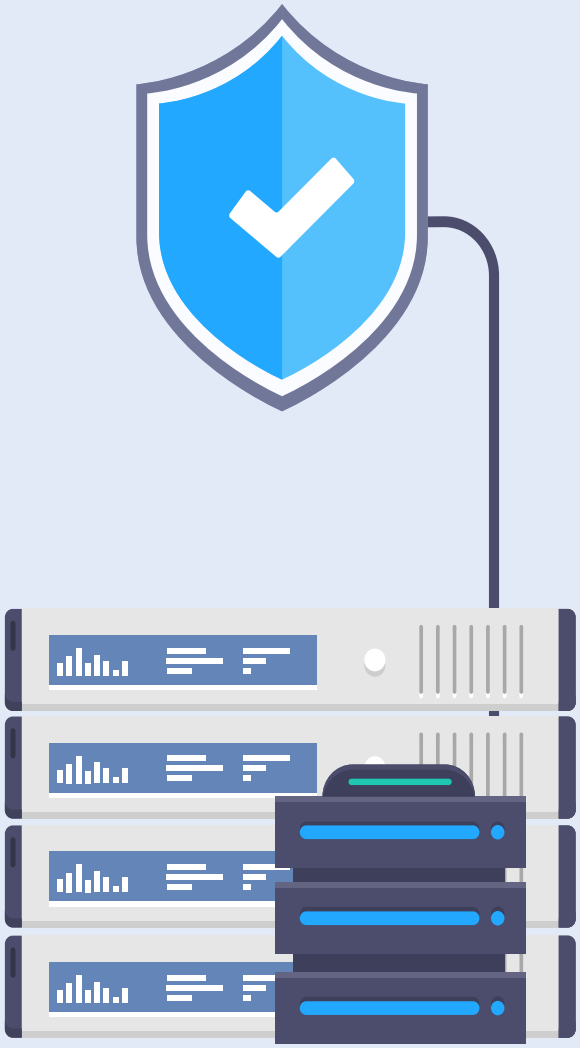
CCN-CERT

BP/22



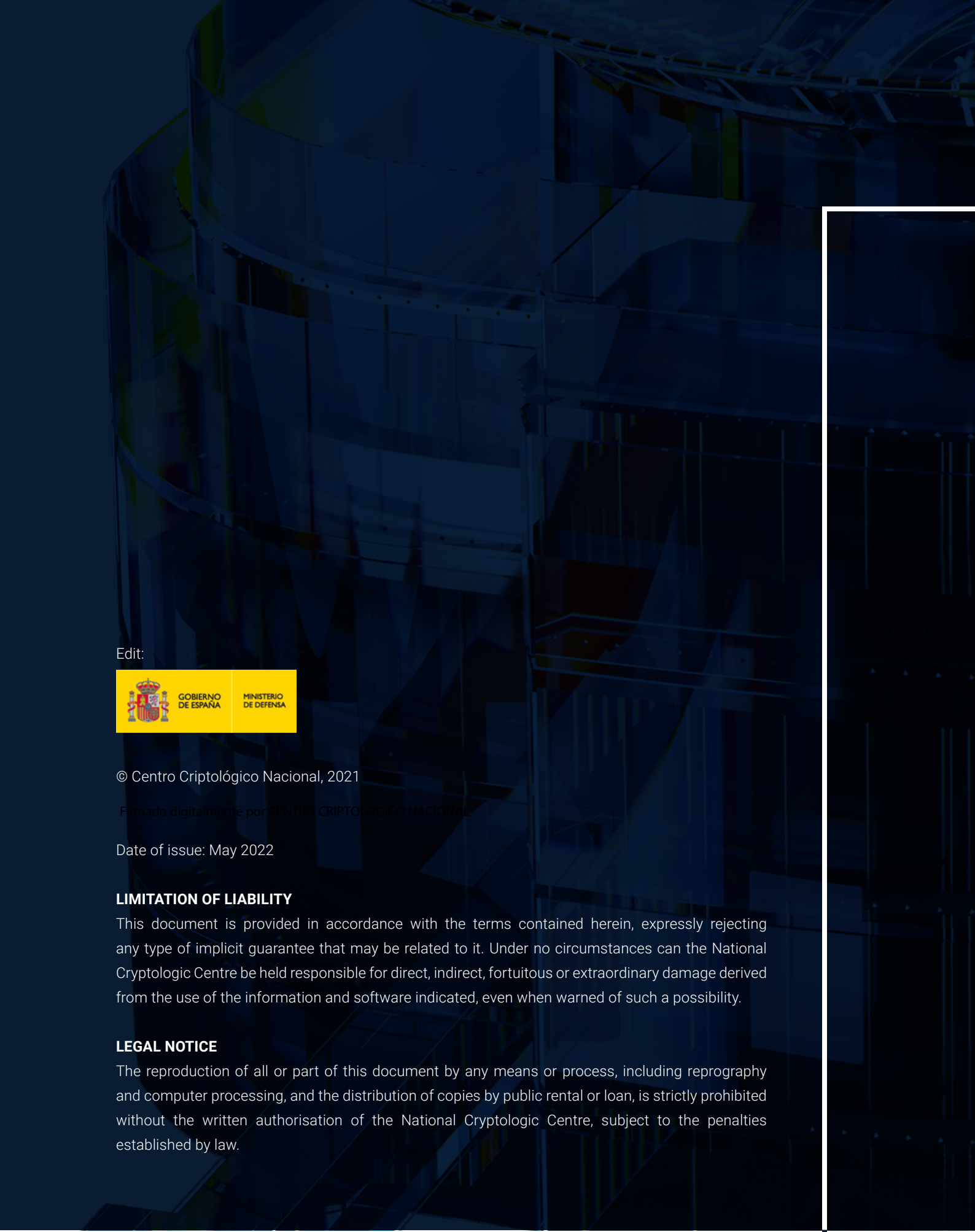
Security recommendations for Oracle Database 19C

BEST PRACTICES REPORT

MAY 2022

ccn-cert centro criptológico nacional

CCN centro criptológico nacional



Edit:

GOBIERNO DE ESPAÑA MINISTERIO DE DEFENSA

Firmado digitalmente por CENTRO CRIPTOLOGICO NACIONAL

Date of issue: May 2022

Index

1. About CCN-CERT, National Governmental CERT

The CCN-CERT is the Information Security Incident Response Capability of the National Cryptologic Center, CCN.

The **CCN-CERT** is the Information Security Incident Response Capacity of the National Cryptologic Centre, CCN, attached to the National Intelligence Centre, CNI. This service was created in 2006 as the **Spanish National Governmental CERT** and its functions are set out in Law 11/2002 regulating the CNI, RD 421/2004 regulating the CCN and in RD 3/2010, of 8 January, regulating the National Security Scheme (ENS), modified by RD 951/2015 of 23 October.

Its mission, therefore, is to **contribute to the improvement of Spanish cybersecurity**, being the national alert and response centre that cooperates and helps to **respond quickly** and efficiently to cyber-attacks and to actively face cyber-threats, including the coordination at state public level of the different existing Incident Response Capabilities or Cybersecurity Operations Centres.

All of this, with the ultimate **aim of achieving a more secure and reliable** cyberspace, preserving classified information (as stated in art. 4. F of Law 11/2002) and sensitive information, defending Spain's Technological Heritage, training expert personnel, applying security policies and procedures and using and developing the most appropriate technologies for this purpose.

In accordance with these regulations and Law 40/2015 on the Legal Regime of the Public Sector, the CCN-CERT is responsible for the management of cyber-incidents affecting any public body or company. In the case of critical public sector operators, cyber-incident management will be carried out by the CCN-CERT in coordination with the CNPIC.

2. Fundamentals of database security

Database management systems run on specific platforms and operating systems that provide them with the fundamental elements of communication and access.

The security model of a database management system, therefore, from a simplified point of view, can be said to be divided into these **two areas of action:**

The scope of the platform where the service runs

The environment and capabilities provided by the database manager itself

2. Fundamentals of database security

The Oracle 19c product is a generalist relational database manager, which means that it can be used in multiple environments and applications, and can be deployed on Unix, Linux and Microsoft Windows servers.

In all cases, it will be important not to lose sight of the security aspects that are configured at the operating system level, such as users, services, communications and protocols, as well as those that are configured in the Oracle 19c environment, such as authorisation processes and access control to the data residing in the different databases.

Authentication is the process by which **a system verifies the identity of a user.** In Oracle 19c, this process is performed outside the application environment, through an authentication module. By means of different modules that Oracle 19c incorporates, it is possible to make use of authentication protocols such as LDAP, OS, TNS, Kerberos, by SID or service name.

