18

Oracle Database Security

Objectives

After completing this lesson you should be able to do the following:

- Apply the principal of least privilege
- Manage default user accounts
- Implement standard password security features
- Audit database activity

Database Security

A secure system ensures the confidentiality of the data it contains. There are several aspects of security:

- Restricting access to data and services
- Authenticating users
- Monitoring for suspicious activity

Authenticating User

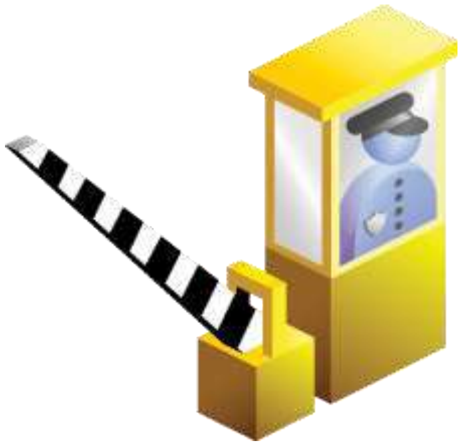
- The most basic form of user authentication is by challenging the user to provide something they know such as a password.
- An even stronger form of authentication is to identify the user through a unique biometric characteristic such as a fingerprint, iris scan, bone structure patterns, and so on.
- Oracle supports advanced authentication techniques such as token-, biometric-, and certificate-based identification through the Advanced Security Option.

Monitoring for Suspicious Activity

- Even authorized, authenticated users can sometimes compromise your system.
- Identifying unusual database activity such as an employee who suddenly begins querying large amounts of credit card information, can be the first step to detecting information theft.
- Oracle provides a rich set of auditing tools to track user activity and identify suspicious trends.

Apply the Principle of Least Privilege

- Protect the data dictionary
- Revoke unnecessary privileges from `PUBLIC`
- Restrict the directories accessible by users
- Limit users with administrative privileges
- Restrict remote database authentication



ORACLE

Protect the Data Dictionary

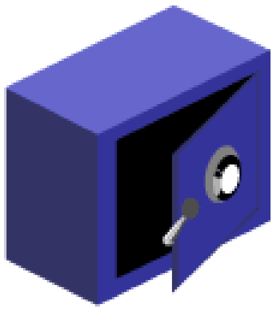
- Protect the data dictionary by ensuring the following initialization parameter is set to `FALSE`:

```
O7_DICTIONARY_ACCESSIBILITY = FALSE
```

- This configuration prevents users with `ANY TABLE` system privileges from accessing data dictionary base tables.
- The default value of this parameter is `FALSE`. If you find it set to `TRUE`, ensure there is a good business reason.

Revoke Unnecessary Privileges from `PUBLIC`

- Revoke all unnecessary privileges and roles from the database server user group `PUBLIC`.
- Many built-in packages grant `EXECUTE` to `PUBLIC`.
- Execute on the following packages should usually be revoked from `PUBLIC`:
  - `UTL_SMTP`
  - `UTL_TCP`
  - `UTL_HTTP`
  - `UTL_FILE`
  - `DBMS_OBFUSCATION_TOOLKIT`



- Example:

```
SQL> REVOKE execute ON utl_http FROM PUBLIC;
```

Limit Users with Administrative Privileges

- Restrict the following types of privileges:
  - Grants of system and object privileges
  - `SYS`-privileged connections: `SYSDBA` and `SYSOPER`
  - DBA-type privileges, such as `DROP ANY TABLE`
  - Run-time permissions
- Example: List all users with the `DBA` role:

