ORACLE

Automated Kernel Refresh

Leverage the capabilities of automated kernel refresh methodology to achieve greater resiliency with Oracle JD Edwards EnterpriseOne

October, 2021 | Version 1.0

Purpose Statement

This document provides an overview of features and enhancements included in release 9.2.6. It is intended solely to help you assess the business benefits of upgrading to 9.2.6 to leverage the benefits of automated kernel refresh for your JD Edwards and to 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

List of images

List of tables

Executive Summary

Today's dynamic businesses are working round the clock to meet their customer demands across the globe. Enterprise resource planning systems are a critical backbone to these businesses and continuous system availability is an imperative. In order for Oracle JD Edwards customers to provide continuous system availability, it is necessary for EnterpriseOne kernels to be in active state of communication at any given point of time. EnterpriseOne kernels are the foundation for every operation, and as such kernel failures have drastic impact on the business availability. To avoid kernel failures, it is necessary to have an active state of kernel environment and their associated database connections, both of which make kernel connection management important.

Historically kernel connection management is a manual process requiring downtime. System administrators tend to keep their EnterpriseOne kernels running for long durations spanning several weeks or months and do not refresh the static kernels regularly. This practice can result in stale state of the kernel environments, database connections, users, and associated table handles for static kernels, which can lead to kernel failures. It also introduces challenges in re-establishing static kernel database connections in case of any disruptions, because to do so requires manual intervention.

The automated kernel refresh functionality provides graceful zero-downtime refresh of static kernel states, which greatly reduces the risk of kernel failures, thus enabling customers to benefit from improved business continuity and data integrity. This refresh ensures static kernels are in an active state with necessary database connections at any given point of time by performing a graceful refresh for the environment, users, and associated table handles. As opposed to manual intervention, a refresh provides a faster connection to re-establish the normal function of static kernels as a result of unplanned disruptions. This capability frees the system administrators from the overhead of manual kernel lifecycle management.

This document describes the various methods and best practices for leveraging the automated kernel refresh capability for JD Edwards. The document also describes the performance characterization of automated kernel refresh capability and its benefits towards providing improved system availability and resiliency for JD Edwards customers.

Overview of JD Edwards Kernels

An EnterpriseOne kernel process is an essential part of the design of the EnterpriseOne architecture that runs on the Enterprise Server and performs a specific set of functions and responds to requests from client machines, servers, and third party software. In order to meet specific business needs with different functional requirements, a number of kernel definition types have been designed. The kernels presented in Table 1 are those that are impacted by the automated kernel refresh feature.

There are four kernel processes that are involved with the EnterpriseOne interactive and batch testing presented in this document and are highlighted in **bold** in Table 1. Each kernel process has a specific function design for use in the EnterpriseOne application. It is important to understand the function of each kernel process as to the observed effects of database refresh event.

KERNEL DEFINITION	KERNEL NAME	DESCRIPTION
[JDENET_KERNEL_DEF2]	**UBE**	**Controls the submission of batch process requests from clients and schedules batch jobs in the job queue.**
[JDENET_KERNEL_DEF4]	**Security**	**Provides the credentials and application tokens for security access to the EnterpriseOne application. Additionally, it also handles all user profile and security related table updates.**
[JDENET_KERNEL_DEF6]	**Call Object (COK)**	**Responsible for executing the business function logic through requests received from JDENET. If the user has already established a request to the COK process, JDENET will route subsequent requests to the same COK process. The only time a business function will run on a different COK process is when the BSFN is OCM mapped to a different server.** **Only COK kernels support multi-threaded processes. All other kernels are meant to handle data/cache one at a time.**
[JDENET_KERNEL_DEF10]	Scheduler	Handles the scheduler application requests. The scheduler launches batch processes based on information in the job master table (F91300)
[JDENET_KERNEL_DEF11]	Package Build	Responsible for processing package build requests.
[JDENET_KERNEL_DEF12]	UBE Subsystem	Controls the submission of UBE subsystem requests. This kernel adds/updates/deletes records from F986113.
[JDENET_KERNEL_DEF13]	Workflow	Manages workflow requests from the application.
[JDENET_KERNEL_DEF14]	Queue	This kernel launches the job queues from its initiation and processing. It updates the Work with Job Queues table (F986130) table and is responsible for the movement and writing of the kernel log and movement of the logs to the PrintQueue directory.
[JDENET_KERNEL_DEF16]	XML List	This kernel provides List/GetNext functionality to collect records from ERP. XML List takes an XML document as an input and returns an XML document with the requested data. A list can be a table, business view, or data from a table conversion.
[JDENET_KERNEL_DEF22]	XML Dispatch	Kernel for routing XML messages to their proper kernel.
[JDENET_KERNEL_DEF30]	**Metadata Kernel**	**The metadata kernel runs in a JVM and processes the XML specification access requests readable by the call object kernel.**
[JDENET_KERNEL_DEF31]	XML Publisher	Handles XML publishing requests in the application