Authentication is the process by which a system verifies the identity of a user.

For activating and configuring the **different authentication methods,** see the guide:

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-authentication.html

Kerberos can be implemented as indicated via the following link:

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-kerberos-authentication.html

2. Fundamentals of database security

Security enforcement on the Oracle 19c database engine is part of a series of tasks that must be performed on an ongoing basis. Oracle is a well-known database engine and its user accounts, ports, file paths and default settings in an installation can pose a serious security threat to an organisation.

This new Oracle release offers several enhancements and features concerning automatic indexing through machine learning algorithms and adds a point of improvement on stability through Active Data Guard DML redirection in terms of backups.

Active Data Guard is a high availability architecture model that enables a high availability architecture as a data loss prevention model in synchronous or asynchronous mode. Oracle (Active) Data Guard capabilities in Oracle Database 19c further enhance its strategic goal of preventing data loss, providing high availability, eliminating risk, and increasing ROI by enabling highly functional active disaster recovery systems that are easy to deploy and manage. It achieves this by providing the management, monitoring and automation software infrastructure to create and maintain one or more synchronised standby databases that protect Oracle data from failures, data corruption, human error, and disasters.

Active Data Guard uses the simplicity of physical replication, with its integration with Oracle it provides unique isolation between primary and standby databases to offer the highest level of protection against data loss. Active Data Guard supports both synchronous (zero guaranteed data loss) and asynchronous (near zero data loss).

Active Data Guard uses the simplicity of physical replication, with its integration with Oracle providing unique isolation between the primary databases.

2. Fundamentals of database security

To maintain high availability for mission-critical applications, database administrators can choose manual or automatic failover in the event that, for any reason, the primary system becomes unavailable. Active Data Guard is a licensed option for Oracle Database Enterprise Edition. All capabilities that are explicitly named "Active Data Guard" require an Active Data Guard licence. All capabilities explicitly referred to as "Data Guard" are included with Oracle Enterprise Edition, no option license is required. Active Data Guard is a superset of Data Guard and inherits all Data Guard capabilities.

One of the big advantages of Active Data Guard 19c is the improved ability to perform intensive offline reads against standby applications. It is now also possible to issue occasional DML against the standby database, making it now a fully functional reporting database. This leverages the ROI as the primary database is used optimally and the resources of the disaster recovery system are used optimally.

More information about Oracle Active Data Guard 19c can be obtained from the following link:

Link: https://www.oracle.com/technetwork/database/availability/dg-adg-technical-overview-wp-5347548.pdf

In terms of traceability and the sudden growth that each of the Oracle tables may have depending on the needs of each organisation, hybrid partitioned tables have been implemented allowing the management of a table between partitions within the database and also outside the database, in the external case with read access.

Further information can be found at the following link:

Link: https://oracle-base.com/articles/19c/hybrid-partitioned-tables-19c

2. Fundamentals of database security

Authorisation is the process of determining whether an authenticated user has access to the information and permissions they are requesting. This process is carried out entirely within Oracle 19c, consulting the permissions associated with a specific identity. In this sense, there are different types of permissions that can be granted.

- **Primary permissions:** Those that are granted directly to the authorisation identifier.
- **Secondary permissions:** Those that are granted to groups and roles of which an authorisation identifier is a member.
- **Public permits:** Those that are granted to the entity PUBLIC.
- **Context-based permissions:** Those that are granted to a trusted context role.

These permissions can be granted to users at various levels or categories:

- **System level authorisation:** These are the authorities that perform administration tasks. There are several users with different roles.
- The SYS user is system administrator. Its password must be changed from the manufacturer's default password. It is not advisable to create objects within its schema.
- The SYSTEM user in charge of system control has the role of DBA and must also change his default password. In his schema, tables and administration views can be created.
- SYSBACKUP, SYSDG, SYSKM, and SYSRAC users are automatically created on installation to facilitate administration.
- The SYSBACKUP user facilitates Oracle Recovery Manager (RMAN) backup and recovery operations from either RMAN or SQL * Plus.
- The SYSDG user facilitates Data Guard operations. The user can perform operations with Data Guard Broker or with the DGMGRL command line interface.

- The SYSKM user facilitates the operations of the transparent data encryption keystore.
- The SYSRAC user facilitates Oracle Real Application Clusters (Oracle RAC) operations by connecting to the database through the Clusterware agent against Oracle RAC utilities such as SRVCTL.
- The SYSRAC administrative privilege cannot be granted to database users and is not supported in a password file. The SYSRAC administrative privilege is only used by the Oracle Clusterware Oracle agent to connect to the database using operating system authentication.

Users who have been granted the CREATE USER system privilege can create user accounts, including user accounts to be used as proxy users. Because the CREATE USER system privilege is a powerful privilege, a database administrator or security administrator is usually the only user who has this system privilege. If you want to create users that have the create user privilege, you can include the WITH ADMIN OPTION clause in the GRANT statement.

- **Database level authorisation:** Oracle 19c has 79 predefined roles during installation. The definition of these can be found in the following link:

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-privilege-and-role-authorization.html

Independently of these, other roles can be generated with the create role statement, which can then be assigned specific permissions. Therefore, an audit of permissions in the database must be dynamic and not restricted only to the roles and permissions generated in an installation. The vendor provides at least 30 role management views to facilitate these tasks. Authorities with grant and revoke privilege permissions can assign and revoke permissions to users and roles.

- **Object level authorisation:** Object level authorisation involves the checking of privileges when a specific operation is performed on a specific object.

Content-based authorisation: One way to authorise content-based access is through views. Views allow to control which columns or rows of a table can be read by specific users. On the other hand, Oracle, through Oracle Label Security, allows access control to specific (labelled) rows of a database. With Oracle Label Security in place, users with different levels of privileges are automatically granted (or excluded) the right to view or modify labelled rows of data. The Oracle Label Security Administrator's Guide describes how to use Oracle Label Security to protect sensitive data. It explains the basic concepts behind label-based security and provides examples to show how it is used.

Another important component in defining the security of a database manager is encryption, both for data in transit and data at rest. Oracle 19c offers different data encryption and transport options, which are discussed later in this document.

The information in this section can be completed from the following link.

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/admin/getting-started-with-database-administration.html

3. Secure database implementation

This section provides recommendations on the installation of the Oracle 19c product oriented to most of the possible use cases of each organisation depending on the system.

The commands described below shall be appropriate to the environment and system on which they are being executed, taking into account the minimum requirements set by the manufacturer.

The steps described below are intended to be repeated after performing a software upgrade on the Oracle 19c database engine.

To install Oracle 19c you must first install the necessary prerequisites using the following command:

```
dnf install and https://yum.oracle.com/repo/
OracleLinux/OL8/baseos/latest/x86_64/getPackage/
oracle-database-preinstall-19c-1.0-1.el8.x86_64.rpm
```

Upon executing the command, the system will produce an output window similar to the following image:

