ORACLE

Oracle Autonomous Health Framework

Oracle Technical White Paper

September 2021, Version 21.3

Purpose statement

This document provides an overview of features and enhancements included in Oracle Autonomous Health Framework release 21c. It is intended solely to help you assess the business benefits of upgrading to 21c and plan your IT projects.

Disclaimer

This document in any form, software, or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Table of Contents

Introduction

Businesses today are global. They have customers across the world using their applications and performing transactions 24x7. Databases power applications and provide relevant data to other applications through various database services. Therefore, to give customers a continuous and consistent application experience, businesses need to ensure that their underlying databases run smoothly 24x7. Furthermore, databases not only need continuous availability but also need to provide consistent performance. Therefore, any issues affecting this availability and performance need to be addressed and resolved quickly to bring these databases back online.

Currently, human reaction time prevents timely problem resolution due to delays in identification and diagnosis. This delay can prove to be costly by adversely affecting ongoing business transactions and user experience.

Oracle Autonomous Health Framework (AHF) presents the next generation of tools, powered by applied machine learning technologies in 21c, as components that autonomously work 24x7 to keep database systems healthy and running while minimizing human reaction time. Oracle AHF components include Cluster Health Monitor, Cluster Verification Utility, ORAchk/EXAchk, Quality of Service Management, Hang Manager, Memory Guard, Cluster Health Advisor, and Trace File Analyzer, which have been integrated into a service model in 21c as shown in Figure 1.

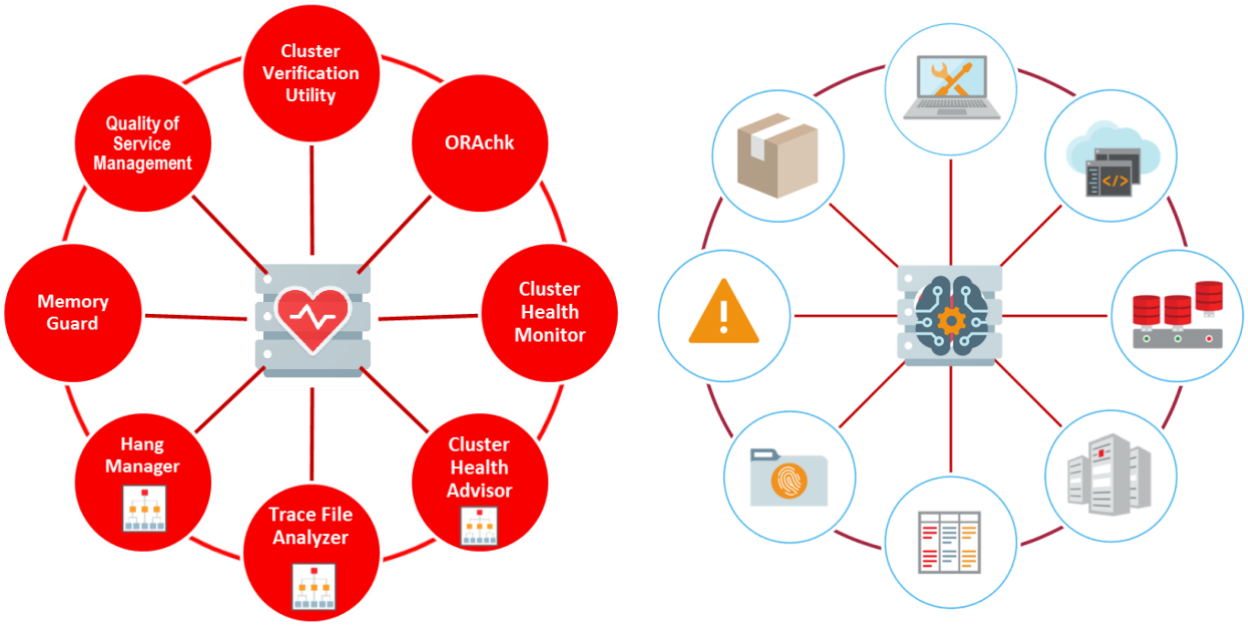


Figure 1: Oracle AHF with its applied machine learning components – AHF 21c with its integrated service model

Oracle AHF provides an early warning or automatically solves operational runtime issues faced by database and system administrators in the areas of availability and performance.

New Features in Oracle Database 21c Oracle Autonomous Health Framework

In Oracle Database 21c, Oracle AHF uses applied machine learning technologies to support the diagnosis of a broader range of operational runtime issues, provide resolutions, and provide intelligent log analysis of the issues. It has also becomes better integrated into a service model that is easier to configure and manage with these new features:

- Oracle AHF auto-updating to automatically and securely install the latest release from a staging area ensuring the latest release is available.
- Oracle ORAchk and EXAchk now show only the most critical checks by default bring their reports into critical focus.
- Oracle ORAchk and EXAchk reporting on Autonomous Health Certification to ensure proper configuration and status of the AHF components.
- Oracle Cluster Health Advisor new optimized Exadata Database and Node models improve issue detection on Oracle's Engineered Database Systems.
- Oracle Quality of Service Management now auto-enabled for measuring database workload performance providing detailed response time and bottleneck reporting for database services.
- Oracle Grid Infrastructure Management Repository Service for centralized AHF metrics management eliminating its production cluster footprint while maintaining full AHF client functionality.
- Oracle Grid Infrastructure Management Repository in separate database home for ease of upgrades.

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 by autonomously monitoring and managing the hardware resources and the database stack.

While many 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 the 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 are 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 stress 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 node eviction.
- Network issues, for example, network congestion on private interconnect caused by a change in configuration, 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 impossible to anticipate: For example, network failures on private interconnect due to a network card failure or cable pull resulting node eviction.

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 can deny critical database resources in locks, latches, CPU to other sessions. These 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. These 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 the system's performance 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 the 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 caused by deviations from best practices in configuration;
- Bottlenecked resource issues 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;
- Known bugs resolved by 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 on time;
- Degraded performance due to misconfigured parameters in SGA versus PGA allocation, number of sessions/processes, CPU counts, etc., based upon the type of workload and level of concurrency required;
- Client demand exceeds server or database capacity.

Thus, Oracle Autonomous Health Framework addresses many operational runtime issues in availability and performance for the database system's hardware and software resources.

How Does Oracle Autonomous Health Framework Address These Issues?

Oracle Autonomous Health Framework components utilize applied machine learning technologies and work 24x7 in daemon mode to address availability and performance issues and ensure high availability and consistent performance for the database system. In addition, they collaborate 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 their cause. Installing the Oracle Grid Infrastructure (GI) enables Cluster Health Monitor by default across each cluster node.

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 running on each cluster node and managed as a High Availability Services (HAS) resource. The collected metrics are then forwarded to the cluster logger service that stores data in the Oracle Grid Infrastructure Management Repository database. If the GIMR is not installed either locally in the cluster or a centralized location such as a Doman Services Cluster, collected metrics will only be stored locally on the file system. These files are available in a variety of formats including CSV and JSON.

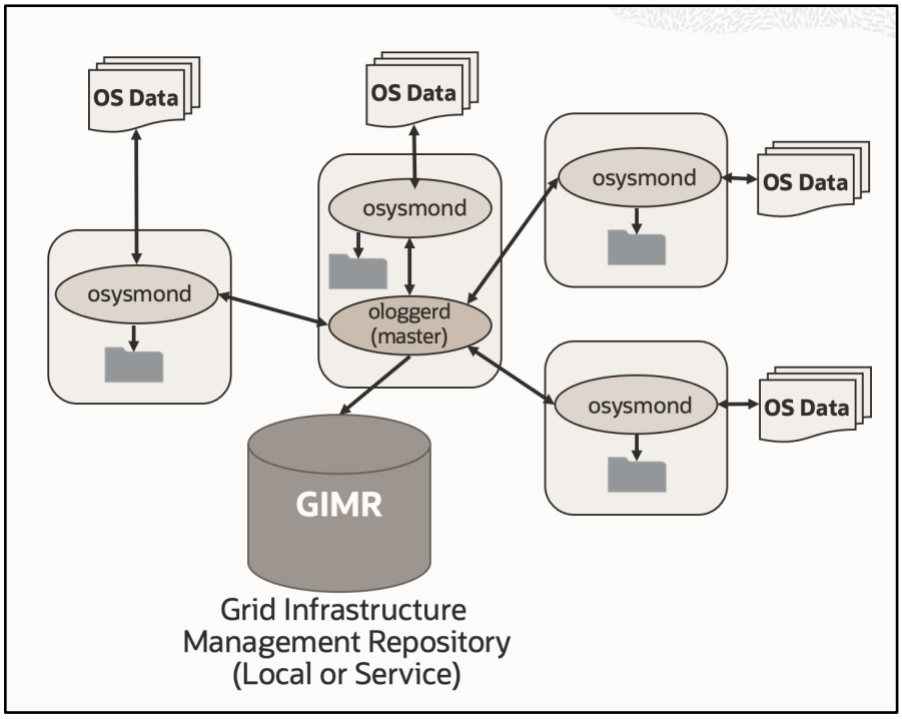


Figure 2: Architecture of Cluster Health Monitor

There is an initial cluster logger service for up to the first 32 nodes in a cluster with an additional logger service for each subsequent 32 nodes. If logger service fails and cannot 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 historical metric data, including CPU utilization, memory utilization, and total transfer rate, as shown in Figure 3. This metric data from Cluster Health Monitor via the

GIMR is available in the graphical display within Enterprise Manager Cloud Control. In addition, complete cluster views of this data are accessible from the cluster target page.

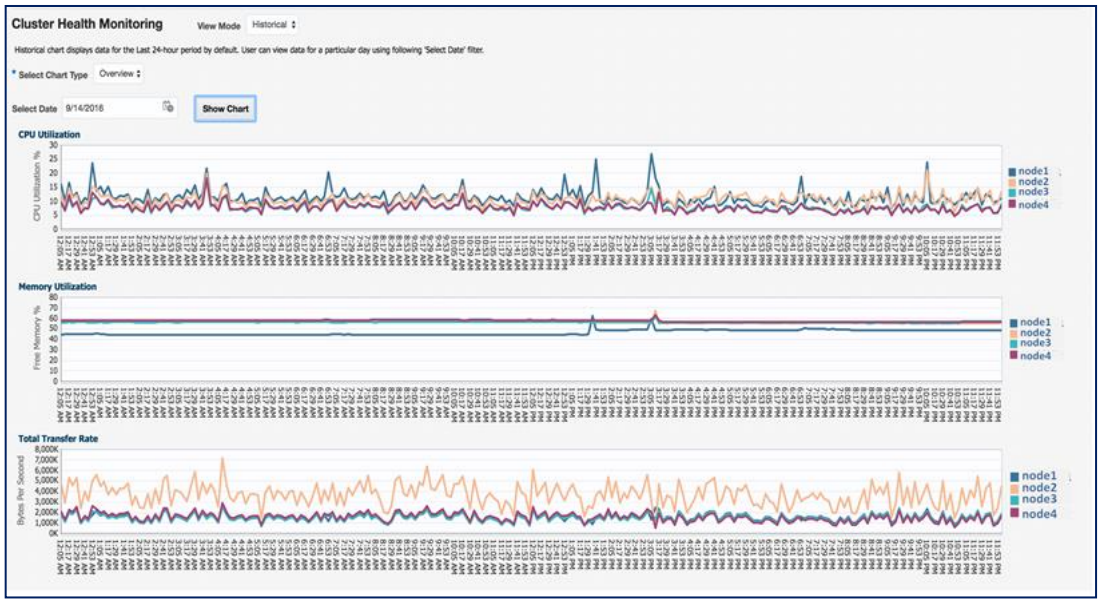


Figure 3: History of metrics collected by Cluster Health Monitor as seen in Enterprise Manager

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 the cluster as seen in Enterprise Manager

These metrics are broken down for further analysis, as shown below in Figure 5. For example, Admins can see CPU utilization factored into CPU usage, CPU system usage, and CPU user usage, as well as CPU utilization metrics into 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 essential processes, for example, crsd, cssd, etc. CHM also allows the monitoring of critical user-specified processes.