Table 1. EnterpriseOne Kernel Processes

OVERVIEW OF AUTOMATED KERNEL REFRESH

All kernel processes listed in Table 1 use static pooled connections to the database. A database connection is a physical process connection between the EnterpriseOne server and a database instance. A connection pool is a cache of connections to an Oracle database used by the EnterpriseOne application.

For EnterpriseOne static kernels, all connections to the database remain static for each kernel process as long as there are EnterpriseOne user processes active in that kernel process. A dynamic pooled connection would, however, create and destroy connections in the kernel process. For example, if a kernel process in EnterpriseOne is connected to a primary node in a 2-node Oracle RAC configuration, it remains on the primary node for the life of the kernel process.

The static EnterpriseOne application kernel processes requires a 'refresh' event trigger to force the connections to the database to terminate and refresh again. A 'refresh' event is triggered through the Server Manager Console, either manually or automatically as scheduled.

In the example of a 2-node Oracle RAC failover, this entails moving the database connections from the primary node to the secondary node. Although this example is suited for the situation with database planned maintenance outages, it can also occur on a single database instance where communications to the database need to be re-established without the need to recycle EnterpriseOne services. This can occur when one or more of the EnterpriseOne kernel processes has lost its active kernel state in cache or within the file system of the process.

A database session is the logical structure in the database instance memory that represents the state of a current user accessing the database. A session lasts from the time the user is authenticated by the security kernel process in EnterpriseOne by the database until the time the user disconnects or exits the EnterpriseOne application. The database connection is also terminated if the EnterpriseOne session timeout expires. Database sessions established by the EnterpriseOne application can be viewed through the Server Manager Console or by querying the active sessions on the database server directly (see appendix).

After the database session is certified by the security kernel, the other kernel processes listed in Table 1 may be involved in the processing of the EnterpriseOne interactive user application and batch requests.

The key feature of kernel refresh is to provide a method for the kernel processes to establish a new database connections and sessions to a database without having to recycle EnterpriseOne services. To maintain business continuity, when an EnterpriseOne kernel refresh event is requested, the kernel process retains the same process identification, keeping a caching state within the process.

For more information on the two methods that EnterpriseOne kernel processes are coded, see the 9.2 Server Manager Guide and Appendix B.

Automated Kernel Refresh Testing

The main objective of the testing to identify gaps in the EnterpriseOne processes during a kernel refresh event of the Oracle database that would affect customers business continuity. The test process included batch and interactive testing as shown in Table 2.

TEST CASE	TESTCASE TYPE	DESCRIPTION
1	Batch	Submission of set of nine short running UBEs • Submission initiated using runube utility • Approximately 300 batch jobs are executed for each test
2	Interactive	Interactive user applications • JMeter utility used to generation load • Load consisted of a load of 40 users running the different EnterpriseOne module application for a period of an hour

Table 2. EnterpriseOne Kernel Testing Scenarios

Table 2 refers the two test case categories – batch and interactive. Batch test cases is the submission of a set of nine short running UBEs. Interactive testing involves ten simultaneous EnterpriseOne users various interactive applications.