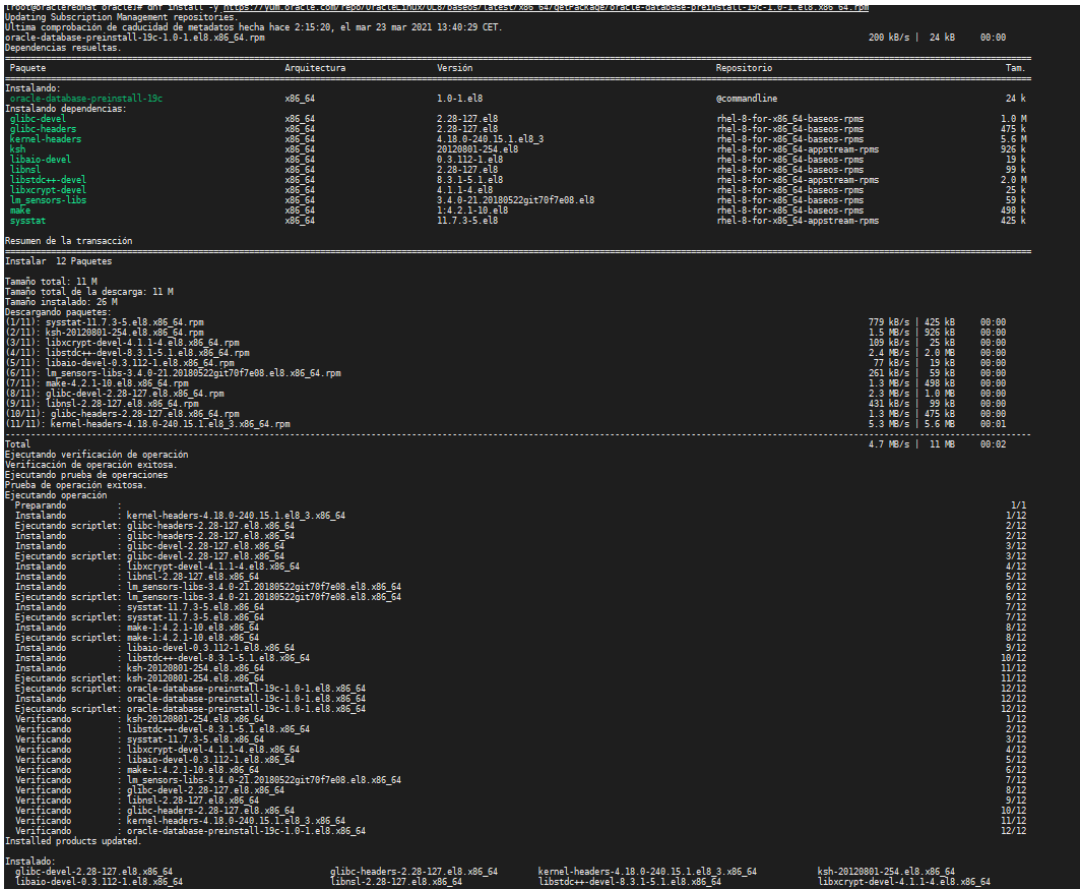


Illustration 1 - Installation Command Oracle Prerequisites

Once finished, you must download the Oracle installation software and install it with the following command:

```
rpm -i oracle-database-ee-19c-1.0-1.x86_64.rpm
```

Upon executing the command, the system will return information similar to the following:

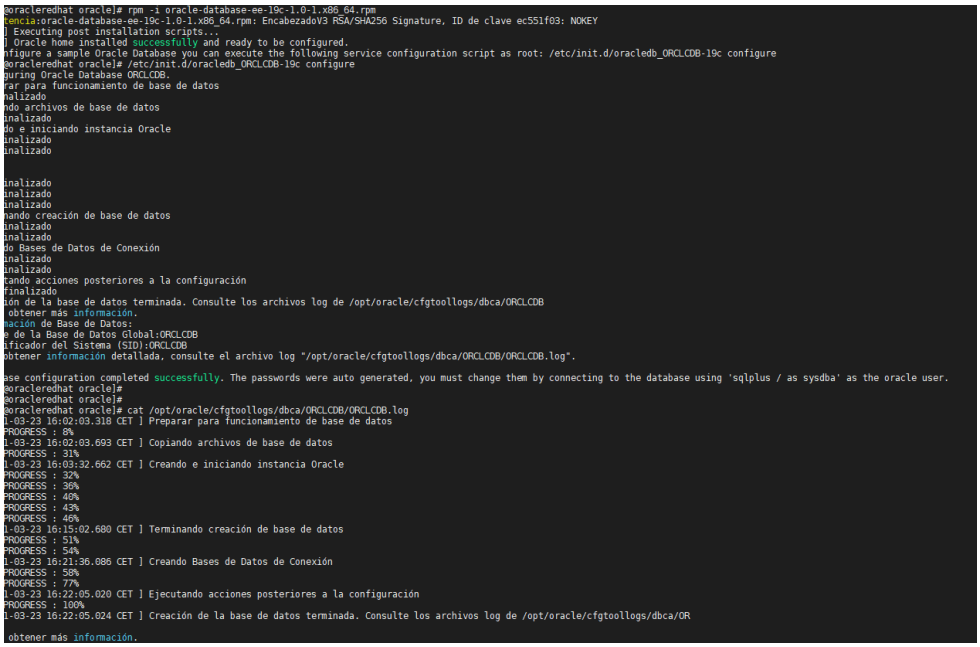


Illustration 2 - Oracle Installation Software command

Note: More information on installation can be found in the manufacturer's document titled **"database-installation-guide-linux.pdf".** Please note that some vendor documents may require login to your platform.

Due to the new Oracle release, there are 2 new options when creating a database during installation. Therefore, the following options should be taken into consideration depending on the needs of the organisation:

NON-CDB

Database similar to previous versions 9.x, 10.x or 11.x

CDB

Container database for the storage of pluggable databases.

This CDB database enables the "multitenant" option of version 19c, which makes it possible to create several "pluggable" databases on this container, sharing the metadata contained in the container database or "CDB".

The creation of a CBD is hardly different from the creation of another database in previous versions.

At start-up the DBCA offers the following options as shown below:

- **Create a CDB together with a pluggable database ("PDB").**
- **Create a CBD in advanced mode, which allows the creation of the empty CBD.**

By design, you can quickly connect a PDB to a CDB, disconnect the PDB from the CDB, and then connect this PDB to a different CDB. You can also clone PDBs as long as they are available.

Documentation on the different architectures can be found in the following links:

Link: https://docs.oracle.com/en/database/oracle/oracle-database/18/rilin/deciding-between-multitenant-container-databases-and-non-cdbs-in-oracle-rac.html

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/multi/introduction-to-the-multitenant-architecture.html

3. Secure database implementation

After installation of the product or patching of the product, the status of the solution should be checked and the manufacturer's documentation should be reviewed, as previously bastioned objects may need to be bastioned or new objects may exist after installation.

At the software level, the following compliance tasks must be performed on a regular basis:

- Keep the engine version up to date.
- Keep versions of any extra software to the engine up to date, for example Apex or any other product that may modify or incorporate database server objects. There are multiple products that can add roles, permissions, packages, etc. At this point the security of the database engine should be reviewed again.
- Verify that ORA_DBA user accounts are not root in the operating system.
- Review the vulnerabilities of each component belonging to the installation. Known vulnerabilities (CVEs) per component (CPE) can be consulted on portals such as NIST.
- In case vulnerabilities are published, and have not been fixed by Oracle, this should be reported to senior security managers.
- Clean temporary files after product or patch installation (TMP_DIR, TMPDIR, TEMP, TMP...).

4. Secure database configuration

The following are recommendations for strengthening the security of the Oracle 19c database after the installation process has been completed.

4.1 Access control

Designing appropriate access controls tailored to the needs of data exploitation by users and tools is essential to reduce the risks of exfiltration or unauthorised access. Most threats fall into this category and are minimised or eliminated by maintaining strict controls.

Access to an instance or a database requires the user to authenticate. Oracle provides different authentication protocols as discussed at the point 2 of the document.

It is recommended to use strong authentication mechanisms such as SERVER, LDAP or Kerberos and to avoid using CLIENT authentication, especially in environments where client security cannot be guaranteed.

It is recommended to follow the principle of least privilege, where only users are allowed to access the information and do the actions they really need to do, minimising the exposure surface.

Designing appropriate access controls tailored to the needs of data exploitation by users and tools is essential to reduce the risks of exfiltration or unauthorized access.

It is recommended to review and, if necessary, revoke permissions of users or groups that do not need them.

In scenarios where sensitive data is stored, it is recommended, in addition to **reviewing privileges**, to establish granular access controls such as cell, column or row, in order to prevent access to sensitive data from untrusted environments, see section 4.4.1 of this document to apply.

By default, a DBA has **access to any table** in his or her database instance. This is a risk, especially if the account has been breached or if these privileges are abused. It is recommended to revoke the DBA's data access privileges if he/she has no real need to access the data.

It is recommended to check that PUBLIC access has not been granted to any database.

An unauthorised user can access information residing in system tables if they have not been properly protected. It is recommended to review and protect important system tables and views such as the plsql code containers: **All_source, dba_source** or **ALL_OBJECTS, DBA_OBJECTS** objects.

It also recommends assigning privileges through a role model, avoiding direct assignment to users. Do not forget to subsequently assign roles to specific or concrete users who can identify who does what.

In addition, it is recommended to use operating system controls to prevent operating system administrators from gaining too much access.

On the other hand, **it is recommended to assign DBA permissions only through a role, and to control access to this role through trusted contexts.** This allows restricting access only to connections originating from trusted computers.

It is also recommended to revoke the privilege to create databases for all users except the DBA user.

The Listener is one of the components most likely to be susceptible to attacks, mainly distributed denial of service (DDoS) attacks. For this reason, the components of this service must be secured and audited.

It also recommends assigning privileges through a role model, avoiding direct assignment to users.

