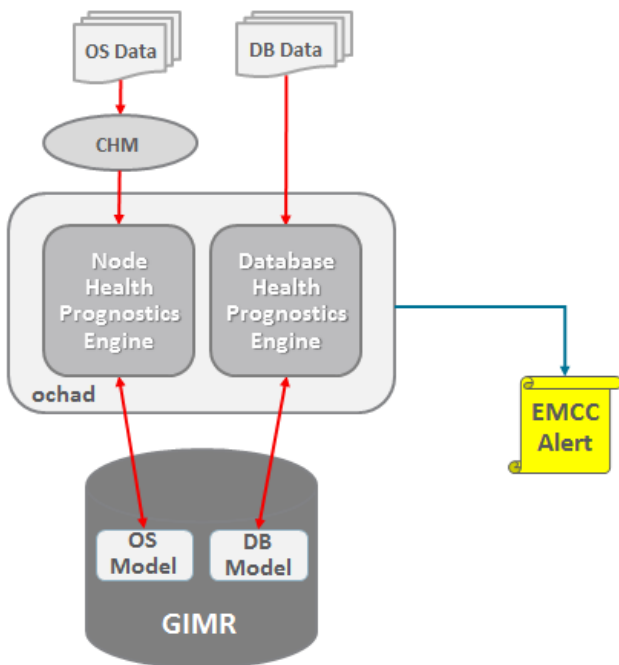



Figure 19: Flow diagram for Cluster Health Advisor architecture



The ochad daemon receives OS metric data from the Cluster Health Monitor and gets Oracle RAC database instance metrics from a memory-mapped file. The daemon does not require a connection to each database instance. This data, along with the selected model, is used in the Health Prognostics Engine of Oracle Cluster Health Advisor for both the node and each monitored database instance in order to analyze their health multiple times a minute.

The results of this analysis along with any diagnosis and corrective action are stored in Grid Infrastructure Management repository (GIMR) along with its metric evidence for later triage. CHA accesses stored data through Oracle Enterprise Manager Cloud Control (EMCC) or cluster terminal through CHACTL.

### Using Cluster Health Advisor for Prognosis of Potential Threats

By default, Cluster Health Advisor models are designed to be conservative to prevent false warning notifications. However, default configuration may not be sensitive enough for critical production systems. Therefore, Cluster Health Advisor provides an onsite model calibration capability to use actual production workload data to form the basis of its default setting and increase accuracy and sensitivity of node and database models. Since workloads may vary on specific cluster nodes and Oracle RAC databases, Cluster Health Advisor also provides the capability to create, store, and activate multiple models with their own specific calibration data. This functionality is also managed by CHACTL. Sample problems detected by CHA along with their corrective actions using CHACTL query diagnosis are as shown:

Problem: DB Control File IO Performance

Description: CHA has detected that reads or writes to the control files are slower than expected.

Cause: The Cluster Health Advisor (CHA) detected that reads or writes to the control files were slow because of an increase in disk IO.

The slow control file reads and writes may have an impact on checkpoint and Log Writer (LGWR) performance.

Action: Separate the control files from other database files and move them to faster disks or Solid State Devices.

Problem: DB CPU Utilization

Description: CHA detected larger than expected CPU utilization for this database.

Cause: The Cluster Health Advisor (CHA) detected an increase in database CPU utilization because of an increase in the database workload.

Action: Identify the CPU intensive queries by using the Automatic Diagnostic and Defect Manager (ADDM) and follow the recommendations given there. Limit the number of CPU intensive queries or relocate sessions to less busy machines. Add CPUs if the CPU capacity is insufficient to support the load without a performance degradation or effects on other databases.

When OCHAD detects an Oracle RAC or Oracle RAC One Node database instance running, it autonomously starts monitoring cluster nodes. However, to monitor Oracle RAC database instances, Oracle Grid Infrastructure user is required to use CHACTL to explicitly turn on monitoring for each database.