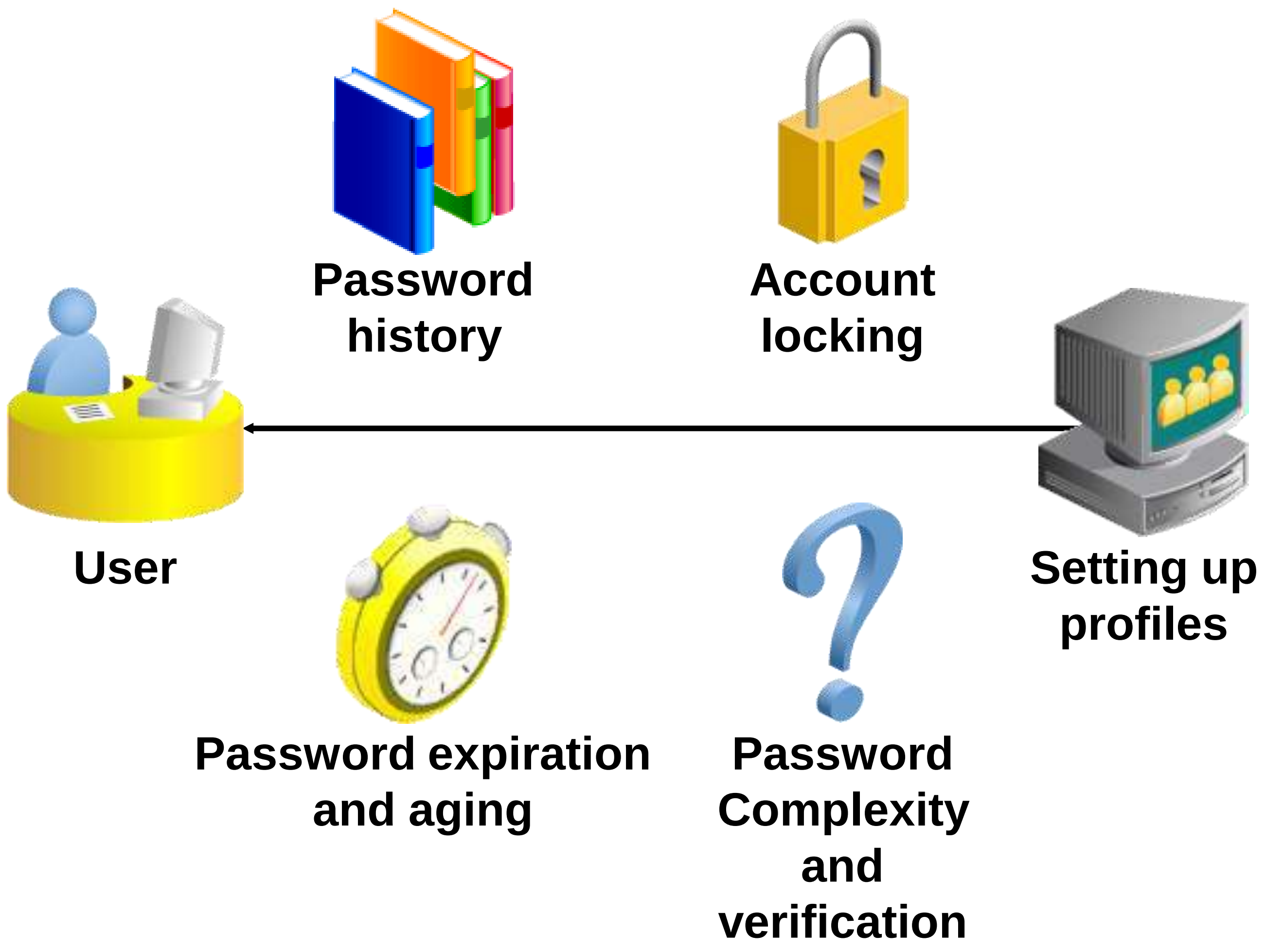
```
SQL> SELECT grantee FROM dba_role_privs
  2    WHERE granted_role = 'DBA';
GRANTEE
------------------------------
SYS
SYSTEM
```

Disable Remote Operating System Authentication

- Remote authentications should be used only when you trust all clients to appropriately authenticate users.
- Remote authentication process:
  - The database user is authenticated externally.
  - The remote system authenticates the user.
  - The user logs on to the database without further authentication.
- To disable, ensure that the following instance initialization parameter is at its default setting:

```
REMOTE_OS_AUTHENT = FALSE
```

Implement Standard Password Security Features


Password history
Account locking
User
Setting up profiles
Password expiration and aging
Password Complexity and verification