The following are **recommendations on the security configuration** of the service.

a. **Security measures must be applied on access to the service files: lsnrctl, listener.ora, sqlnet.ora** and **tnslnsr.** Both **lsnrctl** and **tnslnsr** are executables that must have 0700 permissions.

Based on these files, access to the service must be securely configured. The default SID name must be changed, allowing only local authentications for administration.

The following steps can be followed to configure these parameters:

```
LOCAL_OS_AUTHENTICATION_ = ON , ADMIN_RESTRICTIONS_
LISTENER=ON
```

The following audit commands can be run on the service to catch possible brute force attacks it may receive:

```
set current_listener <listener name>
set  log_directory  <oracle_home  path>/network/
admin
set log_file <sid name>.log
set log_status on
save_config
```

Service access permissions should also be checked, using a unique name for each service. Listener auditing should be enabled to check the values against the following flags to identify possible live attacks

Message
TNS-01169
TNS-01189
TNS-01190
TNS-12508
ORA-12525
ORA-28040
ORA-12170

b. **It is advisable to set up network packet logging.**

Within the file **"Listener.ora"** the following parameters must be set as at least the next log level:

- `"SEC_PROTOCOL_ERROR_TRACE_ACTION" for "TRACE", "LOG or "ALERT"`
- `"SEC_PROTOCOL_ERROR_FURTHER_ACTION" by "DROP,3".`

c. **The manufacturer's default ports must be edited and changed by editing the "Listener.ora" file or using the "Netmgr" utility.**

- Listener TNS default port 1521, 1522.
- Oracle Names Server default port 1575.
- Oracle Connection Manager default port 1630 - client connections
- Oracle Connection Manager default port 1830 - administrative connections.
- Port 2483 is the default port for TNS in TCP/IP protocol.
- TNS default port 2484 in TCP/IP protocol with SSL.

d. **The parameter "INBOUND_CONNECT_TIMEOUT" must be set to 60 in the ".ora" files to prevent DDoS attacks.**

e. **The entry in the "Sqlnet.ora" file of whitelists and blacklists of IPs and ranges with access to the server (Valid Node Checking) must be configured.**

You can take the following example values:

```
tcp.validnode_checking = yes
tcp.invited_nodes = (x.x.x.x | name, x.x.x.x | name)
tcp.excluded_nodes=( x.x.x.x | name, x.x.x.x | name)
```

f. **SQL traffic between clients and server must be encrypted.**

In the **"Sqlnet.ora"** file, the required entries must be configured as mandatory:

```
SQLNET.ENCRYPTION_SERVER = [accepted | rejected |
requested | required ]
SQLNET.ENCYRPTION_TYPES_SERVER = (algorithm name)
```

On the client side, the **required** entries must be configured as mandatory:

```
SQLNET.ENCRYPTION_CLIENT = [ accepted | rejected |
requested | required ]
SQLNET.ENCYRPTION_TYPES_CLIENT = ( algorithm name
)
```

The configuration can be verified with the command:

```
SELECT NETWORK_SERVICE_BANNER FROM V$SESSION_
CONNECT_INFO;
```

Note: More information on the above security settings can be found in the following links:

https://www.integrigy.com/files/Integrigy_Oracle_Listener_TNS_Security.pdf

https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/

https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/keeping-your-oracle-database-secure.html

4.2 Audit

Auditing is a fundamental component in strengthening the security of an IT environment, especially in multi-user environments, where there is a need to know the actions performed by each of the users.

Logging of unwanted actions or unauthorised access to data and subsequent analysis improves the levels of data access control and the prevention of unauthorised access, malicious access or misconfiguration.

Monitoring of individual user and application access, including system administration actions, can provide a historical record of activity on your database systems.

Oracle 19c audit generates and maintains audit evidence for a series of pre-defined database events. The records generated in an audit log file or table and their analysis can reveal patterns of usage that would identify system misuse. Once identified, actions can be taken to reduce or eliminate such system misuse. The create audit policy command can be used to generate the log of the event to be audited. You can also define actions to be audited on the object such as reading from a table or executing a function. You can review all auditable objects and on which actions in the link:

You can define actions to be audited on the object such as reading from a table or executing a function.

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/CREATE-AUDIT-POLICY-Unified-Auditing.html

The audit function allows auditing at instance level as well as at individual database level, with all activities being recorded independently in separate logs for each.

It should be noted that if you want to audit and/or bastion the accesses to the records of a table, you must verify that all accesses to views, materialised views, synonyms or possible file outputs via ETLs based on the records of that table are also audited and/or bastioned.

In addition, Oracle 19c incorporates filtering tools, security policies and audits on all incoming and outgoing requests to the database engine with "AVDF Oracle Audit Vault" and "Database Firewall". Policies must be generated using the WASS (Web Application Acceleration and Security Policy) before Oracle WAF rules can be created.

In case one of the databases is under continuous development, the application of the OSSA methodology for the application of security in databases under construction and testing should be considered.

Once the WASS policies have been configured, WAF rules must be created with the following parameters as standard recommendations:

- **AccessRules.** The **ALLOW, DETECT**, and **BLOCK** values of the WASS policy must be configured.
- **AddressRateLimiting.** Limiting on the entire number of requests for an IP address.
- **CachingRules.** Caching rules for accessing a web application.
- **Captchas.** Configuration of captchas to prevent access by bots.
- **CustomProtectionRules.** OCIDs blocking rules and allowed actions.

- **DeviceFingerprintChallenge.** Engine enumeration denial rules by bots using fingerprinting techniques.
- **GoodBots.** Whitelist of bots with access to the web server.
- **HumanInteractionChallenge.** List of human interactions such as mouse movements, reaction times, page scrolling, etc. to identify bots.
- **JsChallenge.** List of Javascript request configuration options for blocking bots.
- **Origin.** Container key within the Origins defined in the WASS Policy.
- **OriginGroups.** Origin group of the Origin object to be accessed defined in the WASS.
- **ProtectionRules.** List of protection rules and their description.
- **ProtectionSettings.** List of options to apply to the protectionRules.
- **ThreatFeeds.** Actions to be applied when malicious traffic is detected.
- **Whitelists.** Whitelist of IP addresses that can pass through the Firewall.

4.3 Communications protection measures

The manufacturer allows encrypting communications at the socket level or at the transport layer (TLS). The **advantages and disadvantages of each option** can be reviewed at the following link:

Link: https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-network-data-encryption-and-integrity.html

Oracle Database provides native data network encryption and integrity to ensure that data is secure as it travels over the network.

Oracle Database supports the Federal Information Processing Standard (FIPS), Advanced Encryption Standard (AES) encryption algorithm. Triple DES encryption is also possible.

The algorithms that the manufacturer marks for native network encryption as deprecated and should not be used are **DES, DES40, DES40, 3DES112, 3DES168, RC4_40, RC4_56, RC4_128 and RC4_256.** The algorithms enhanced in this release with patch 2118136.2 are: AES128, AES192 and AES256.

To increase the security of the native network encryption, the sql.ora files on the clients must be configured by removing the following entries if and only if they exist:

```
SQLNET.ENCRYPTION_TYPES_CLIENT

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT
```

The sql.ora files on the servers must be configured by deleting the following entries if and only if they exist:

```
SQLNET.ENCRYPTION_TYPES_SERVER

SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER
```

The sql.ora file must be configured on the server with the parameters:

```
SQLNET.ENCRYPTION_SERVER = REQUIRED

SQLNET.ENCRYPTION_TYPES_SERVER = (AES256)

SQLNET.CRYPTO_CHECKSUM_SERVER = REQUIRED

SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (SHA512)

SQLNET.ALLOW_WEAK_CRYPTO_CLIENTS = FALSE
```

Each client must be configured in the sql.ora file with the following entries:

```
SQLNET.ENCRYPTION_CLIENT = REQUIRED

SQLNET.ENCRYPTION_TYPES_CLIENT = (AES256)

SQLNET.CRYPTO_CHECKSUM_CLIENT = REQUIRED

SQLNET.CRYPTO_CHECKSUM_TYPES_CLIENT = (SHA512)

SQLNET.ALLOW_WEAK_CRYPTO = FALSE
```

Note: More information can be found in the following links:


https://ittutorial.org/oracle-19c-network-encryption/


https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/release-changes.html


https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/configuring-transparent-data-encryption.html


https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/configuring-network-data-encryption-and-integrity.html

4.4 Information protection measures

Information protection measures include both those that are configured or implemented in the database server environment as well as in the operating system environment running the server.