CHM supports plug-in collectors, for example, traceroute, netstat ping, etc., to provide enhanced network insight. 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 transports blocks between instances.

Establishes Baseline and Maintains Best Practice Configurations

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

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. As shown in Figure 6, the XML files control the checks/verifications performed for each of these components. These files are processed to generate XML data, which generates a list of verification task Java objects processed by the Verification engine. Finally, verification results and summary are displayed. In addition, CVU generates baseline components from the XML files, XML data about the pre-requisites, and data on implicit Java tasks. A separate XML file stores this baseline component.

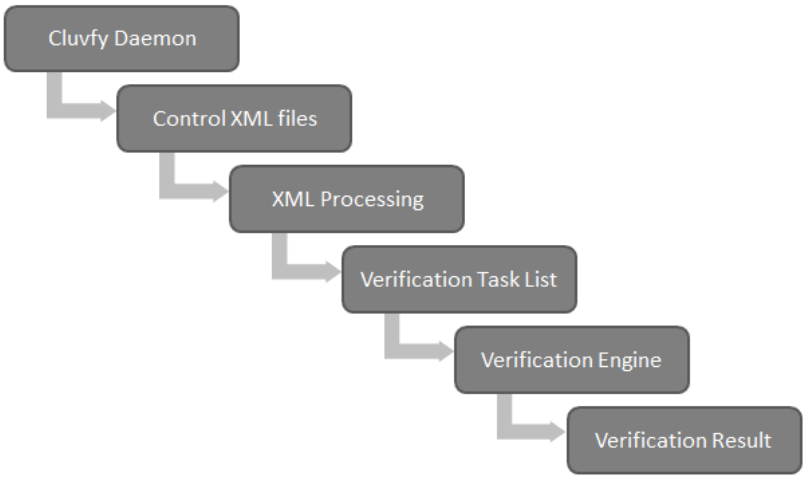
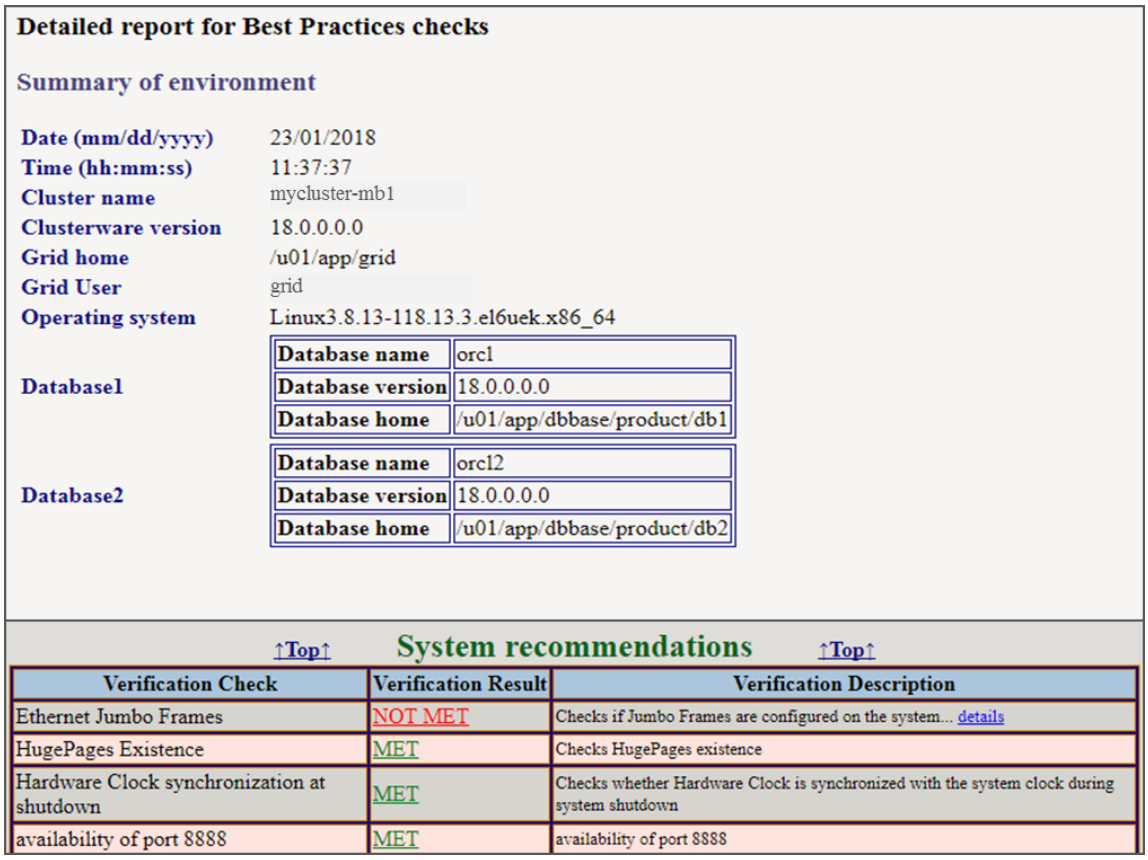


Figure 6: Cluster Verification Utility Architecture

Using Cluster Verification Utility to Perform Health Checks

Cluster Verification Utility runs in daemon mode to maintain system health before and after new installations, patches, or upgrades. It allows administrators to establish a baseline for a healthy system. Then, it performs checks against this baseline for O/S, Grid Infrastructure, and Database compliance and best practices in the event of a configuration change. Admins can access the results of CVU checks through its generated report in text or HTML file format. Figure 7 displays an example HTML report. They also can extend CVU 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)	23/01/2018
Time (hh:mm:ss)	11:37:37
Cluster name	mycluster-mb1
Clusterware version	18.0.0.0.0
Grid home	/u01/app/grid
Grid User	grid
Operating system	Linux3.8.13-118.13.3.el6uek.x86_64

Database1	**Database name**	orcl
	Database version	18.0.0.0.0
	Database home	/u01/app/dbbase/product/db1
Database2	**Database name**	orcl2
	Database version	18.0.0.0.0
	Database home	/u01/app/dbbase/product/db2

↑Top↑ **System recommendations** ↑Top↑

Verification Check	Verification Result	Verification Description
Ethernet Jumbo Frames	NOT MET	Checks if Jumbo Frames are configured on the system... details
HugePages Existence	MET	Checks HugePages existence
Hardware Clock synchronization at shutdown	MET	Checks whether Hardware Clock is synchronized with the system clock during system shutdown
availability of port 8888	MET	availability of port 8888

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/EXAchk is a lightweight and non-intrusive health check for the Oracle stack of software and hardware components. It proactively scans database systems for known issues, analyzes them, and recommends resolutions. Installing the Oracle Grid Infrastructure (GI) for RAC or RAC One Node database enables ORAchk/EXAchk by default.

In 21c, ORAchk/EXAchk is rewritten to focus on performance and extensibility, resulting in a 3x speed improvement and smaller resource footprint.

ORAchk/EXAchk Architecture

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

- Checks if the version is out of date; if so, either downloads or recommends downloading the latest version
- Discovers all Oracle RAC stack components (both hardware and software) for servers within the same database cluster
- Executes health check scripts that compare node data against the baseline that ORAchk creates for a 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.

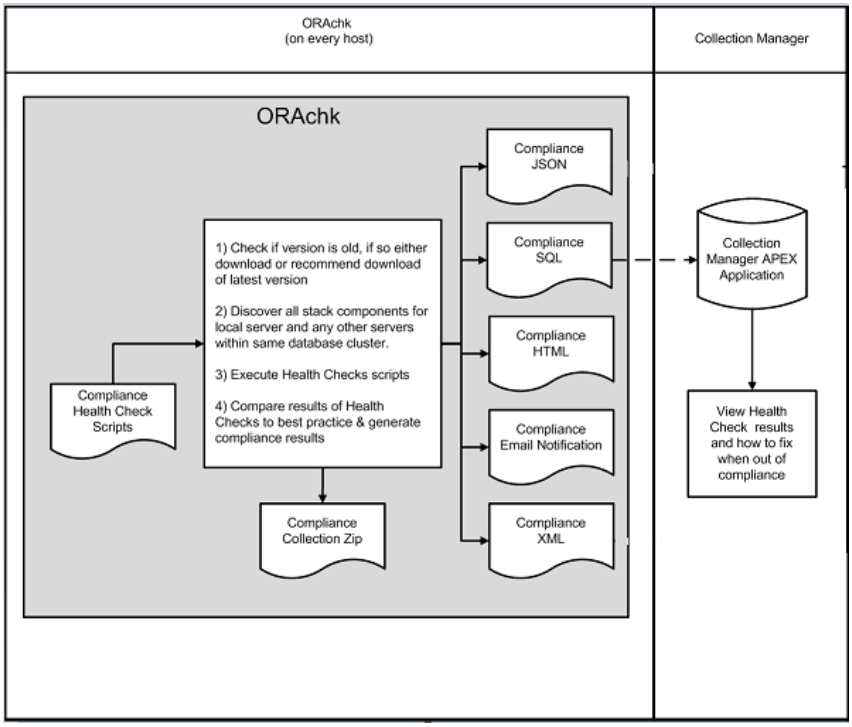


Figure 8: ORAchk/EXAchk Architecture

Using ORAchk to Maintain Compliance

ORAchk/EXAchk 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/EXAchk and uses it to display the entire database system's health conveniently and can be extended to multiple clusters, as shown in Figure 9. Each bar on the cluster health chart denotes the health of a cluster. The green section of the bar indicates healthy cluster checks, blue indicates informational results, yellow indicates warnings, while the red section indicates problems on the cluster.