Test results performed with these use cases referred in Table 2, will be covered in the upcoming section of this document. Details listing of use cases are specified in Appendix C.

Metrics

Metrics are used to indicate whether the application is performing within the range of expected parameters or whether the performance is below acceptable standards. For testing EnterpriseOne using the kernel refresh feature, the criteria for success was the metrics surrounding the success or failure of the interactive and batch processes and whether the kernel process was able to establish a new database connection after the refresh action event was instantiated. The test cases are a means to exercise the specific kernel processes that are evaluated in this testing.

Results Summary

Table 3 below provides the conditions of the test and results when the 'refresh' event action was initiated.

TEST CASE	TESTCASE TYPE	TESTING RESULTS
1	No interactive or batch activity	Kernel refresh activity logs show that connections to the database are re-established and available for EnterpriseOne requests
2	Interactive and batch activity during automated kernel refresh	Variability in success and failure of interactive and batch processes when database connections are re-established
3	A kernel refresh event occurred allowing new batch and interactive users to continue business continuity	New interactive and batch processes are able to complete successfully

Table 3. Kernel Refresh Test Results

For test scenarios 1 and 3 in Table 3, interactive and batch processes completed successfully, which is normal. Test 2 in Table 3 however had variable results as described below.

The variability in the success of interactive and batch processes depends greatly on when the refresh event occurs and what state the EnterpriseOne process was in its processing. EnterpriseOne applications already have safeguards in place for momentary database connections issues and the kernel processes will retry the database requests, but only for a predetermined and non-configurable amount of time and duration, which is 75 seconds.

The variability in the success or failure of an EnterpriseOne process when a database connection is established is explained by the following situations:

- Successful completion depends largely on the state of the cache of the process at the time of refresh. Process cache is the area in the database where table, business view, or other process information is stored from initiation to time of completion. In cases where the cache is lost as a result of a forced kernel refresh activity, and that caching information is essential for the completion of the request, then that process fails. This is considered a catastrophic process failure.
- Successful completion depends on the database connection re-establishing kernel process connections within an acceptable time period. If the EnterpriseOne kernel processes refresh and subsequent refresh occurs after a number of EnterpriseOne process timeout variables, the process fails. There are two timeouts that are important in this process, the user session timeout on the HTML server, the EnterpriseOne session timeout, and result set timeout.
    - User Session Timeout (SessionTimeout): Governs HTML, ADF, and IAS processes. This timeout setting is specific for interactive users using the EnterpriseOne graphical user interface. The default value is 20 minutes.
    - EnterpriseOne Server Timeout (enterpriseServerTimeout): The maximum time before a JDENET operation is considered to have timed out. Time out value for requests sent to the Enterprise Server. Default is 60 seconds (Enterprise Server) and 90 secomds (HTML server). There is an enterpriseServerTimout on both the Enterprise and HTML Server configurations.
    - Result Set Timeout (resultSetTimeout): The interval to wait before retrying database operations returns before failure. Default is 60 seconds

  These are the more common timeout causes of EnterpriseOne process failures due to timeout constraints

The failures in the testing for kernel refresh for use case 2 of Table 3 fell into these two main categories:

- Within 1 minute of the refresh event initiation, UBE (DEF2), Security (DEF4), and Metadata (DEF30) process logs showed both a refresh event request action and a refresh message in the logs on the EnterpriseOne Server.
- In the case of the Call Object kernels, a new process is automatically spawned, establishing a new connection to the EnterpriseOne Database Server. If there is activity in either the batch process or interactive user requests through the EnterpriseOne application, the original Call Object kernel process remains. This is by design to allow active database sessions to the EnterpriseOne application to complete.