Oracle 19c allows you to generate encryption keys and encrypt your databases. It also allows you to encrypt specific objects such as tables, table columns or cells. It is the responsibility of the organisation to know and secure the most sensitive data. This depends on the information content of the different databases. Not all data has the same criticality and it is up to the organisation to first categorise the information, and then secure access to it according to the sensitivity of the data.

The administration views of encrypted objects as well as the permissions of these objects should be reviewed. Only administrators should be able to execute **"Select"** on these views. Permissions can be assigned with the command **"GRANT SELECT ON vista TO user";**

The views that will yield information on encrypted objects are:

View
ALL_ENCRYPTED_COLUMNS
DBA_ENCRYPTED_COLUMNS
USER_ENCRYPTED_COLUMNS
DBA_TABLESPACE_USAGE_METRICS
V$CLIENT_SECRETS
V$DATABASE_KEY_INFO
V$ENCRYPTED_TABLESPACES
V$ENCRYPTION_KEYS
V$ENCRYPTION_WALLET
V$WALLET

a. The option "Generate Encryption Keys" must be considered in the engine with the command:

```
ADMINISTER KEY MANAGEMENT CREATE KEY [USING TAG 'tag']
[FORCE KEYSTORE]
IDENTIFIED BY [EXTERNAL STORE | keystore_password]
[WITH BACKUP [USING 'backup_identifier']];
```

b. **Oracle databases must be encrypted with the following command.**

```
ALTER TABLESPACE SYSTEM ENCRYPTION Type ENCRYPT
OPTIONS;
```

c. **The key can be activated with the following command:**

```
ADMINISTER KEY MANAGEMENT CREATE KEY USING TAG
OPTIONS;
```

The database engine object dictionary is one of the common sources of attacks on the database engine. If an attacker gains access to the database engine, he could add or modify database objects. It is necessary to know which objects and with which code, if any, are legitimate database objects. It is also essential to take an inventory of the dates on which objects have been created and modified.

Oracle 19c has Machine Learning objects that automatically create indexes to optimise engine performance. Therefore, index objects should be excluded from the inventory analysis.

- **Control the inventory of objects. Whenever there is a change or upgrade from development to production, the object content of the engine, as well as its "PLSQL" code, must be documented and dated.**

- **The permissions of the engine synonyms must be checked. It is important to check the security of the created synonym.**

- **As in the previous point, the access and content of the materialised views must be secured.**

Oracle allows accesses to other database servers by linking them. The security of these accesses and their exposed data must be given the same consideration as that of the Oracle engine's own data.

Often applications generate temporary tables from the code. Therefore, the use of "Global Temporary" is recommended, so that access to this data is only possible from the active connection that generates it and the table is destroyed at the end of the execution of the "PLSQL" code.

Security should be reviewed for requests to incorporate Java classes or other objects other than those incorporated by the manufacturer.

Oracle allows the incorporation of non-engine classes that may compromise the security of the product.

The security of all "PLSQL" code container objects developed for the operation of applications such as stored procedures, functions and packages must be reviewed. Particularly sensitive are container objects (procedures, functions and packages) for dynamic SQL execution such as execute immediate. If these objects have not been parameterised correctly, they can be subject to dynamic SQL attacks.

The use of "Global Temporary" is recommended, so that access to this data is only possible from the active connection than the general one.

The code of all "PLSQL" code container objects developed for the operation of applications: stored procedures, functions and packages must be encrypted. In this way a user editing the object will not be able to see its contents. The **"all_source"** and/or **"dba_source"** views can be used to quickly view the code of these objects.

Default user accounts are a clear vector of attack on the solution. Therefore, they must meet certain security criteria to minimise their exposure and possible exploitation.

The security of user accounts is recommended to meet the following criteria.

Default user accounts are a clear attack vector on the solution.

- **Segregation of Privileges and Minimal Exposure: Permissions should only be given to objects to which access should be granted.**
- **Special care must be taken with privileges granted with the "ANY" clause which grants privileges to all objects of the same type. In Oracle 19c there are 148 possible object permission assignment statements that include such an ANY clause. In addition, there are 84 other possible generic permission assignment commands that grant authorisation to a set of objects.**

The following are the recommended user account password security guidelines:

- Contain at least **12 characters.**
- Include **capital letters,** at least two.
- Contain at least **two lower case letters.**
- Contain at least **two numbers.**
- Contain at least **two special characters.**

- **Do not contain the user's name.**
- Set the following parameters.
  - **PASSWORD_REUSE_TIME:** Number of days during which a password cannot be reused.
  - **PASSWORD_REUSE_MAX:** Number of passwords that must be used before the first password can be reused.
- In addition to the passwords, it is recommended to configure the following elements:
  - **PASSWORD_LIFE_TIME:** The parameter defining the number of days before the password expires.
  - **PASSWORD_GRACE_TIME:** The parameter defining the maximum number of days after password expiry that the user has to change his password on expiry before all connections are refused.
- Account blocking. It is recommended to configure the variables:
  - **FAILED_LOGIN_ATTEMPTS:** Number of failed login attempts allowed before the user account is locked out.
  - **PASSWORD_LOCK_TIME:** Number of days an account will be locked after a series of failed login attempts.
- It is of particular interest to limit the account session time for non-server accounts to approximately 90 minutes.

- Non-application server user accounts may be blocked after **two unsuccessful attempts.**

- Logout time for **inactivity 20 minutes for non-application** server accounts.

- **Failed login** attempts before account lockout **equal to or less than 6**.

- Maximum **lifetime of a password** before being forced to change it **180 days.**

- Modify **SYS account:**

  This account can perform all administrative functions. All base (underlying) tables and views of the database data dictionary are stored in the SYS schema. These base tables and views are critical to the operation of Oracle Database.

  To maintain the integrity of the data dictionary, the SYS schema tables are only manipulated by the database.

  On the other hand, they must never be modified by any user or database administrator. No tables should be created in the SYS schema. The SYS user is granted the SYSDBA privilege, which allows the user to perform high-level administrative tasks, such as backup and recovery.

  Passwords can be changed with the following command:

```
ALTER USER SYS IDENTIFIED BY "new password";
```

Note: New passwords must meet the minimum complexity requirements.

Modify **SYSTEM account**

This account can perform all administrative functions except backup and recovery, and database update.

It is true that this account can be used to perform day-to-day administrative tasks, but Oracle strongly recommends the creation of named user accounts to administer the Oracle database to allow tracking of database activity.

You can change the password of the SYSTEM user from "SQLPLUS" as follows:

```
ALTER USER SYSTEM IDENTIFIED BY "new password ";
```

Note: New passwords must meet the minimum complexity requirements.

It is recommended to set up specific user accounts for the application servers.

User accounts must be nominative in order to guarantee the traceability of the different actions executed in the engine. Generic accounts associated with the different roles should not be used, but rather accounts that uniquely identify the author of any change.

It is desirable to have a user certificate for each account with access to the engine.

Establishing a two-factor authentication to the database engine for application server accounts, with logins such as Google Authenticator or other social networks (Social Sign-In Authentication), is recommended for application servers where an undetermined number of users will connect.

4. Secure database configuration

Often after performing an installation, databases to be exploited are restored and scripts are executed that can modify permissions associated with user accounts, roles, read permissions, modification, deletion of objects, etc.

Therefore, before and after restoring a database to the engine, the following dictionary views should be checked.

- All existing roles **"DBA_ROLES"** must be checked for new roles.
- Users with association to these **"DBA_TAB_PRIVS"** roles must be checked.
- System privileges associated with system roles and their associated accounts **"DBA_SYS_PRIVS"** must be checked.
- **APEX** users should be taken into account.

  Note: APEX is the web interface for managing workspaces. An "ADMIN" user is generated with the same password as the system account. The password of this account must be reviewed and changed according to the minimum complexity requirements.
- User accounts must be associated with types or roles. The minimum roles or account types defined by the manufacturer are:
- **Regular database users:** They are usually restricted to their schema containing their tables, views, indexes and stored procedures. If hackers hack into their accounts, they could not only view/update data within the user's schema, but also access objects in other schemas that the user may be authorised to access.

Application accounts: These are the database accounts that are used to run your applications, both commercial and homegrown. These accounts are similar to your regular database user accounts, but since applications must run 24/7, their passwords are often stored on multiple middle-tier servers. Compromise on these database accounts can result in data loss for the entire application, including end-user data.

Application administrators: These accounts are used to administer, patch and update your application, and therefore have full access to all data and stored procedures used for the application.

Data analysts or business intelligence users: These users typically have unrestricted read access to the application schema without going through application-level access controls.