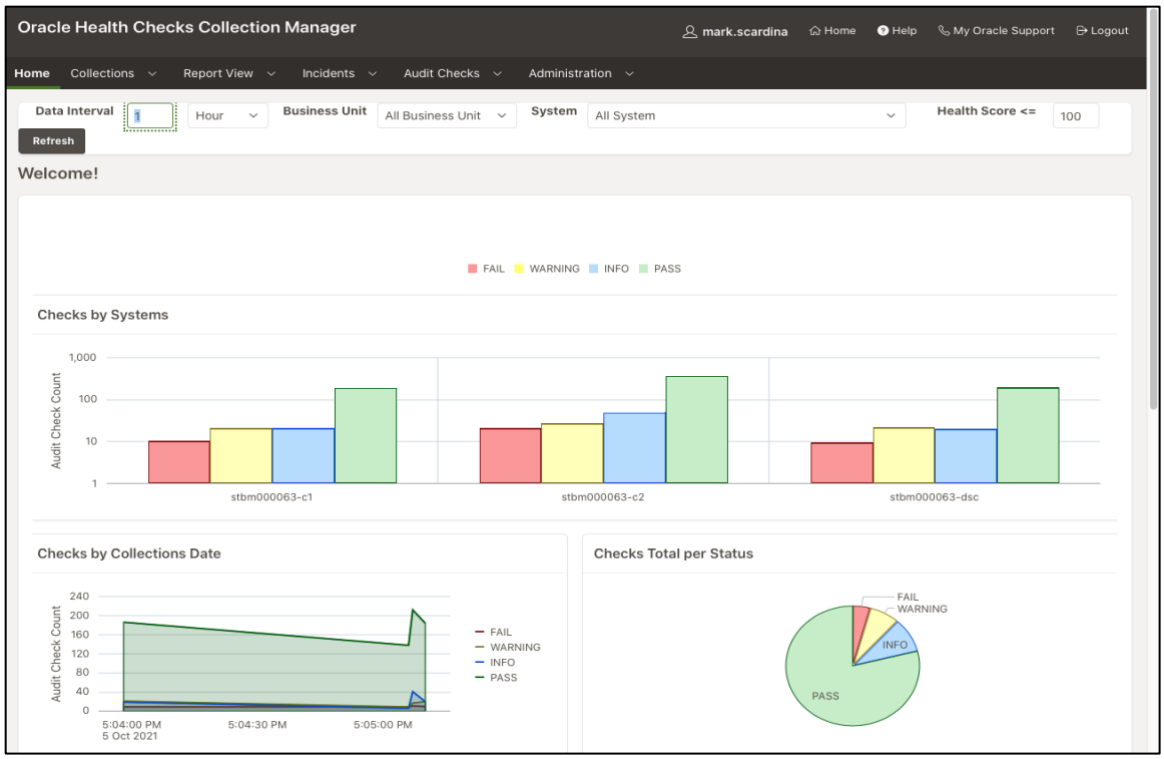


Figure 9: Collection Manager Dashboard

Collection Manager also allows users to compare audit check results of two different collections based on Business Unit, System, DB Version, and Platform. Using Collection Manager comparison, users can also check the best practices incorporated during an upgrade/patch. For example, Figure 10 below shows how the system failed certain best practices checks performed by ORAchk before the upgrade in the 1st collection. However, in the 2nd collection after the upgrade, the system passed these best practices checks indicating the upgrade already incorporated the best practices.

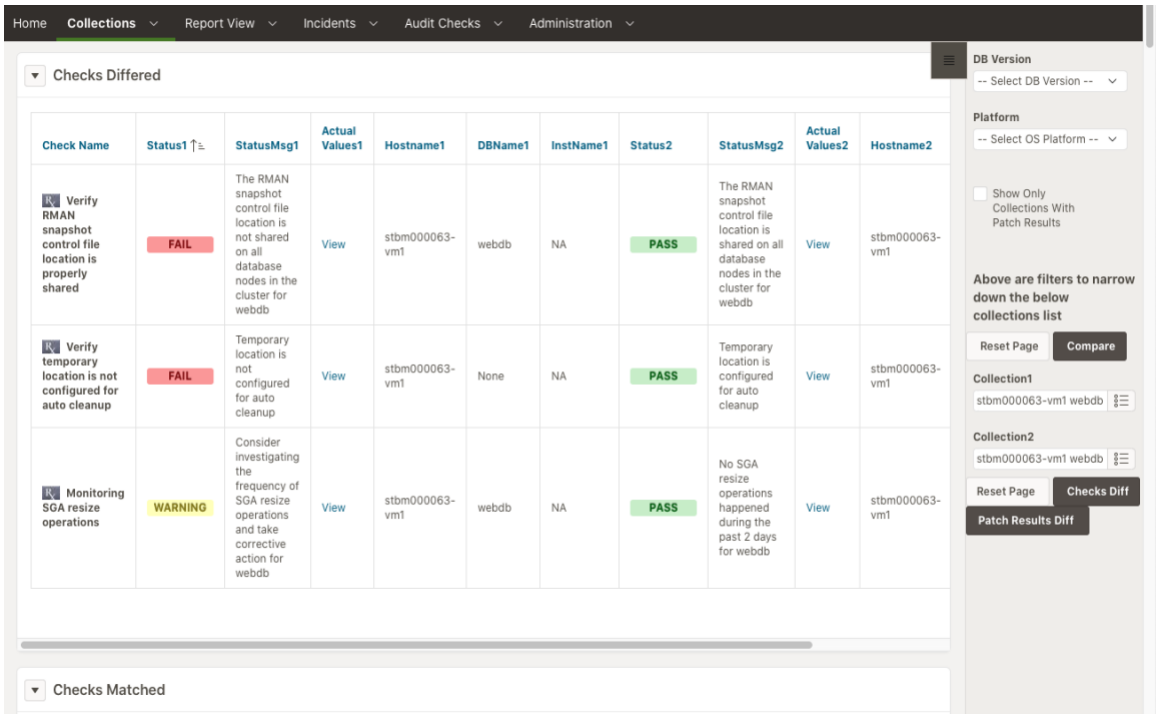


Figure 10: Comparison of collections in Collection Manager

These comparisons are advantageous in situations such as upgrades to identify any issues that may have occurred during the upgrade. For example, as shown below in Figure 11, a comparison of collections just before and after the upgrade in Collection Manager shows that one of the checks that had passed previously failed after the upgrade due to improper usage of a hidden database initialization parameter.

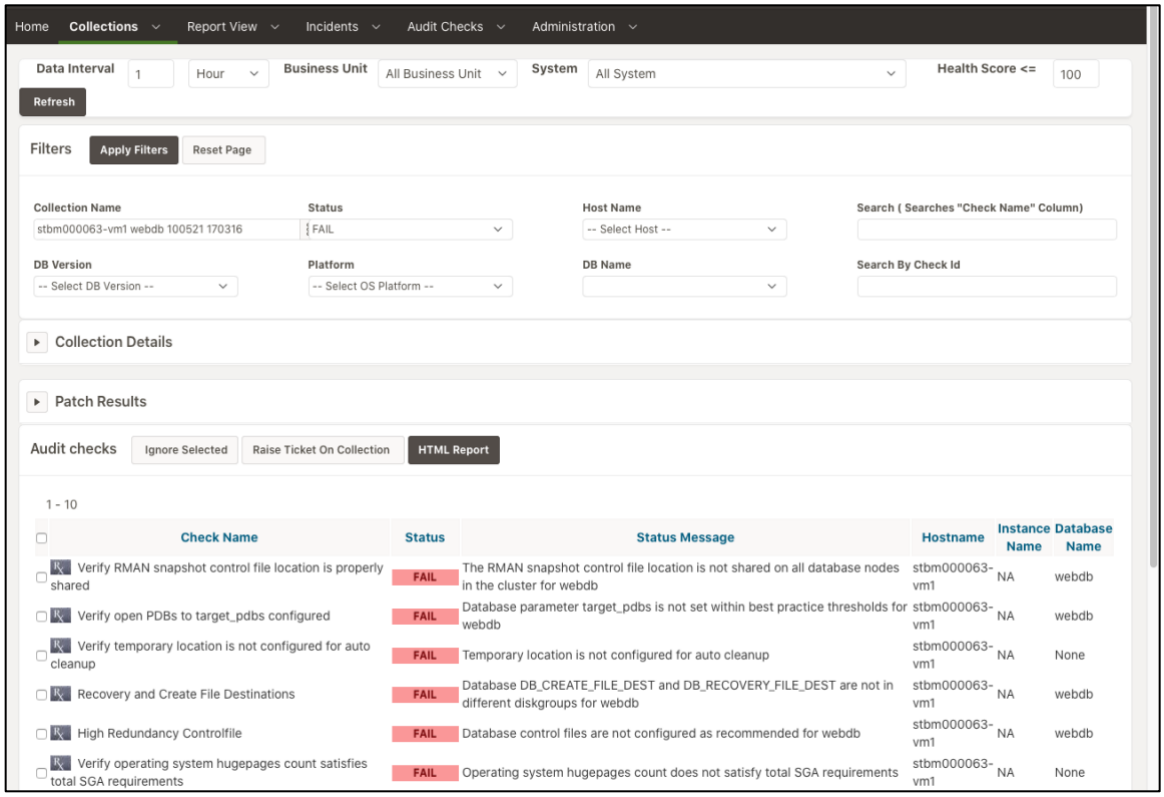


Figure 11: Identifying Critical Issues to be addressed

Using Collection Manager, users can identify the issues and get a detailed root cause analysis of the issue and the corrective action to resolve the issue. Figure 12 below shows the root cause analysis and corrective action for the issue identified during comparison in Figure 11. The failed check shows a hidden database initialization parameter setting as a workaround for a specific problem in the previous version. However, the upgrade already contained the fix for the issue, and therefore, no longer requiring the workaround parameter. Collection Manager further provides the list of actions to take to correct the issue.