Testing Summary

The purpose of the automatced kernel refresh functionality is to decrease the number failures with EnterpriseOne interactive and batch processes. Interactive user and batch processes will unavoidably fail if the kernel processes supporting their processing have lost state because of a lack of connection to the database. Testing demonstrated that the kernel refresh feature available in EnterpriseOne Tools 9.2.6.0 works as designed.

Conclusion

Automated kernel refresh provides JD Edwards customers with improved resiliency and system availability. The automated handling of kernel refresh with decreasing customer downtime provides improved system availability without manual intervention. Automated kernel refresh aids in graceful handling of disruptions and reduces impact on the operations of JD Edwards customers.

Appendix A: Testing Information

Pre-requisites

In order to configure your JD Edwards to leverage automated kernel refresh, the release levels required are:

- JD Edwards EnterpriseOne Applications 9.2
- JD Edwards EnterpriseOne Tools Release 9.2.6 or above
- Supported Database Versions:
    - Oracle Database 19c (19.11 or above)
    - Oracle Autonomous Database 19c running on Dedicated Exadata Infrastructure, Shared Exadata Infrastructure or Exadata Cloud@Customer
    - Microsoft SQL Server 2019
    - IBM DB2 LUW 11.5.4 on IBM AIX 7.2
    - IBM DB2400 on IBM i 7.3 and 7.4
- Testing was performed with Oracle Connection Pool enabled

Appendix B: EnterpriseOne Kernel Refresh

The automatic kernel refresh and on-demain manual kernel refresh features were introduced with Tools Release 9.2.6.0 and the documentation is available in the 9.2 Server Manager Guide. It also will be covered here in a lesser detail.

Automatic Kernel Refresh

In order to avoid planned and unplanned Oracle database connection disruptions, the JD Edwards logic allows an idle kernel process to re-establish connections to the Oracle database. An automatic kernel refresh involves a kernel process logout and reconnection of the bootstrap session with the Oracle database, ensuring connection integrity. The bootstrap security login is controlled on the EnterpriseOne server and is shown in Image 2.