Password Account Locking

Parameter	Description
`FAILED_LOGIN_ATTEMPTS`	Number of failed login attempts before lockout of the account
`PASSWORD_LOCK_TIME`	Number of days the account is locked after the specified number of failed login attempts

Password Expiration and Aging

Parameter	Description
`PASSWORD_LIFE_TIME`	Lifetime of the password in days after which the password expires
`PASSWORD_GRACE_TIME`	Grace period in days for changing the password after the first successful login after the password has expired

Password History

Parameter	Description
`PASSWORD_REUSE_TIME`	Number of days before a password can be reused
`PASSWORD_REUSE_MAX`	Number of password changes required before the current password can be reused

Password Verification

Parameter	Description
`PASSWORD_VERIFY_ FUNCTION`	A PL/SQL function that makes a password complexity check before a password is assigned

Password verification functions must:

- Be owned by the `SYS` user
- Return a Boolean value (true or false)

```
ALTER PROFILE default LIMIT
    PASSWORD_LIFE_TIME 60
    PASSWORD_GRACE_TIME 10
    PASSWORD_REUSE_TIME 1800
    PASSWORD_REUSE_MAX UNLIMITED
    FAILED_LOGIN_ATTEMPTS 3
    PASSWORD_LOCK_TIME 1/1440
    PASSWORD_VERIFY_FUNCTION verify_function;
```

```
CREATE PROFILE app_user2 LIMIT
FAILED_LOGIN_ATTEMPTS 5
PASSWORD_LIFE_TIME 60
PASSWORD_REUSE_TIME 60
PASSWORD_REUSE_MAX 5
PASSWORD_VERIFY_FUNCTION verify_function
PASSWORD_LOCK_TIME 1/24
PASSWORD_GRACE_TIME 10;
```

Supplied Password Verification Function: `VERIFY_FUNCTION`

The supplied password verification function enforces password restrictions where the:

- Minimum length is four characters
- Password cannot be equal to username
- Password must have at least one alphabetic, one numeric, and one special character
- Password must differ from the previous password by at least three letters

Monitoring for Suspicious Activity

Monitoring or auditing should be an integral part of your security procedures.

Oracle's built-in audit tools include:

- **Standard Database auditing**
- **Value-based auditing**
- **Fine-grained auditing (FGA)**

Maintaining the audit trail is an important administrative task. Depending on the focus of the audit options, the audit trail **can grow very large very quickly**. If not properly maintained, the audit trail can consume so much space that it affects the **performance** of the system.

Standard Database Auditing

Enabled through the `AUDIT_TRAIL` parameter

- `NONE`: Disables collection of audit records
- `DB`: Enables auditing with records stored in the database
- `OS`: Enables auditing with records stored in the operating system audit trail

Can audit:

- Login events
- Exercise of system privileges
- Exercise of object privileges
- Use of SQL statements

Specifying Audit Options

- SQL statement auditing

```
AUDIT table;
```

- System privilege auditing

```
AUDIT select any table, create any trigger;
AUDIT select any table BY hr BY SESSION;
```

- Object privilege auditing

```
AUDIT ALL on hr.employees;
AUDIT UPDATE,DELETE on hr.employees BY ACCESS;
```

- Session auditing

```
AUDIT session whenever not successful;
```

Viewing Auditing Options

Data Dictionary View	Description
`ALL_DEF_AUDIT_OPTS`	Default audit options
`DBA_STMT_AUDIT_OPTS`	Statement auditing options
`DBA_PRIV_AUDIT_OPTS`	Privilege auditing options
`DBA_OBJ_AUDIT_OPTS`	Schema object auditing options

Viewing Auditing Results

Audit Trail View	Description
`DBA_AUDIT_TRAIL`	All audit trail entries
`DBA_AUDIT_EXISTS`	Records for AUDIT EXISTS/NOT EXISTS
`DBA_AUDIT_OBJECT`	Records concerning schema objects
`DBA_AUDIT_SESSION`	All connect and disconnect entries
`DBA_AUDIT_STATEMENT`	Statement auditing records

Summary

In this lesson you should have learned how to:

- Apply the principal of least privilege
- Manage default user accounts
- Implement standard password security features
- Audit database activity