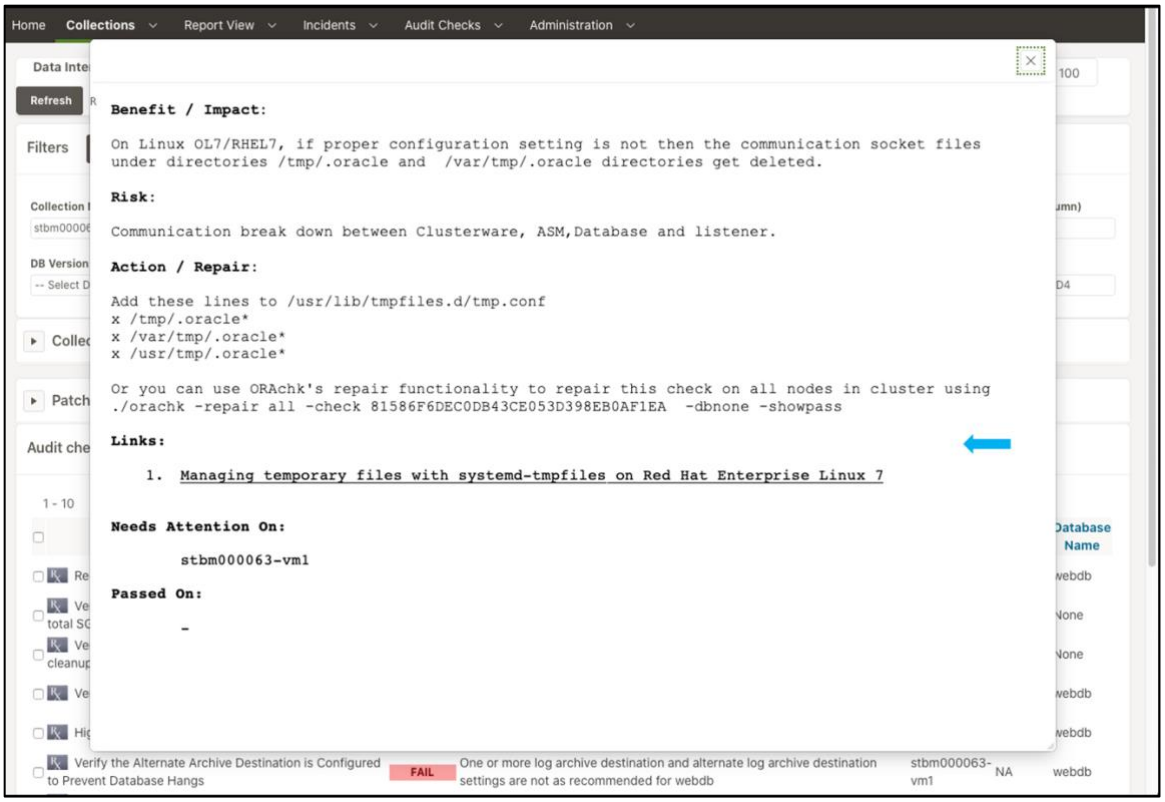


Figure 12: Root Cause Analysis and Corrective Action by Collection Manager

Apart from the built-in checks that ORAchk comes with, users can also add checks based on their business requirements for ORAchk to monitor, as shown in Figure 13 below.

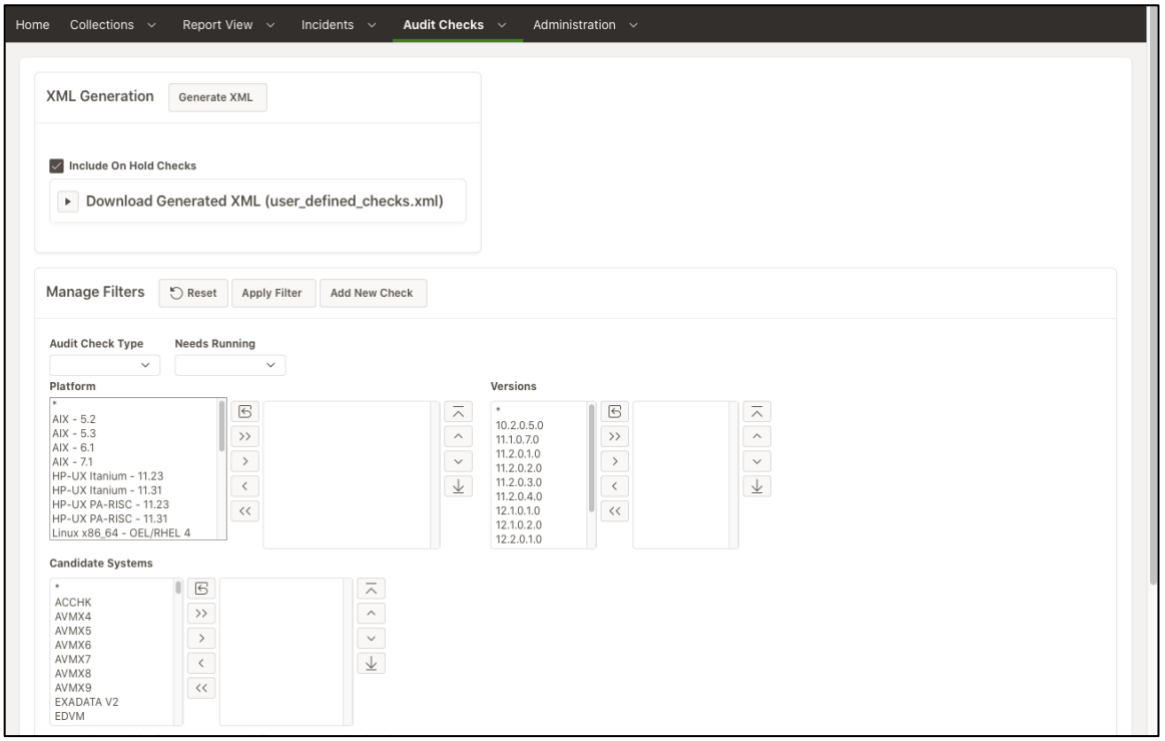


Figure 13: User-defined checks in Collection Manager

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. In addition, QoSM allocates server resources where they are required the most based upon performance requirements in terms of performance objectives and business criticality rankings to manage workloads to their service level agreements (SLAs).

Today, multiple and varied workloads are now being handled by a single server, each with its own set of performance objectives regarding its 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 tight 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 to optimize performance. Installing the Oracle Grid Infrastructure (GI) for RAC or RAC One Node database configures Quality of Service Management for enablement on a database-by-database basis.

In 21c, Oracle Database QoS Management now supports automatic policy set provisioning when adding databases to existing clusters improving provisioning and management in fleet or cloud deployments. So, while adding additional services, users no longer have to create a separate policy set that includes these new services. Instead, the new services can now be provisioned directly into the existing policy set through a simple script eliminating the rework and saving time and effort.

Quality of Service Management Architecture

Oracle Database QoS Management Server, as diagramed in Figure 14, retrieves database and OS metrics and topology from data sources including Oracle RAC and RAC One Node databases, Oracle Clusterware, and Cluster Health Monitor. Then, 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 correlation of this data by Performance Class occurs every five seconds. Added to this data is information about the current topology of the cluster and the health of servers. Finally, the Policy and Performance Management engine of Oracle Database QoS Management analyzes the data to determine the system's overall performance and resource profile 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, its bottlenecked resource, and, if possible specific corrective actions. The recommendation also includes its projected impact on all Performance Classes in the system.

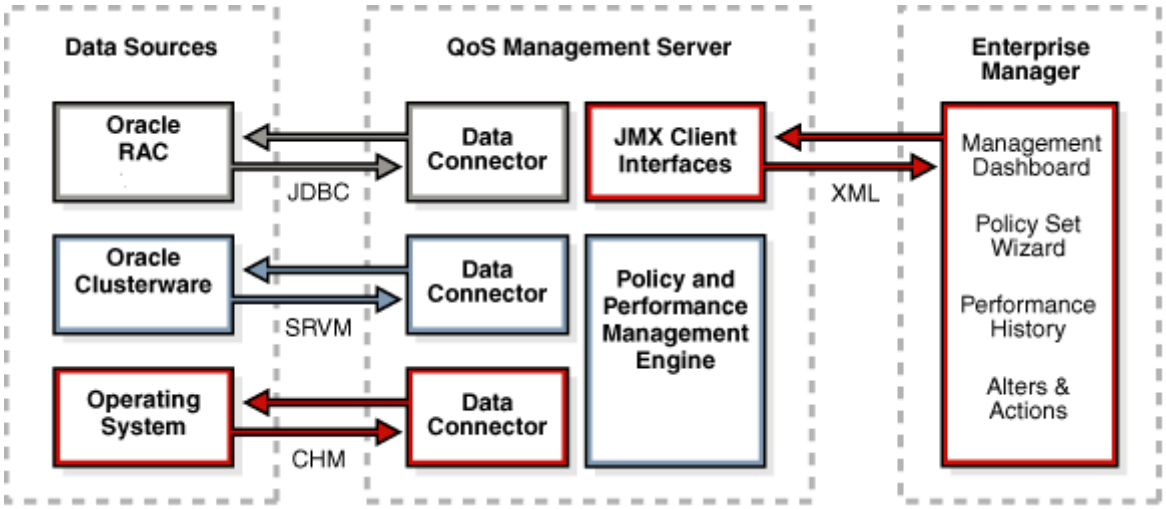


Figure 14: Quality of Service Management Architecture

Using Quality of Service Management to Manage Resources and Maintain SLAs

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

QoSM can be used in three phases or in combination: Measurement phase, Monitoring phase, and Management phase. In the measurement phase, QoSM helps to analyze the 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 15. This breakdown helps determine realistic performance objectives (in terms of average response time) for workloads.

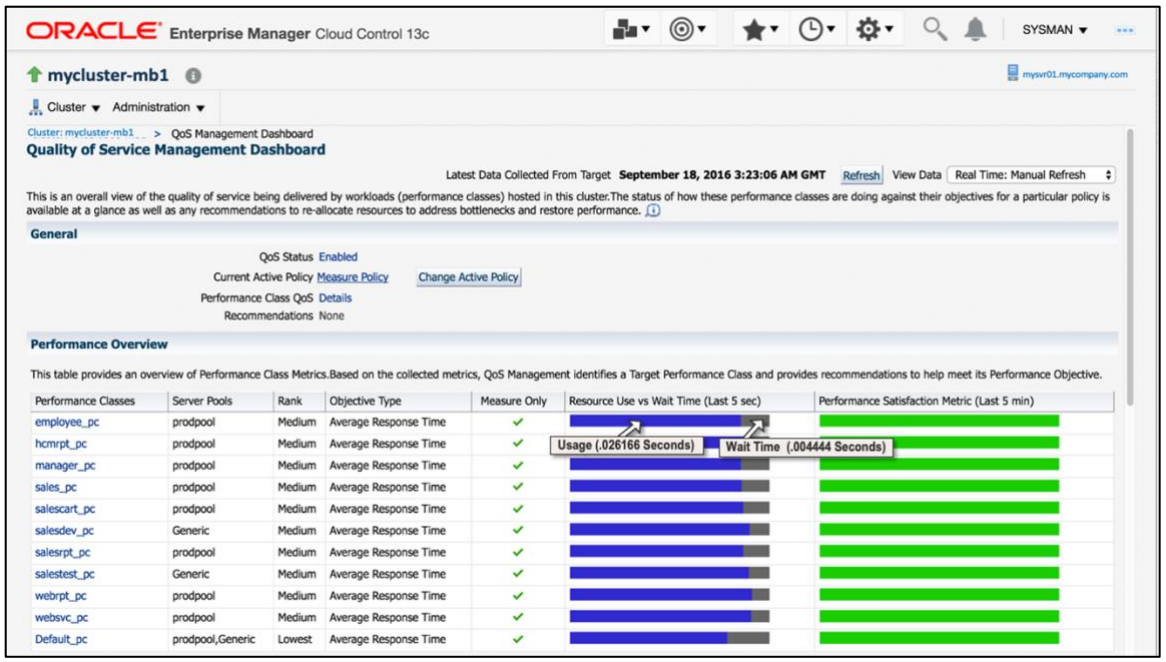


Figure 15: Quality of Service Management dashboard in the measurement phase

Quality of Service Management also identifies bottlenecked resources that degrade the performance of a workload. QoSM classifies resource wait time for a workload into CPU, I/O, Global cache, and Other wait time, as shown in Figure 16, where the highest wait time value 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 16: 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 17, 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 a sudden surge in the workload performance.