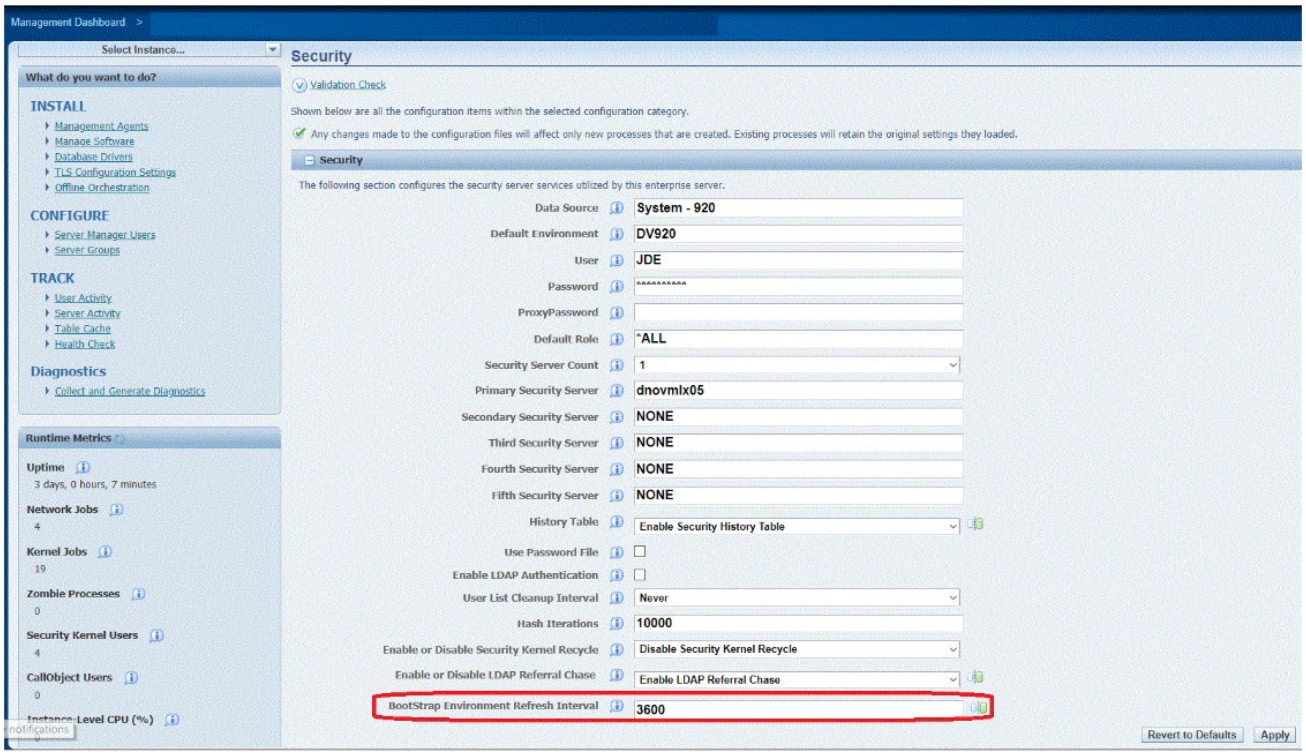

Image 1. EnterpriseOne Automatic Kernel Refresh

Image 1 shows the bootstrap value of 3600 (60 minutes), and is found in the Advanced Configuration setting under Security. The default value for automatic kernel refresh is 0 (disabled).

Performance Tuning Automatic Kernel Refresh

As shown in Image 1, the Bootstrap Environment Refresh Interval variable is set to 3600 (60 minutes). The question of what is the best value for this variable setting? To answer this question, two concepts must be discussed:

1. Recovery Time Objective (RTO) – The time duration in which a customer wants to be able to recover from an EnterpriseOne kernel database connection loss. For purposes of this discussion the setting for the Bootstrap Environment Interval is equal to the RTO.
2. Timeout Values – The EnterpriseOne time out variable settings that, if less, will make this variable setting of no consequence.

In testing, the first consideration was to make the RTO setting less than the any of the EnterpriseOne timeout settings. For the Result Set Timeout or EnterpriseOne Server Timeout, the value of RTO can be set to 60 seconds. Testing continues to determine if this low of a setting has an impact to performance. Initial indications are that the additional code contributes less than 1% to the overall process runtime, almost indistinguishable from a larger RTO setting where the automatic kernel refresh code is called more frequently.

Result Set Timeout operations assist when there are large data finds, scroll to end operations, grid fetches, data browser or any process that is database intensive. EnterpriseOne Server Timeouts can occur if certain logic (BSFN) on the Enterprise Server depends on database connection performance resulting from delays in connecting to the database which in turn causes a timeout.

An RTO value of 20 minutes (1200) is set to cause the EnterpriseOne kernel processes to reconnect to the database prior to the user session timeout default value of 20 minutes.

At present, a value of 60 seconds seems to be a better setting than 60 minutes for the RTO variable to control kernel refresh activity within the kernel processes on the EnterpriseOne Server.

On-Demand Kernel Refresh

EnterpriseOne Tools Release 9.2.6.0 also provides a mechanism to force the refreshing of the Oracle sessions and connection of the database by the Enterprise kernel processes. The Refresh Environment button on the Enterprise Server dashboard in the Server Manager Console is shown in Image 2.