Database Administrators (DBAs): They are responsible for a wide variety of database tasks including performance management, diagnostics and tuning, upgrading and patching, database start-up and shutdown, and database backup. Their highly privileged database access also gives them access to any sensitive data contained in the database (personal records, health records, corporate financial records, etc.) although such access is not required to perform DBA tasks. Database administrators have access to account management and are therefore often trusted by their organisations. These user accounts are often the target of attacks.

Security administrators: Many organisations have specialised database administrators who also have the responsibility of security administrators, including user account management, encryption key management and audit management.

Oracle recommends that under no circumstances should an account be created that is associated with both the DBA role and the Security Administration role. **Two different nominative accounts** should be generated if the credentials of both roles are to be given to the same natural person to improve the management of role segregation.

All accounts must be assigned by default to workspaces other than "SYSTEM".

- User accounts assigned to application servers must **not have quotas.**
- Access to objects owned by **"SYS"** (system administrator objects), **"DBA_"** (database administrator objects), **"USER_"** (user role and permission tables) must be checked and only accounts belonging to **"SYS"** or **"DBA"** profiles must have access to these objects. If account access to a particular object is subsequently required, this should be analysed and documented.
- The role permissions on the catalogue **"SELECT_CATALOG_ROLE", "EXECUTE_CATALOG_ROLE", "DELETE_CATALOG_ROLE", "RECOVERY_CATALOG_OWNER"** need to be revised.

4.4.1 Row and column access control

Sensitive data can be encrypted at record, column, row and even cell level.

It is advisable to encrypt the tables and/or columns with the most sensitive data.

a. An encrypted column can be generated with the command **"ENCRYPT"** in the "ddl" of the table creation as in the following example:

```
CREATE TABLE table_name
  (campo_a VARCHAR2(11),
  Encryption_field VARCHAR2(16) ENCRYPT NO SALT);
```

b. The procedures “packages”, “functions”, “all Source” can be encrypted. This makes it impossible for a person with access to the code of the “PLSQL” object to see its code.

The following command can be used:

```
wrap iname=input_file [ oname=output_file ]
```

c. Consider the option of encrypting disks, operating system level partitions, with the option of encrypting the data of a particularly sensitive cell with the utility **“DBMS_CRYPTO.SQL”** with the following piece of code.

```
DECLARE
         input_string           VARCHAR2(16)  :=
'tigertigertigert';
    raw_input        RAW(128) :=
UTL_RAW.CAST_TO_RAW(CONVERT(input_
string,'AL32UTF8','US7ASCII'));
    key_string       VARCHAR2(8)  := 'scottsco';
    raw_key          RAW(128) :=
UTL_RAW.CAST_TO_RAW(CONVERT(key_
string,'AL32UTF8','US7ASCII'));
    encrypted_raw    RAW(2048);
    encrypted_string VARCHAR2(2048);
    decrypted_raw    RAW(2048);
    decrypted_string VARCHAR2(2048);
-- Begin testing Encryption:
BEGIN
          dbms_output.put_line('>  Input  String
: ' ||
            CONVERT(UTL_RAW.CAST_TO_VARCHAR2(raw_
input),'US7ASCII','AL32UTF8'));
     dbms_output.put_line('> ========= BEGIN TEST
Encrypt =========');
    encrypted_raw := dbms_crypto.Encrypt(
        src => raw_input,
        typ => DBMS_CRYPTO.DES_CBC_PKCS5,
```

```
        key => raw_key);
           dbms_output.put_line('> Encrypted hex
value              : ' ||
          rawtohex(UTL_RAW.CAST_TO_RAW(encrypted_
raw)));
decrypted_raw := dbms_crypto.Decrypt(
        src => encrypted_raw,
        typ => DBMS_CRYPTO.DES_CBC_PKCS5,
        key => raw_key);
    decrypted_string :=
      CONVERT(UTL_RAW.CAST_TO_VARCHAR2(decrypted_
raw),'US7ASCII','AL32UTF8');
dbms_output.put_line('> Decrypted string output
: ' ||
        decrypted_string);
if input_string = decrypted_string THEN
     dbms_output.put_line('> String DES Encyption
and Decryption successful');
END if;
dbms_output.put_line('');
dbms_output.put_line('> ========= BEGIN TEST Hash
=========');
    encrypted_raw := dbms_crypto.Hash(
        src => raw_input,
        typ => DBMS_CRYPTO.HASH_SH1);
dbms_output.put_line('> Hash value of input string
: ' ||
          rawtohex(UTL_RAW.CAST_TO_RAW(encrypted_
raw)));
dbms_output.put_line('> ========= BEGIN TEST Mac
=========');
    encrypted_raw := dbms_crypto.Mac(
        src => raw_input,
        typ => DBMS_CRYPTO.HMAC_MD5,
        key => raw_key);
dbms_output.put_line('>  Message  Authentication
Code      : ' ||
          rawtohex(UTL_RAW.CAST_TO_RAW(encrypted_
raw)));
dbms_output.put_line('');
dbms_output.put_line('> End of DBMS_CRYPTO tests
');
END;
/
```

4.4.2 Label based access control

Signature-based security has been implemented for LOB locators. LOB CLOB, NLOB or BLOB data types are used to store files or large text fields of up to 4GB capacity.

LOB CLOB, NLOB or BLOB data types are used to store files or large text fields.

Emphasis should be put on how to store the LOB signing key in encrypted format, the database or PDB must have an open TDE keystore. From this version onwards, it is allowed to configure signature-based security for large object locators.

These LOB signing keys can be encrypted with the following command:

```
ALTER DATABASE DICTIONARY ENCRYPT CREDENTIALS
```

The TDE algorithms allowed in Oracle 19c are:

- **Advanced Encryption Standard (AES) 128, 192, 256 bits**
- **Triple Data Encryption Standard (TDES) 168 bits.**

Note: Further information can be found at the following link:

https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/configuring-transparent-data-encryption.html

Those security measures that privilege scanning has inherited from previous versions.

Note: For more information on privilege scanning security, visit the Oracle Database Vault and Oracle Database Security Guide.

https://docs.oracle.com/en/database/oracle/oracle-database/19/dvadm/index.html

https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/index.html

It is recommended that roles be established to grant and renew administrative privileges from any scheme. SYSOPER and SYSBACKUP groups can be segmented to schemes defined by the administrator.

Simple Authentication and Security Layer (SASL) and Transport Layer Security (TLS) bindings should be supported for Microsoft Active Directory connections.

Continuing with Oracle connectivity, this product version would require Oracle Native Encryption and SSL authentication for different users connected simultaneously.

It is recommended to make use of the new support for partial DN (domain name) matching based on host names for server certificate matching thus adding a two-factor authentication between client and server.

On the other hand, auditing options will be established on top-level SQL statements. Such a unified auditing top-level statement feature corresponds to an audit on those object dictionary modifications executed by users, thus allowing auditing of top-level user (or direct user) activities in the database, but without collecting audit data on indirect user activity. This adds a further layer of security when looking for traces of possible incidents.

In this product version, Oracle Native Encryption and SSL authentication would be required for the different users connected simultaneously.

Note: Further information can be found in the security guides linked below:

https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/

https://docs.oracle.com/en/database/oracle/oracle-database/19/dbseg/

Depending on the data stored in each engine's database, Oracle 19c has auto-analysis services based on artificial intelligence and data patterns associated with the regional configuration of the installation. The tool searches for data such as names, credit cards, salaries, amounts, personal addresses, etc. and allows secure access to them.

Many times, database administrators do not have the necessary information about which data is the most sensitive in each database to properly secure the information. The Oracle 19c "Label Security" component enables row-level data classification and provides access mediation, ready to use based on data classification and user label authorisation or security authorisation.

Oracle Label Security" software is installed by default, but is not automatically enabled. You can enable Oracle Label Security in SQL* Plus or by using the Oracle Database Configuration Assistant (DBCA).

The default administrator of Oracle Label Security is the user **"LBACSYS"**. To administer "Oracle Label Security", you can use a set of "PL/SQL" packages and separate functions at the command line level or "Oracle Enterprise Manager Cloud Control". For more information about "Oracle Label Security" policies, you can refer to **"ALL_SA_*"**, **"DBA_SA_*"**, **"SA_*"** or **"USER_SA_".**

The objects and utilities for categorising the data are as follows:“

Package	Purpose
SA_SYDBA	To create, modify, and delete Oracle tag security policies.
SA_COMPONENTS	To define the policy levels, compartments and groups.
SA_LABEL_ADMIN	To perform standard tag policy administrative functions, such as tag creation.
SA_POLICY_ADMIN	To apply policies to schemas and tables.
SA_USER_ADMIN	Manage user authorisations for levels, shares and groups, as well as program unit privileges. Also to manage user privileges.
SA_AUDIT_ADMIN	Manage user authorisations for levels, shares and groups, as well as program unit privileges. Also to manage user privileges.
SA_SESSION	Manage user authorisations for levels, shares and groups, as well as program unit privileges. Also to manage user privileges.
SA_UTL	To configure options for auditing administrative tasks and use of privileges.