Figure 17: Quality of Service Management display of the performance history of the workloads

By default, workloads are classified based on service names. However, users can set additional parameters in the monitoring phase to classify workloads more granularly and set performance objectives and priority ranking for workloads through performance policy. QoSM uses this policy to compare current workload performance with set performance objectives. If performance objectives violations occur, additional workload resource wait time is represented by the red bar under the Resource Use vs. Wait Time column, as shown in Figure 18. The green bar represents the extra headroom when performance exceeds objectives. In addition, QoSM displays workload performance relative to its performance objective for the last 5 mins under the Performance Satisfaction Metric column. The red bar represents how long its response time exceeded its performance objective. QoSM 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.

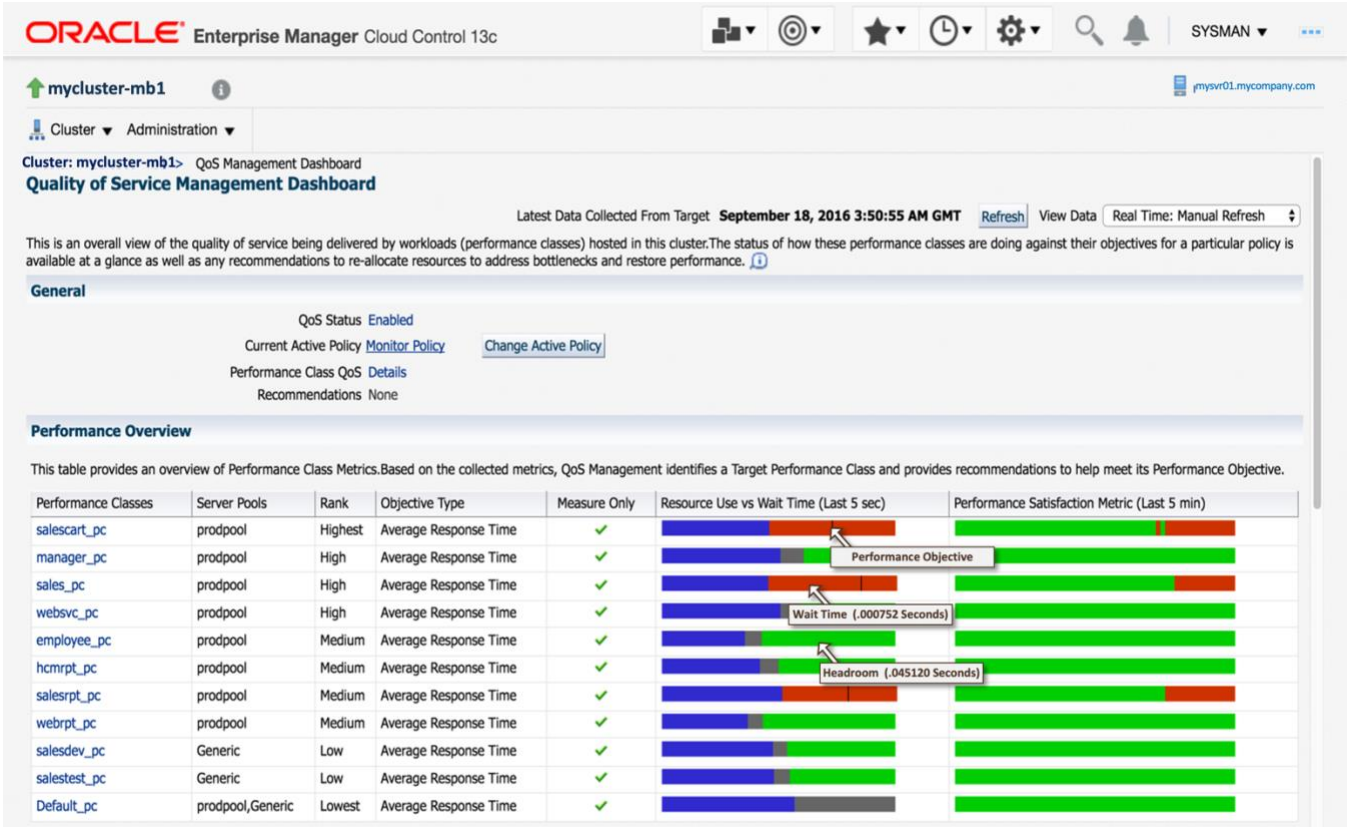


Figure 18: Quality of Service Management dashboard in the monitoring phase

In the management phase, users can set a new policy to manage workloads actively. In this phase, the user defines server pool resource parameters and performance objectives and ranks for workloads. Based on this policy, QoSM recommends resource reallocation to fulfill performance objectives for business-critical workloads and optimize performance for other workloads, as shown in Figure 19. Note that QoSM manages reallocation of CPU resources only to manage workload SLAs. Management mode is only available if the GIMR is installed locally in the cluster or a centralized location such as the Domain Services Cluster.

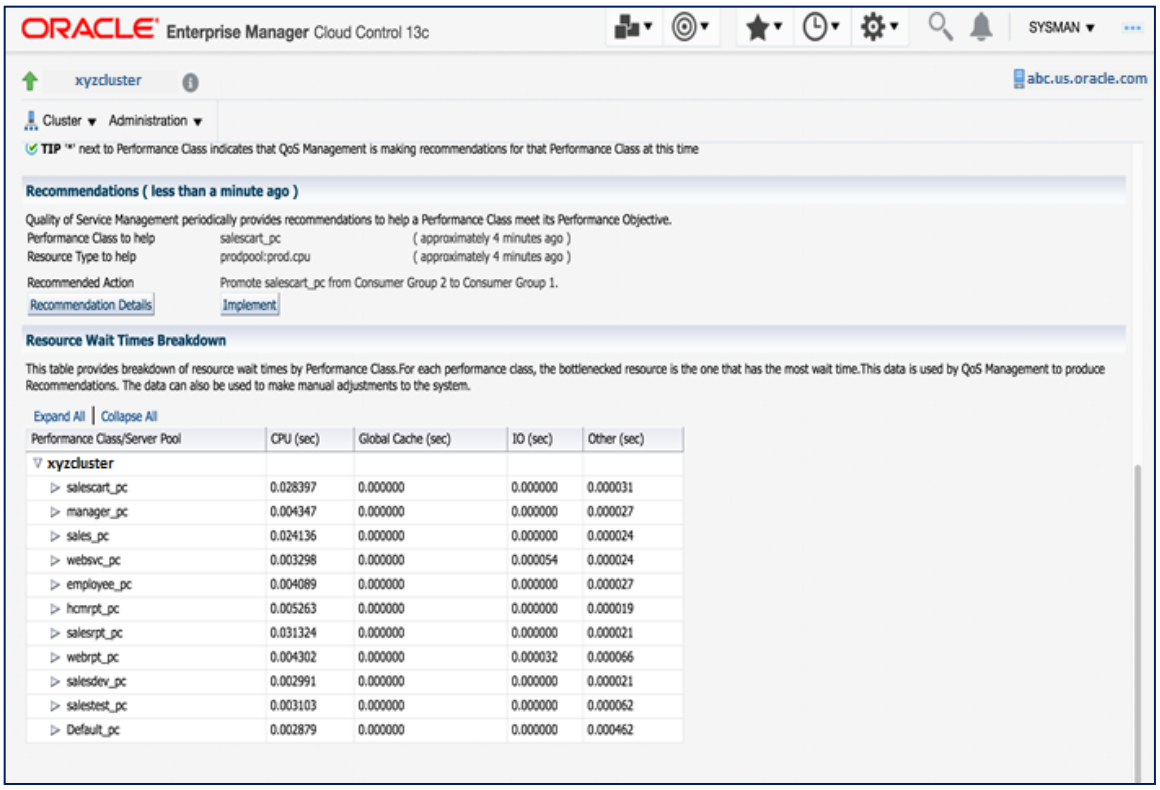


Figure 19: Quality of Service Management dashboard presenting recommendations in the Management phase

Baselining and Tracking Performance

While EMCC provides performance graphs for the most current hour, it is valuable to track performance over days or weeks, especially when determining a baseline set of performance objectives or whether more than one policy is required. Beginning in Oracle 21c, the Grid Infrastructure Management Repository (GIMR) that resides in its own Oracle DB Home or remotely in the DSC stores the historical data. Figure 20 shows reports generated in interactive HTML format using the **qosctl -gethistory** command.

Figure 20: Historical Performance Report - Overview

Users can interact with this report from a time axis as well as the Performance Class dimension. In addition to Performance Satisfaction Metric, Demand, and Average Response Time graphs, users can explore the associated Resource Use Times and Resource Wait Times to provide increased insight into the nature of any performance bottlenecks. This data is also presented for each discrete data point, as seen in Figure 21 using your mouse, and available for machine processing in JSON format in its data.js file located in the report output directory.