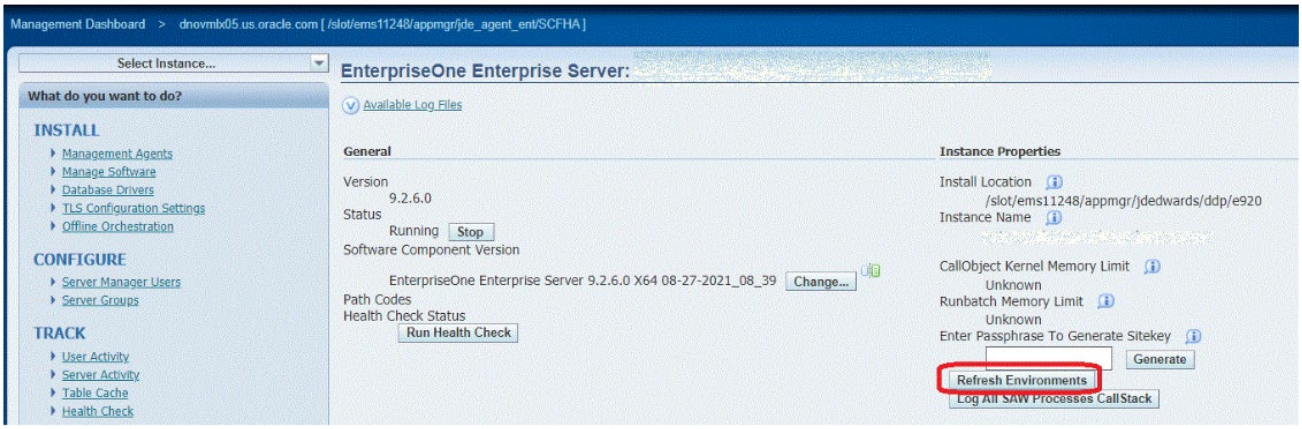

Image 2. EnterpriseOne On-Demand Kernel Refresh

Image 2 shows the Refresh Environment button to be used to force the EnterpriseOne kernels to re-establish connections to the database.

Appendix C - Test Cases

Extended testing includes more batch jobs as well as interactive processes along with baseline tests described in the main section of this document. Test cases are categorised and described in the table below.

S.NO	TESTCASE TYPE	UNIT TESTING	LOAD TESTING

1	Batch	Submission of long running UBEs - R31410, R3483, R42565 and R43500	Submission of 9 short running UBEs and long running UBEs mentioned in the unit testing
2	Interactive	Execution of below use case script in JMeter for 10 User 10 Iterations • Supplier Ledger Inquiry (P0411) • Apply Receipts (P03B102) • Work Order Completion (P31114) • Sales Order Entry (P42101)	All the Interactive scripts mentioned in unit test executed together as a 40 User test with 10 Iterations each

Table 4. EnterpriseOne Detail Testing Use Cases

Table 4 describes the detailed use cases of batch and Interactive testing process.

Appendix D – Test configuration

The JD Edwards EnterpriseOne components that were configured in the architecture for the testing process discussed in this document are listed below:

The JD Edwards EnterpriseOne components were implemented on Oracle Cloud Infrastructure as a standard VM.

JD Edwards EnterpriseOne Enterprise Server:

- Oracle Enterprise Linux 7.8
- Oracle Database 19c (19.12.0.0.0) client

Database Server:

- Oracle Enterprise Linux 7.8
- Oracle Database 19c

  Enterprise Edition Release 19.12.0.0.0 - Production

HTML Server

- Oracle Enterprise Linux 7.8
- WebLogic Server 12c (12.2.1.4.0); Java JDK (1.8.0_272)

Deployment Server:

- Windows Server 2016 Standard
- VM Standard 2.4 with 4 OCPUs
- 4 OCPUs x Intel Xeon Platinum 8167M CPU @ 2.00 GHz
- 60 GB RAM

Server Manager Console:

- Oracle Enterprise Linux 7.8

Test Controller:

- Windows Server 2016 Standard
- VM Standard 2.4 with 4 OCPUs
- 4 OCPUs x Intel Xeon Platinum 8167M CPU @ 2.00 GHz

- 60 GB RAM
- Apache JMeter 5.1.1.r1855137

Software: JD Edwards EnterpriseOne Application 9.2 Update 4 with Tools Release 9.2.6.0

Connect with us

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

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