## Speeds Issue Diagnosis, Triage and Resolution

**While Oracle Autonomous Health Framework components - ORAchk, Cluster Verification Utility, Quality of Service Management and Cluster Health Advisor autonomously identify issues and recommend solutions, some issues require Oracle Support Services. For example, network failures on the private interconnect which can immediately result in evicted nodes should a cable be pulled or a NIC fail. While such issues cannot be detected early, Oracle Autonomous Health Framework component Trace File Analyzer (TFA) helps to collect and extract relevant information in a timely manner across multiple nodes involved for issue analysis by Oracle Support Services. This is especially useful when data is frequently lost or overwritten as diagnostic collections may not happen until some time after the issue occurred. TFA's daemon mode is enabled by default when Grid Infrastructure (GI) is installed for RAC or RAC One Node database.**

### Trace File Analyzer Architecture

As shown in Figure 20, when running in daemon mode, TFA Collector monitors Oracle logs for events symptomatic of a significant problem as step 1. In step 2, based on the event type detected, TFA then starts an automatic diagnostic collection. The data collected

depends on event detected. TFA coordinates collection cluster-wide, trims the logs around relevant time periods, and then packs all collection results into a single package on one node.

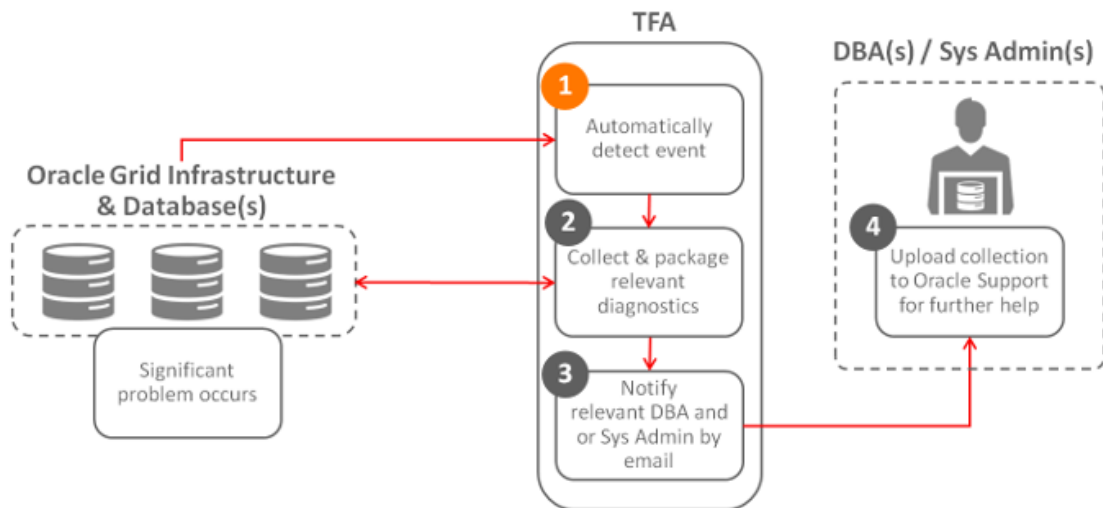


Figure 20: Trace File Analyzer Architecture

TFA does not do a collection for every event detected. When an event is first identified, it triggers a start point for a collection and then waits for five minutes before starting diagnostic gathering in order to capture any other relevant events.

- » If events are still occurring after 5 minutes, then TFA waits to complete diagnostic collection for up to a further five minutes for 30 seconds with no events occurring.
- » If events are still occurring 10 minutes after first detection, then TFA forces a diagnostic collection and generates a new collection start point for the next event.

Once the collection is complete, TFA sends email notification that includes the details of where the collection results are, to the relevant recipients as step 3. And finally, in step 4, the recipients can then upload the collections to Oracle Support Services for further help.

### Using Trace File Analyzer to Collect Relevant Information for an Issue

TFA collects data from nodes based on parameters set, for example, collecting data related to a specific incident time or time range, complete files that have relevant data, or just a time slice of data from those files. Exactly what is collected is dependent on the string and in which alert it is found but potentially trimmed versions of all O/S, CRS, ASM and RDBMS logs can be collected for each string. This type of operation could be called many times a second if not controlled so there can never be a collection generated at less than 5 minute intervals due to Trace File Analyzer implementation of flood control. Furthermore, if Trace File Analyzer repository reaches its repository max size then no auto collection takes place. After data is collected, Trace File Analyzer, then, generates reports based on that data as shown in Figure 21.