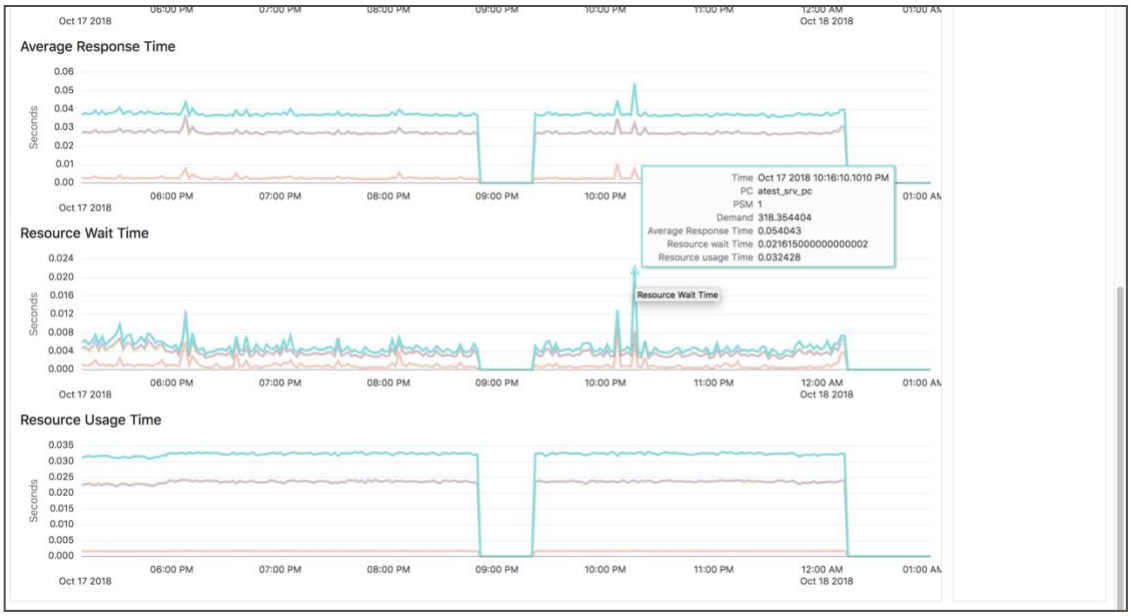


Figure 21: Historical Performance Report - Detail

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 identify bottleneck resources, analyze the performance history of the workloads, and manage the resources with its targeted bottleneck resolution recommendations to meet the SLAs.

Autonomously Preserves Database Availability and Performance During Hangs

Database hangs occur when another session blocks a chain of one or more sessions and prevents them from making 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 hangs and, in 21c, deadlocks. Creating a RAC or RAC One Node database enables Hang Manager automatically.

Hang Manager Architecture

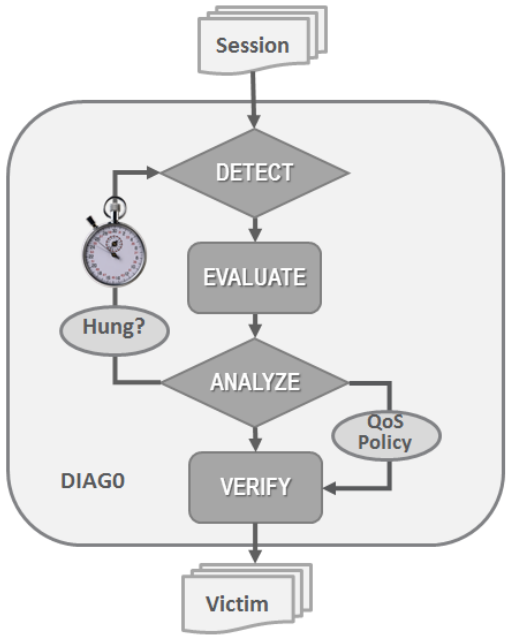


Figure 22: Hang Manager Architecture

Hang Manager autonomously runs as a `DIA0` background process within Oracle databases, as shown in Figure 22. 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. Next, 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 genuinely hung. After a set time, Hang Manager verifies these sessions as hangs in its Verify phase and selects a final blocker session as a victim session. Finally, it applies hang resolution heuristics to the victim session. If the hang does not resolve, it terminates the victim session, and if that fails, Hang Manager terminates the session process.

Applied Machine Learning in Hang Manager

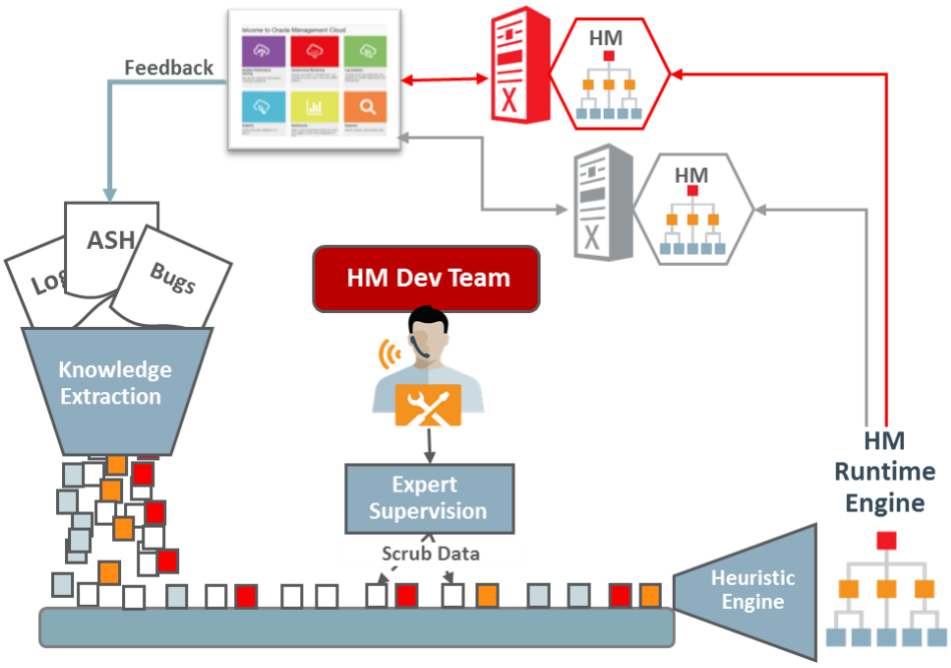


Figure 23: Applied Machine Learning in Hang Manager

Hang Manager uses Applied Machine Learning to enhance its model for hang detection and resolution continuously. Actual internal data collected by Oracle Support over the years and external customer data form the basis for the model. Purpose-built diagnostic technology extracts knowledge from the data collected. A team of experts is also dedicated to scrub the data to increase the accuracy of the model. Then this processed data helps create the model for the Hang Heuristics Engine, deployed to customers in the product. Finally, this engine autonomously performs real-time database hang detection and resolution.

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. However, admins 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, suppose a hang includes a session associated with a highly ranked critical Performance Class in the QoSM policy. In that case, Hang Manager expedites the termination of the victim session to maintain the 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. The details of complete hang resolution are also available in dump trace files for later reference, as shown below in Figure 24.

```
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
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 s                    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
  significant number of affected sessions.
   by terminating session sid:40 with serial # 43179 (ospid:13031)
```

Hang detected by hang manager

Session victim identified & requested termination

Blocker session terminated

Figure 24: Full Resolution Dump Trace File and DB Alert Log Audit Reports

Now the infrastructure may also cause performance issues. Let's look at a case where a hung or blocked ASM instance that prevents DB I/O. The same Hang Manager background code, but with different modes that resolved session hangs, is implemented in ASM instances, as shown in Figure 25.

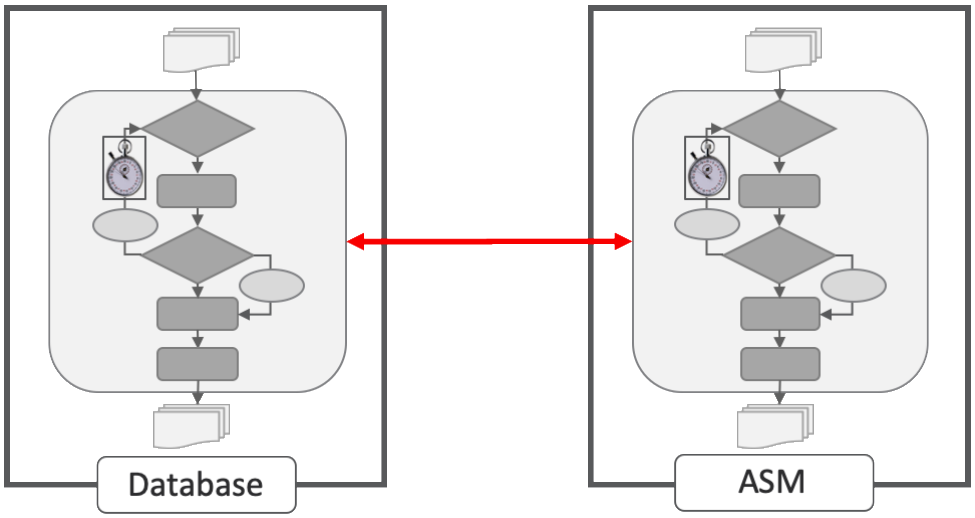


Figure 25: Bi-directional Hang Management between compute and storage tiers

However, its enhancements include communication with the DB instances it is serving. Should a hang develop in either tier, Hang Manager now resolves it, whether it means terminating a session or even the ASM instance. Killing an ASM instance is no longer an issue as starting in 12.2, all RAC clusters use Flex ASM, which allows the DB instances to connect to a remote ASM instance simply should the local one go down without data loss or corruption.

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 to prevent node eviction and maintain server availability. Installing the Oracle Grid Infrastructure (GI) for RAC or RAC One Node databases enables Memory Guard by default.

Memory Guard Architecture

As shown in Figure 26, Memory Guard runs as an MBean daemon in a J2EE container managed by Cluster Ready Services (CRS). Hosted on the qosmserver singleton resource, Memory Guard runs on any cluster node for high availability. Cluster Health Monitor sends a metrics stream to Memory Guard, providing real-time memory resource information for cluster nodes, including the 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. Thus, it relieves memory stress without affecting already running sessions and their associated transactions. After completion, the memory used by these processes starts freeing up and adding to the pool of available memory on the node. When Memory Guard detects that amount of available memory is healthy, it restarts services on the affected node.

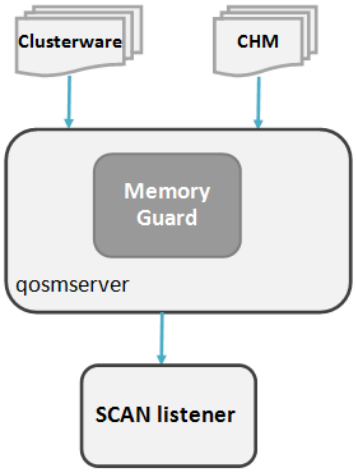


Figure 26: Memory Guard Architecture

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 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. In addition, Memory Guard sends alert notifications when it detects memory stress on a database node. The audit logs under $ORACLE_BASE/crsdata/node name/qos/logs/dbwlm/auditing contain the Memory Guard alerts and clears.