“Enterprise Manager provides a graphical environment for the discovery and categorisation of sensitive data.

The information contained in databases must be categorised and documented regardless of the level of security settings applied. It must be known what information is stored where and how sensitive it is.

4.5 Backup policies

Sometimes a poor backup protection policy allows unauthorised access to data that is no longer protected by server security.

If data stored in backups are left unprotected, they can be accessed directly from the backup service.

It is recommended to encrypt all backup files and archive images, regardless of the medium on which they are stored.

It is recommended to ensure that the restoration of any backup should require controlled access to the encryption key and should be audited, both the access and the restoration itself.

The backup manufacturer's recommended practices should be maintained.

Regular backups are recommended. At least one incremental backup should be generated daily and kept for seven days. A weekly incremental backup should also be generated on Sunday and kept for four weeks. An incremental backup must also be generated every first day of the month, with the last twelve months retained. An annual backup must also be generated and retained for five years.

Backups must be stored at locations other than the physical location of the production server.

Periodic recovery testing is recommended as well as periodic disaster recovery drills.

A daily incremental backup should be generated and retained for seven days.

5. Other considerations

The following are some general good practices regardless of the product version. The manufacturer incorporates product documentation that has been referenced in this document and complements the information in this document. Some of the manufacturer's functionality may require the purchase of separate licenses from the database engine itself. The screenshots incorporated in this document were taken on a Rhell 8 operating system, therefore, the on-screen output of some of the steps may not correspond exactly to that of another operating system version.

It is recommended to activate the Active Data Guard redirection.

Oracle 19c incorporates this extra functionality as a disaster recovery mirrored database. In addition, the mirrored database can be used for read-mode data mining such as report mining or other processes with read access and write denial. Read transactions can be redirected to the mirror. This allows to have a real-time backup in a different physical location than the original database.

Oracle 19c incorporates hybrid partitioning tables that can physically reside in physically different locations from the rest of the tables. Hybrid partitioning must be applied to those tables that must reside in a physical location by applying higher security settings on the physical accesses where these tables reside.

It is recommended to store data on redundant RAID 1 or 5 disk systems for example.

It is recommended to replicate log and redo log files in physically different locations. These are files that store transaction log changes that are particularly sensitive to data change analysis.

In case one of the databases is under continuous development, the application of the OSSA methodology for the application of security in databases under construction and testing should be considered.

- **It is recommended to generate alarms (IAM policies) of consumption and utilisation of the DB engine.**
- **It is recommended to encrypt disks at controller level.**
- **It is recommended to document the procedures for changes in the DB engine as well as the different administration tasks.**
- **It is recommended to set up high availability clusters.**
- **It is recommended that old or damaged disks that have contained critical information should be magnetically erased before they are finally discarded.**
- **It is recommended to check the permissions of the database engine files and backup paths.**

6. Glossary

Authentication: is the process by which a system verifies the identity of a user. In Oracle 19c, this process is carried out outside the application environment, through an authentication module. By means of different modules that Oracle incorporates, it is possible to make use of authentication protocols such as LDAP, OS, TNS or Kerberos. User authentication is usually performed by the operating system or by an external server.

Authorisation: is the process of determining whether an authenticated user has access to the information and permissions being requested. This process is carried out entirely within Oracle 19c, consulting the permissions associated with a specific identity.

Oracle 19c Native Encryption: Oracle 19c Native Encryption provides built-in encryption capability to protect database backup images and key database files from unauthorised access while on external storage media. Encryption is a key component of offline data protection.

TLS: Transport Layer Security is a communications protocol whose main purpose is to provide privacy and data integrity between two communicating applications. The protocol is composed of two layers: the TLS registration protocol and the TLS handshake protocol. During TLS negotiation, a public key algorithm is used to securely exchange digital signatures and encryption keys between a client and a server. The identity information and the key are used to establish a secure connection for the session between the client and the server. Once the secure session is established, the data transmission between the client and the server is encrypted using a symmetric algorithm, such as AES.

Active Data Guard: is one of Oracle's DB replication solutions. Oracle Active Data Guard is an evolution of the previous one with improvements in availability, performance and protection.

Oracle Label Security: is a functionality that logs and enforces data access permissions based on project codes, regions or data classifications. These controls reduce the risk of unauthorised access to sensitive data and help demonstrate regulatory compliance.

6. Glossary

AVDF: Oracle Audit Vault and Database Firewall is a database activity monitoring and filtering solution. It incorporates audit data collection agents, database firewall, analysis and reporting tools.

Global Temporary: Oracle table type whose temporariness can be defined at transaction level (data exists while the transaction is being performed) or at session level (data exists for the duration of the session). The data in a temporary table is unique to the Oracle session that is using it.

Machine Learning: Oracle Machine Learning is a functionality offered in the vendor's SQL Developer product that discovers patterns and can provide insights into stored data.

OUI: Oracle Universal Installer. Oracle Universal Installer, tool for installing Oracle software.

DBCA: DataBase Computer Assistant. Software that facilitates the creation of Oracle databases. Requires Oracle database management software to be installed.

DBUA: DataBase Computer Assistant. Assistant to update databases.
CDB: Container database of objects whether schemas, other schema objects or other objects.

PDB: Pluggable database that behaves as a container within the CDB architecture, it composes a collection of objects independent of other pdbs with their own datafiles.

OPatch: Java-based utility that allows the application and rollback of patches to Oracle software.

Oracle Enterprise Manager: is a management platform that provides a single dashboard for managing all Oracle databases.

SSH: Secure Shell (SSH) is a protocol for secure remote login and other secure network services over an insecure network. SSH can be used as the basis for a number of secure network services as it provides robust encryption, server authentication and integrity protection. It also provides data compression. SSH is used during installation to configure cluster member nodes, and SSH is used after installation by configuration wizards, Oracle Enterprise Manager, Opatch and other features.

6. Glossary

RCAC: Row and Column Access Control. It allows access to a table to be controlled at row level, column level or both and can be used to complement the table privilege model, ensuring that information is adequately protected and that users only have access to the subset of data that is required to perform their job tasks and comply with specific rules and regulations.

LBAC: Label Based Access Control. It is a security model that is primarily intended for government applications or applications with known classification grades, as it requires data and users to be classified with a fixed set of rules that are implemented.

DBA: Database Administrator.

MV: Materialized View. A materialised view in Oracle is a database object that contains the results of a query. They are local copies of remotely located data or are used to create summary tables based on aggregations of data from a table. Materialised views, which store data based on remote tables, are also known as snapshots, the database engine may return data from a materialised view to improve performance. The data consists of pre-computed results from the tables specified in the materialised view definition.

FIPS: Federal Information Processing Standards. Federal Information Processing Standards (FIPS) Publication 140-2 is a U.S. government standard that defines the minimum security requirements for cryptographic modules in information technology products, as defined in Section 5131 of the Information Technology Management Reform Act of 1996.

7. Summary table of security enhancement measures

FIELD	NUM.	MEASURE	MOTIVE
SECURE IMPLEMENTATION	1	On Unix or Linux systems, it is recommended to specify different user names than those created by default.	Avoid using default names to plan attacks on the database.
	2	On Windows systems, it is recommended to change this default setting and specify different user names for each role.	Avoid using default names to plan attacks on the database.
	3	It is recommended to create instance-specific instance owner user IDs for each instance, adding it only as a member of the instance owner group and not using it in any other group.	It allows greater control over the number of users and groups that can modify the instance.
	4	During installation, it is recommended to use strong passwords that comply with the organisation's security policies.	Minimises the possibility of brute force attacks.