NFO: analyzing all (Alert and Unix System Logs) logs for the last 1440 minutes... Please wait...  
 NFO: analyzing host: myHost1

```

    Report title: Analysis of Alert, System Logs
    Report date range: last ~1 day(s)
    Report (default) time zone: PST - Pacific Standard Time
    Analysis started at: 25-Sep-2014 01:52:01 PM PDT
    Elapsed analysis time: 1 second(s).
    Configuration file: /u01/tfa/myHost1/tfa_home/ext/tnt/conf/tnt.prop
    Configuration group: all
    Total message count: 4,539, from 30-Jul-2014 11:59:39 PM PDT to 25-Sep-2014 01:51:54 PM PDT
    Messages matching last ~1 day(s): 258, from 24-Sep-2014 01:54:53 PM PDT to 25-Sep-2014 01:51:54 PM PDT
    last ~1 day(s) generic count: 258, from 24-Sep-2014 01:54:53 PM PDT to 25-Sep-2014 01:51:54 PM PDT
    last ~1 day(s) ignored generic count: 0
    last ~1 day(s) unique generic count: 39

message types for last ~1 day(s)
-----
Occurrences percent server name type
-----
258 100.0% myHost1 generic
-----
258 100.0%

Unique generic messages for last ~1 day(s)
-----
Occurrences percent server name generic
-----
216 83.7% myHost1 myHost1 init: Id "co" respawning too fast: disabled for 5 minutes
3 1.2% myHost1 myHost1 CLSD: The clock on host myHost1 has been updated by the Cluster Time
synchronization Service to be synchronous with the mean cluster time.
3 1.2% myHost1 [OCTSSD(12358)]CRS-2408: The clock on host myHost1 has been updated by the Cluster
Time Synchronization Service to be synchronous with the mean cluster time.
1 0.4% myHost1 Thread 2 advanced to log sequence 232 (LGWR switch)
Current log# 4 seq# 232 mem# 0:
DATA12102/CDB12102/ONLINELOG/group_4.270.854340625
1 0.4% myHost1 myHost1 kernel: cgrep[7720]: segfault at 0 ip 000000004e7365cc sp 00000000ff8bfe3c
error 4 in libc-2.5.so[4e6df000+152000]
1 0.4% myHost1 Thread 2 advanced to log sequence 235 (LGWR switch)
Current log# 3 seq# 235 mem# 0:
DATA12102/CDB12102/ONLINELOG/group_3.269.854340625
1 0.4% myHost1 [CLSECHO(2332)]CRS-10001: 24-Sep-14 14:40 AFD-9204: false
1 0.4% myHost1 [CLSECHO(27286)]CRS-10001: 24-Sep-14 20:40 AFD-9204: false
1 0.4% myHost1 TABLE SYS.WRPS_REPORTS: ADDED INTERVAL PARTITION SYS_PB16 (1729) VALUES LESS THAN
TO_DATE(' 2014-09-26 01:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLE SYS.WRPS_REPORTS_DETAILS: ADDED INTERVAL PARTITION SYS_PB17 (1729) VALUES
LESS THAN (TO_DATE(' 2014-09-26 01:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))

```

Figure 21: Sample Trace File Analyzer files for an issue

Thus, TFA collects only the relevant information regarding an issue based on time specified or event occurrence that triggered the issue. This information is packaged in a compact manner which can be easily sent to Oracle Support Services to resolve the issue quickly.

### Oracle Autonomous Health Framework in Oracle Cluster Domain

Oracle AHF generates and stores a lot of diagnostic data while diagnosing and resolving availability and performance issues in the database system. A 4 node cluster on an average generates 6-7GB of diagnostic data for retention of 3 days. This would create overhead by consuming local resources. Furthermore, Oracle AHF components interact and use data generated by each other. This becomes convenient if the entire data is stored at one place instead of in local repositories of each component.

Oracle Cluster Domain supports four types of clusters:

- » A Standalone Cluster (formerly Flex Cluster)
- » Two types of Member (formerly “Client”) Clusters:
  - » Application Member Cluster
  - » Database Member Cluster
- » Domain Services Cluster



A member cluster, here, is a cluster that is managed in Cluster Domain, in which all clusters are registered with a common Management Repository Service. It can use different services that are offered as Cluster Domain Services through Oracle Domain Services Cluster (DSC). The components of Oracle AHF are also provided as services to all the member clusters of Oracle Cluster domain through the centralized Domain Services Cluster (DSC) as shown in Figure 22. For example, ORAchk component is provided as ORAchk collection service, Rapid Home Provisioning component is provided as Rapid Home Provisioning Service.