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 entry below showing the restarting of services after relief of memory stress:

```
<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 through Enterprise Manager Cloud Control, provides root causes and corrective actions for these issues on 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, Oracle Cluster Health Advisor raises a warning and generates an immediate targeted diagnosis and corrective action. Enterprise Manager Cloud Control integrates the Cluster Health Advisor root cause analysis and corrective action. It then displays these without the need for additional plug-ins.

Oracle Cluster Health Advisor stores the analysis results, 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.

Unlike most other Oracle AHF components, Cluster Health Advisor is only configured by default. It starts monitoring when a RAC or RAC One Node database starts in the cluster.

Cluster Health Advisor Architecture

As shown in Figure 27, Oracle Cluster Health Advisor runs as a highly available cluster resource, CHADDriver, on each node in the cluster. In addition, each Oracle Cluster Health Advisor Java daemon monitors the operating system on the cluster node and, optionally, each Oracle Real Application Clusters (Oracle RAC) database instance on the node.

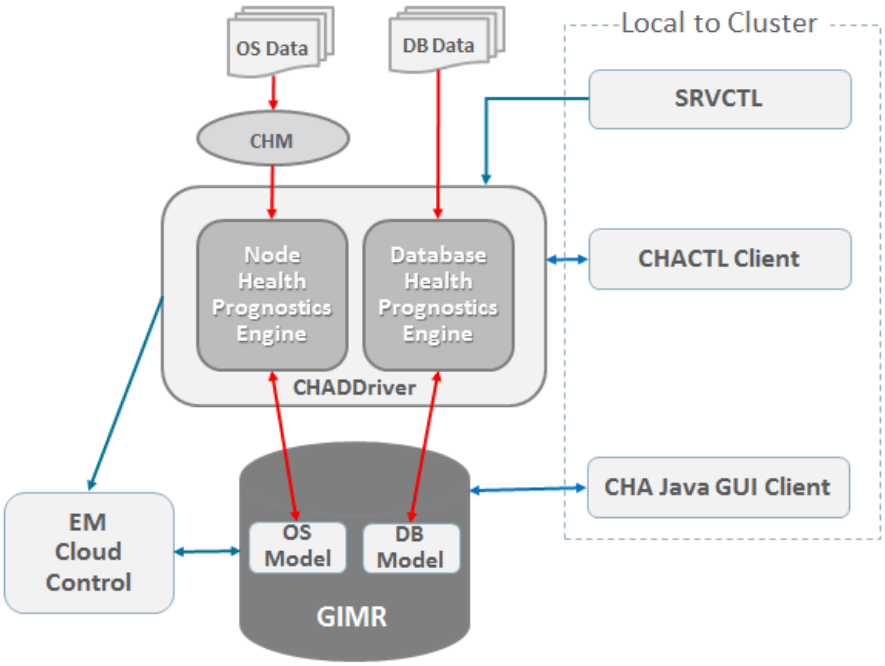


Figure 27: Flow diagram for Cluster Health Advisor architecture

The CHA daemon receives OS metric data from the Cluster Health Monitor and gets Oracle RAC database instance metrics from a memory-mapped file. Thus, 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 to analyze their health multiple times a minute.

The results of this analysis, along with any diagnosis and corrective action, are stored in the 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. If the GIMR is not installed locally in the cluster or centrally as in a Domain Services Cluster, this historical data will not be available either to CHACTL or EMCC.

Applied Machine Learning in Cluster Health Advisor

Cluster Health Advisor uses Applied Machine Learning to continuously enhance its model to detect a more comprehensive range of issues and their associated resolution. Actual internal data collected by Oracle Support and Cloud Services over the years and external customer data provide the training set for the models. In addition, purpose-built diagnostic technology extracts knowledge from the data collected. What differentiates the Applied Machine Learning Model for Cluster Health

Advisor is that a team of dedicated experts scrubs the data to increase the model's accuracy. The processed data is then used to create the sophisticated Bayesian Network-based diagnostic root cause models using over 150 different metrics received from OS and database. Then CHA includes these models for performing real-time prognostics.

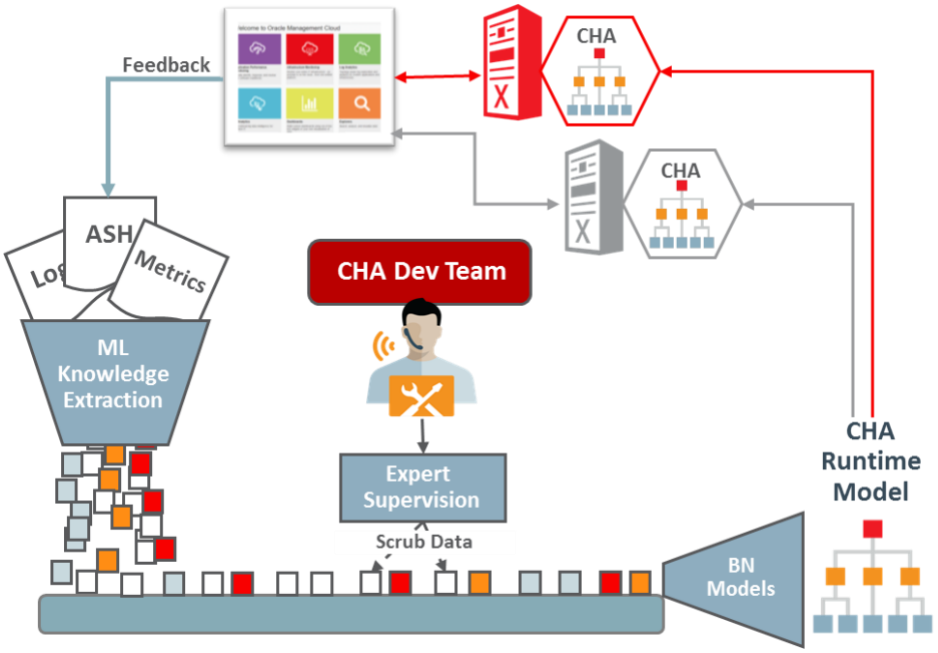


Figure 28: Applied Machine Learning in Cluster Health Advisor

A point to note here is that all the users get ready-to-use models with Cluster Health Advisor. In 21c there are specific Exadata node and database models included and loaded automatically. These pre-calibrated models mean that users do not have to undergo trial and error to train their models to arrive at the suitable model. Furthermore, since the applied machine learning models undergo continuous training and updates, users can receive these through quarterly patches.

Using Cluster Health Advisor for Prognosis of Potential Threats

Previously, Enterprise Manager Cloud Control gave only terse notifications for alerts and incidents that occurred. One such incident shown below reports an incident associated with ASM Cluster-wide disk utilization.

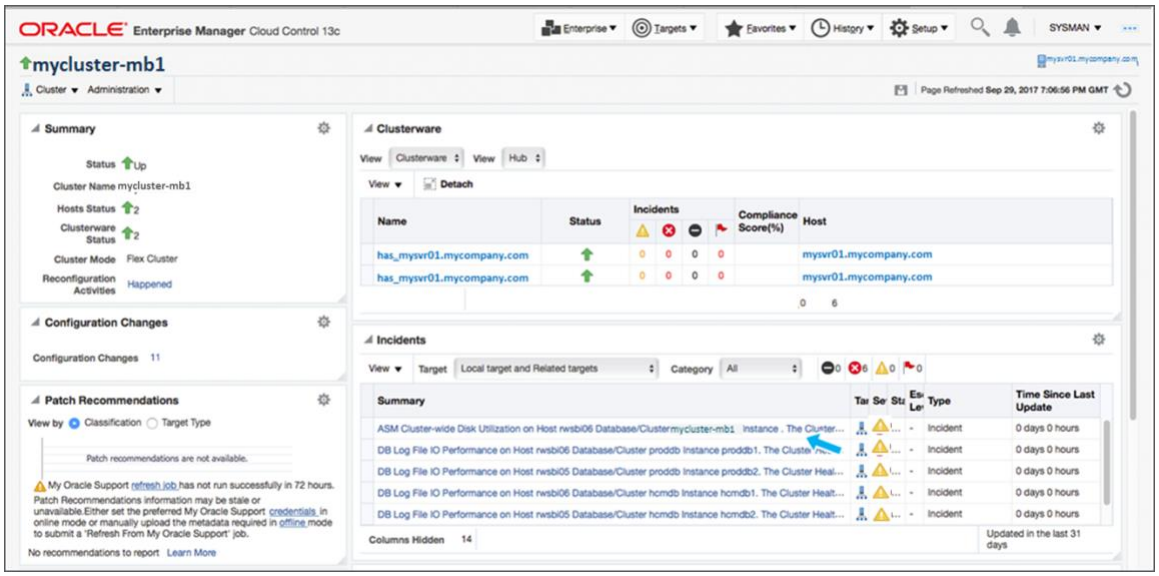


Figure 29: Typical EMCC Screen without CHA providing a Terse Alert Notification

However, with Cluster Health Advisor, users get early warnings of the issue in EMCC, as shown in Figure 29, and get a detailed diagnosis of the problem. For example, in Figure 30 below, CHA shows the precise diagnosis of the problem where CHA detected slower than expected disk performance. It also provides the root cause analysis and the corrective action. In this case, CHA suggests high disk I/O demand from other servers as the root cause, which increased the utilization of the shared disks. And the corrective action is to add disks to the database disk groups.

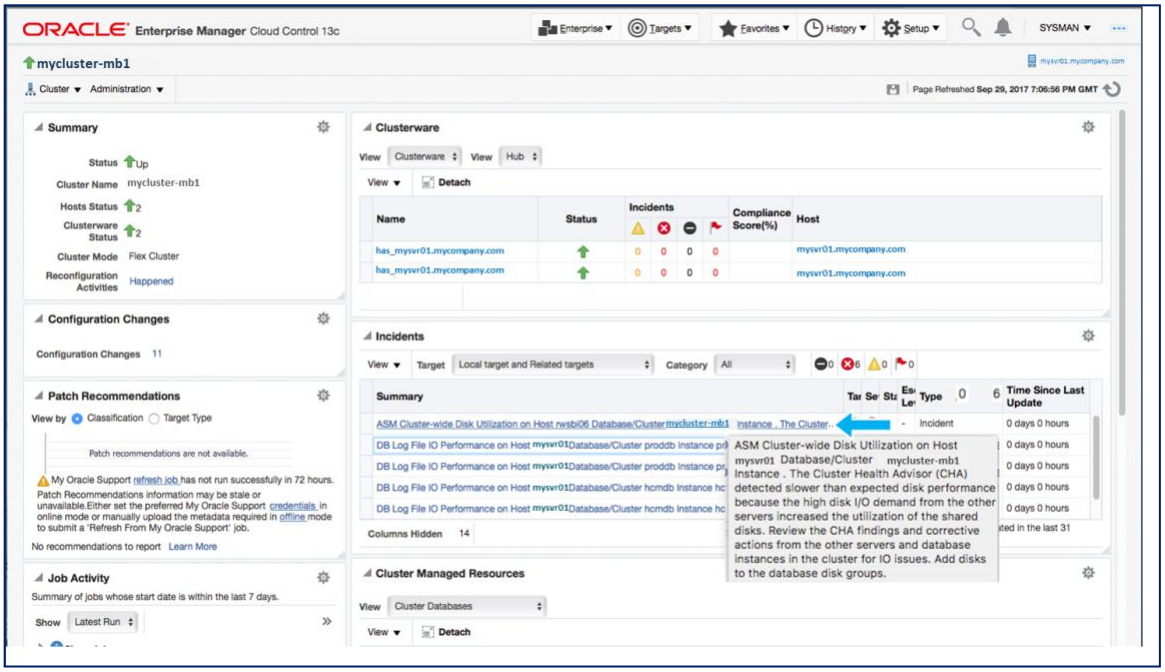


Figure 30: EMCC Screen with Detailed Issue Analysis through CHA

Cluster Health Advisor uses applied machine learning models to provide these analyses. 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 the accuracy and sensitivity of node and database models. Since workloads may vary on specific cluster nodes and Oracle RAC databases, Cluster Health Advisor also can create, store, and activate multiple models with their specific calibration data. CHACTL also manages this functionality. 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 CHA 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 users must use CHACTL to turn on monitoring for each database explicitly.

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 for known issues, there might occur unknown issues that have not been previously encountered.