7. Summary table of security enhancement measures

FIELD	NUM.	MEASURE	MOTIVE
ACCESS CONTROL	5	It is recommended to use strong authentication and communication mechanisms such as SERVER, LDAP TLS or Kerberos and to avoid using CLIENT authentication, especially in environments where client security cannot be guaranteed.	Improve the security and reliability of authentication mechanisms.
	6	It is recommended to follow the principle of least privilege, where only users are allowed to access the information and do the actions they really need.	Minimise the exposure surface.
	7	It is recommended to review and, if necessary, revoke permissions of users or groups that do not need them.	Minimise the exposure surface.
	8	In scenarios where sensitive data is stored, it is recommended, in addition to reviewing privileges, to establish granular access controls.	Prevent access to sensitive roles from untrusted environments.
	9	It is recommended to revoke the DBA's data access privileges if he/she has no real need to access the data.	By default, a DBA has access to any table in his or her database instance. This poses a risk, especially if the account has been breached or if these privileges are abused.
	10	It is recommended to check that PUBLIC access has not been granted to any database.	Minimise the exposure surface.
	11	It is recommended to review and protect important system tables and views such as ALL_OBJECTS, ALL_SOURCE.	An unauthorised user can access information residing in system tables if they have not been adequately protected.
	12	It is recommended to assign privileges through a role model, avoiding direct assignment to users.	Improve control and maintenance of access privileges.
	13	It is recommended to use the access controls of the operating system.	Prevent operating system administrators from gaining too much access.
	14	It is recommended to assign DBA permissions only through a role, and to control access to this role through trust contexts.	Allows to restrict access only to connections originating from trusted computers.
	15	It is recommended to revoke the privilege to create databases for all users except the DBA.	Minimise the exposure surface.

7. Summary table of security enhancement measures

FIELD	NUM.	MEASURE	MOTIVE
AUDIT	16	It is recommended to review the audit event logging needs and to select only those events that are important for the organisation or those that are related to the security of the system.	Control the audit information generated, avoiding irrelevant data and storage problems that may lead to loss of relevant evidence.
	17	It is recommended to create an AUDITOR role and grant the necessary privileges to read and manage audit events.	Control who can access audit information and how.
	18	It is recommended to control access to the AUDITOR role through trusted contexts.	Allows to restrict access only to connections originating from trusted computers.
	19	It is recommended that the generated audit files should not be copied, modified or deleted directly by the operating system administrator or by any other unauthorised user of the platform.	Prevent exfiltration of data or access to sensitive audit information by bypassing database security mechanisms.
	20	It is recommended to make use of a centralised audit trail service.	Unification of different audit sources, facilitating log correlation and avoiding loss or manipulation of evidence.
	21	It is recommended to encrypt the authoring records stored on disk (data at rest), both on the database server and on the log centraliser service, if one is available.	Prevent exfiltration of data or access to sensitive audit information by bypassing database security mechanisms.
	22	It is recommended to audit all DBA actions.	Maintain an audit trail of administrative actions that may compromise the system.
	23	It is recommended to audit user access, in particular those who have access to sensitive data.	Maintain an audit trail of user actions.
	24	It is recommended to audit all accesses to important tables.	Maintain an audit trail of actions that may compromise the system.
	25	It is recommended to audit the SYS schema objects.	Maintaining an audit trail of these objects will allow you to keep track of changes made to objects such as tables, views, indexes, etc....
	26	It is recommended to audit all attempts to create databases.	Maintain an audit trail of administrative actions that may compromise the system.

7. Summary table of security enhancement measures

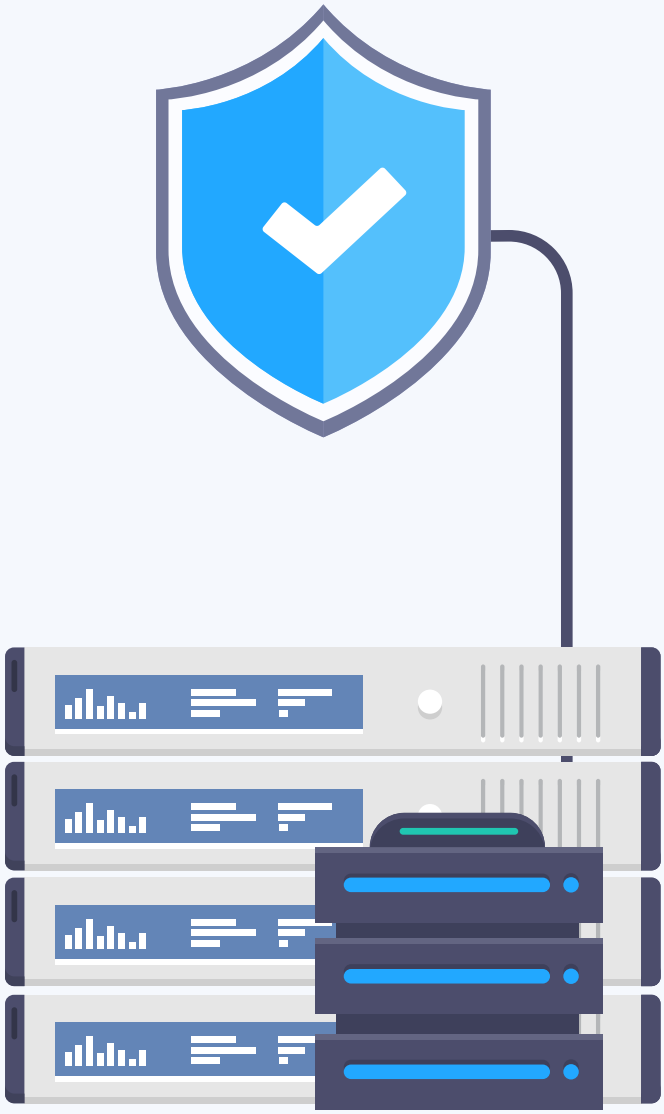
FIELD	NUM.	MEASURE	MOTIVE
COMMUNICATIONS PROTECTION	27	It is recommended to make use of encryption with the secure algorithms exposed in the communications layer.	Prevent data capture in transit through the network.
	28	It is recommended not to encrypt with algorithms marked as obsolete by the manufacturer.	The manufacturer marks the following algorithms as obsolete: DES, DES40, 3DES112, 3DES168, RC4_40, RC4_56, RC4_128 and RC4_256, and they should not be used.
	29	It is recommended to use robust cipher algorithm sets endorsed by the National Cryptologic Centre.	Prevent exploitation of vulnerabilities in weak or obsolete algorithms.
	30	It is recommended to verify that you have a recent version of Oracle 19c.	Older versions make use of weak or vulnerable encryption algorithms that should not be used.
	31	It is recommended to install the advanced algorithm support patch from this document.	Oracle 19c have the patch to 2118136.2install advanced ciphers.
	32	To enable TLS 1.2 (SSL 3.0) in Oracle 19c, it is recommended to use certificates issued by a trusted certificate authority.	It allows to correctly validate the certificate issuing chain and therefore its trust.
	33	It is recommended to review and configure the ports used by all server instances by using the services file to map the service name in the server's database administrator configuration file to its port number.	Minimise the exposure surface, enabling only the necessary communication ports.
	34	It is recommended to configure the Oracle WAF.	El Firewall propio del fabricante le permite establecer reglas específicas adaptadas cada entorno.
PROTECTION OF INFORMATION	35	It is recommended to design and make use of granular record or column access policies (RCAC) in those environments where there are regulations or standards to comply with and access to data has to be made according to the context of the requester.	Comply with the "need to know" principle.

7. Summary table of security enhancement measures

FIELD	NUM.	MEASURE	MOTIVE
PROTECTION OF INFORMATION	36	It is recommended to use LBAC at the registry level when handling sensitive or classified information related to government entities.	Comply with the “need to know” principle.
	37	It is recommended to use LBAC at registry level when the following statements are true: – The degree of classification of the data is known. – The classification of the data can be represented by one or more LBAC security labels. – Authorisation rules can be linked to the components of the security label.	Comply with the “need to know” principle.
	38	LBAC at spinal level is recommended when: – It is required to protect sensitive columns from unauthorised access to the table owners or even the DBA. – It is required to protect entire tables from unauthorised access to the table owners or even the DBA.	Comply with the “need to know” principle.
	39	Regardless of the access controls implemented, it is recommended to make use of encryption at rest mechanisms for data, tables, audit files and backup files at the operating system level.	Prevent unauthorised access to sensitive information outside the scope of protection of the database.
BACKUP	40	It is recommended to encrypt all backup files and archive images, regardless of the medium on which they are stored.	Prevent unauthorised access to backups.
	41	It is recommended to ensure that the restoration of any backup should require controlled access to the encryption key and should be audited, both the access and the restoration itself.	Prevent unauthorised access to backups and log any access through auditing.

centro criptológico nacional

centro criptológico nacional



www.ccn.cni.es

www.ccn-cert.cni.es

oc.ccn.cni.es