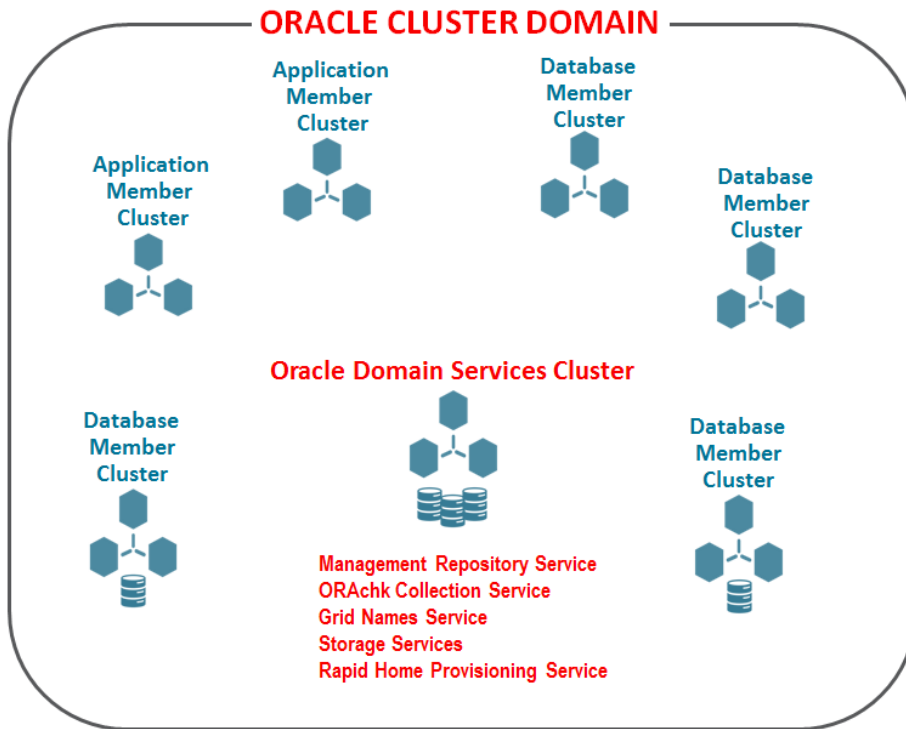


Figure 22: Oracle Cluster Domain

Oracle AHF is therefore supported in Oracle Cluster Domain where the overhead of storing diagnostic data of Oracle AHF is offloaded to the centralized infrastructure repository – Grid Infrastructure Management Repository (GIMR). GIMR is available to all Oracle RAC users for free. Thus, the centralization of the Oracle AHF in DSC makes it easy to manage, easily accessible to all the member clusters and also helps to reduce the local footprint of Oracle AHF.

## Conclusion

With globalization of businesses, database systems need to be available and perform consistently at all times in order that customers may perform transactions 24x7. Any daily operational issues that threaten availability and performance of such database systems, therefore, need to be addressed quickly.

Oracle Autonomous Health Framework is a solution that helps to prevent and resolve these issues. Its components work together to identify situations which are potential threats to the database system and provides corrective actions to resolve them. For issues that do occur, Oracle AHF helps to resolve them quickly with minimal effort by identifying the issue, diagnosing its cause and providing resolutions. For issues that require Oracle Support Service (OSS), Oracle AHF also collects relevant information required by OSS for quickly resolving the issue. Oracle AHF; therefore, provides a solution at every step – prevent issues before they occurs, resolve issues when they occur and expedites resolution of issues that require OSS assistance. This makes Oracle AHF a complete solution to maintain availability and manage performance of Oracle database systems.







**Oracle Corporation, World Headquarters**

500 Oracle Parkway  
Redwood Shores, CA 94065, USA

**Worldwide Inquiries**

Phone: +1.650.506.7000  
Fax: +1.650.506.7200

CONNECT WITH US

-  [blogs.oracle.com/oracle](http://blogs.oracle.com/oracle)
-  [facebook.com/oracle](http://facebook.com/oracle)
-  [twitter.com/oracle](http://twitter.com/oracle)
-  [oracle.com](http://oracle.com)

**Integrated Cloud Applications & Platform Services**

Copyright © 2017, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0116

Oracle Autonomous Health Framework  
March 2017  
Author: Ankita Khandelwal