Oracle Autonomous Health Framework component Trace File Analyzer (TFA) runs in daemon mode a. It helps in the quick resolution of these issues by autonomously collecting data from logs intelligently (Smart Collection) promptly across multiple nodes and speeding issue diagnosis with Oracle Support Services. These real-time collections are critical when data is frequently lost or overwritten, and the 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.

In 21c, TFA extends from collecting data intelligently to allowing for quick self-diagnosis of the issue by finding relevant information, from the collected data, for the issue at hand through TFA Service's receiver component, as discussed below. Oracle Trace File Analyzer also includes new single command, Service Request Data Collections (SRDCs), explained below.

Trace File Analyzer Architecture

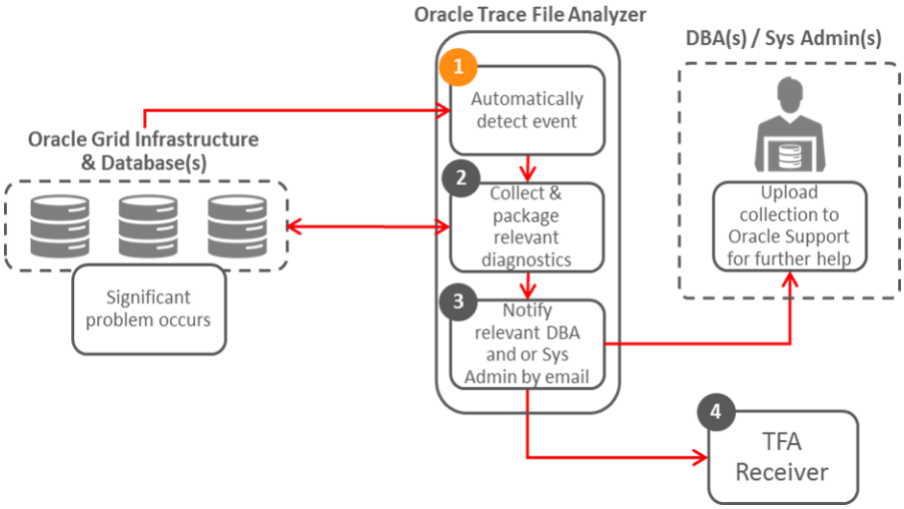


Figure 31: Trace File Analyzer Architecture

As shown in Figure 31, when running in daemon mode, TFA monitors Oracle logs for events symptomatic of a significant problem as step 1. In step 2, TFA then starts an automatic smart diagnostic collection based on the event type detected. The data collected depends on the event detected. TFA coordinates collection cluster-wide, trims the logs around relevant periods, and then packs all collection results into a single package on one node. Once the collection is complete, TFA sends an email notification that includes the details of where the collection results are to the relevant recipients as step 3. The recipients can then upload the collections to Oracle Support Services for further help. Users in 21c can now also upload the collections to TFA Analyzer Service available on Domain Services Cluster (discussed later) and use the TFA Service for a quick self-diagnosis of the issue as step 4. Also, in 21c, using the new one command SRDCs, users can collect precisely the correct diagnostic data required to diagnose a specific type of problem quickly and efficiently when they need help from Oracle Support. Users can then log an SR with the resulting zip file to get quick resolution of their issue.

Smart Collection with Trace File Analyzer using Applied Machine Learning

TFA uses applied machine learning models to autonomously and intelligently collect only the logs relevant to the issue illustrated in Figure 32, reducing the log files to a small list of potential candidates where the issue can be found. The data for these models are extracted from logs, SRs, and bugs collected by Oracle Support over the years. This rich dataset is then refined further by domain experts, which differentiates these models. The knowledge extracted in this step is then used in creating the models which are shipped with TFA to work with live logs on the user's clusters. Just like with Cluster Health Advisor, the models shipped with TFA are also ready-to-use models that do not require any training by the users. These models are also updated regularly. Users can get updates for these models through patches.

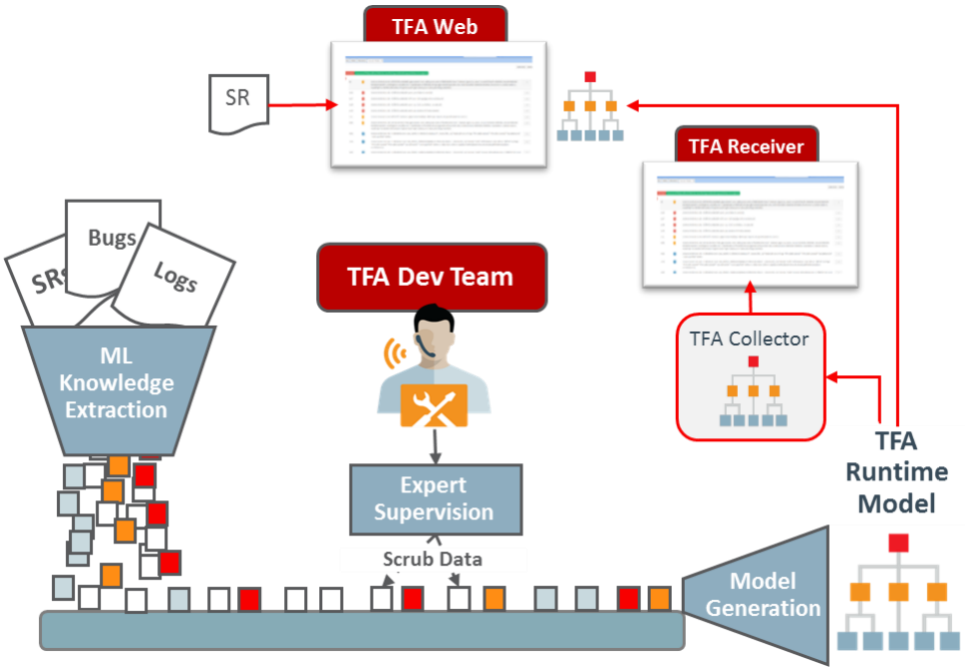


Figure 32. Applied Machine Learning in TFA for Smart Collection

The data collected by TFA with these models can be sent to Oracle Support Services for further diagnosis. Because this data is relevant and complete, it reduces the round trips between the users and Oracle Support for issue diagnosis, thereby increasing the speed of issue resolution.

Oracle Autonomous Health Framework Support for Domain Service Cluster

Oracle AHF generates and stores diagnostic data while diagnosing and resolving availability and performance issues in the database system. A 4-node cluster, on average, generates 6-7 GB of diagnostic data for retention of 3 days. This quantity would create overhead by consuming local resources. Furthermore, Oracle AHF components interact and use data generated by each other. This integration becomes convenient if the entire data is stored in one place instead of in local repositories of each component. In 21c a centralized GIMR Service is supported in the Domain Services Cluster that host data from both 19c and 21c standalone client clusters. Here in Figure 36, client database cluster are registered with a common Management Repository Service.

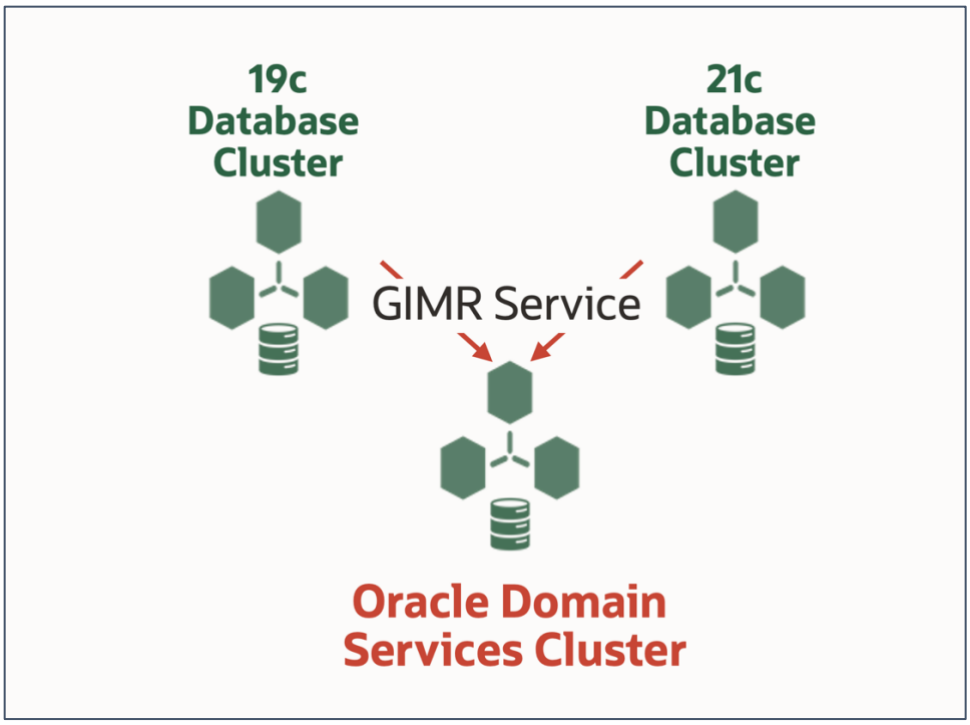


Figure 36: Oracle Centralized GIMR Service on Domain Service Cluster

Oracle AHF is therefore centrally supported in the Oracle Domain Services Cluster, where the overhead of storing diagnostic data of Oracle AHF is offloaded to 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 client clusters, and also helps to reduce the local footprint of Oracle AHF.

Conclusion

With the globalization of businesses, database systems need to be available and perform consistently so that customers may perform transactions 24x7. Any daily operational issues that threaten the 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 potential threats to the database system and provide corrective actions to fix them. For problems that occur, Oracle AHF helps resolve them quickly with minimal effort by identifying the issue, diagnosing its cause, and providing resolutions. For problems requiring Oracle Support Service (OSS), Oracle AHF also collects relevant information needed by OSS to resolve the issue quickly. Oracle AHF, therefore, provides a solution at every step – prevent problems before they occur, resolve issues when they arise, and expedites the resolution of issues that require OSS assistance. Therefore Oracle AHF is a complete solution to maintain the availability and manage the performance of Oracle database systems.

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle