🔹 1. Introduction to Data & Database Security

Concept	Description
Why Database Security?	- Databases store essential assets - All systems ultimately connect to them - Last line of defense - Offer built-in features (Encryption, RAC, Backup)
Security Measures	- Access control - Inference control - Encryption - Authentication - Integrity control - Backups - Application security

🔍 Database Security Threats

Threat Type	Description
Excessive Privileges	Users granted more rights than necessary
Legitimate Privilege Abuse	Authorized users misuse their privileges
Injection Attacks	SQL or code injection into database queries
Malware	Malicious software targeting DB systems
Storage Media Exposure	Data leaks via lost or unprotected media
Vulnerable Databases	Outdated or unpatched database software
Unmanaged Sensitive Data	Poorly classified or unsecured data
Human Factor	Insider threats, mistakes, or negligence

🔐 Security Control Types

Control Type	Purpose / Description	Techniques / Examples
🟢 **Preventive**	Stop attacks before they happen by enforcing security measures and limiting access.	*Grant least privileges*Enforce control to restrict DB operations*Minimize or avoid breaches*Verify all actions againstpolicies*Firewalls, Access Control*(Non-conformance = user or system not complying with policy)*
🟡 **Detective**	Identify and record security incidents, attacks, or policy violations.	*Identify attacks, victims, and damages*Detect footprints before attacks occur*Maintain proper system, audit, access, and network logs*Intrusion Detection Systems (IDS), Log Monitoring
🔵 **Corrective**	Recover systems and data after a security incident or failure.	*System restoration*Backup and recovery processes*Patch and configuration correction *Disaster recovery plans
🔴 **Deterrent**	Discourage potential violations or malicious actions through visible controls.	*CCTV Surveillance*Security warnings and banners*Written policies and awareness training *Legal consequences and sanctions

🦹 Threat Modeling

Term	Meaning
Asset	Anything of value to the organization (data, system, service).
Threat	Undesired act that harms an asset. A potential cause of unwanted incident.
Threat Agent	Subject causing the threat
Vulnerability	A weakness that can be exploited.
Attack (Exploit)	Malicious act by threat agent
Safeguard/controls	Countermeasure addressing vulnerabilities. A measure to reduce risk.(Preventive / Detective / Corrective / Deterrent.)
Probability	Chance of attack happening
Impact	Damage caused when threat materializes. The damage caused by a successful attack.

♻️ Types of Threat Agents

Type	Description
Accidental Discovery	Ordinary User mistakenly finds sensitive info
Curious Attacker	User intentionally explores weaknesses
Insider	Employee or contractor misusing access

🚨 Access Control & Data Security

Ensure users can only perform **authorized operations** on data.

Model	Description
DAC (Discretionary Access Control) <medium>	- owner control access to anyone on their data - Based on ACL (Access Control List) -user can give access his domain. - Example: File permissions,OS
MAC (Mandatory Access Control)	- System-enforced policies(labeling) - Based on security labels/classifications. Strictly controlld - config by system admin ((TS,S,C,UClsify)) - Example: Oracle VPD, SELinux,in gov
RBAC (Role-Based Access Control)	- Access based on organizational role -by sysadmin
RuBAC (Rule-Based Access Control)	- Access controlled by specific conditions/rules

Mandatory Access Control (MAC) Models

Model	Focus	Key Rules / Notes
Bell-LaPadula (fix)	Confidentiality	- *No read up* (simple security property) - *No write down* (star property) *-only same level,cant r/w in u/d* (strong star) Ex : Classified info protection
Biba	Integrity	*only access data in way they are allow to *goals : 1.prevent unauth modifc^ on objct 2. Prevnt Auth ppl's unauth modif^ 3. Protect internal and external consistency of object * no read down means about high integrity data being corrupt by lower integrity source (unclean – low ; clean-high) - *No read down* (simple integrity) - *No write up* (star integrity) - Prevents data contaminatio n Ex: Database record protection
Clark-Wilson	Integrity	- Uses TPs (Transformation Procedures) & IVPs (Integrity Verification Procedures) *2 main principles - Enforces *well-formed transactions* =use rules - *separation of duties = no single has too much of access* *1. contrained data it* *-use in banking,finance* *-allow modifications only thru a set of programs(TP)* *-subject cant modify object directly,must use Trastd program* *2. unconstrained data item* *-data not directly controlled by model* *3. integrity verification procedure* *-after CDI scan check whether total=total* -use multi factor approach
Brewer-Nash (Chinese Wall model) Not a fix model	Confidentiality, integrity	- Dynamic access control based on previous actions -prevent conflict of interest in firm , financial services *RBAC base access Access depends on roles *Dynamic Permitions Permissions changing base on user's previous actions *chineas wall Data is grouped into conflict of interest class.wall separate data in competiting entities.one side access krot anik side ekt access n

File	Level	Read (No Read Down)	Write (No Write Up)
File 1	Secret	✅	✅
File 2	Top Secret	✅	❌
File 3	Confidential	❌	✅
File 4	Unclassified	❌	✅

8. Integrity Principles - Use backups

Goal	Description
Integrity	Trustworthiness and correctness(accurate) of data
Main Objectives	- Prevent unauthorized changes - Prevent improper changes - Maintain consistency(follows rules, constraints,transact^ after th rules wlt adhere ?)

Integrity principle - Well-Formed Transactions (ACID)

- Arbitrary data manipula^ not allowd

Property	Description
Atomicity	All or nothing is success or fail. Rollback(updt 2k ekt dmmot 1k fail nm okkom fail)
Consistency	DB moves from one valid state to another
Isolation	Changes invisible until committed
Durability	Changes are permanent once committed

Consistency Integrity

Consistency integrity(correctness) , Entity Integrity(not null), Referential Integrity(FK), Least Privileges , Constraints : not null, unique, PK,FK,Check

DB sec +AC

Views

** **Virtual representation** of data to users.
** Control visibility of sensitive data.
** Derived using CREATE VIEW statement.
** Same number of columns derived by Sub query's columns

Options

Clause	Function
OR REPLACE	Replaces existing view
FORCE / NOFORCE	Create even if base table missing / restrict creation
WITH READ ONLY	Prevents data modification
WITH CHECK OPTION	Restrict updates to visible rows
CONSTRAINT	Assign custom constraint name(alter table add constraint)

⚙️ Updatable Views

Must not contain:
-Aggregates, DISTINCT, GROUP BY, HAVING
-Set operators, START WITH, CONNECT BY, ROWNUM

⚙️ Join View Restrictions:

-Modify only one **key-preserved** base table.
-INSERT/UPDATE/DELETE apply to that table only.

With GRANT OPTION: allows user to re-grant privileges to others.
* Single level grant : grant for only one person
* Multi level grant : grant for many many many

🧙 **Object owner** always has all privileges (non-revocable).

🧰 Admin Option

-WITH ADMIN OPTION: user can grant/revoke system privileges to others.
-Revoking system privilege has **no cascading effects**.
DBA_SYS_PRIVS = information of system privilegs granted to users

🖥️ 🚫 🍀 ✨ Virtual Private Database (VPD)

Inputs 2	Schema name ,object
Row-level security	Enforce access based on user identity or context,dynamic, by system admin
Shared schema	Different users see only their data,which row user may hv access
Central enforcement	Security tied directly to DB objects

🧩 How VPD Works

Step	Description
1. Policy Creation	Returns a predicate (dynamic WHERE clause) based on user context
2. Policy Association	Function linked to a table/view via DBMS_RLS.ADD_POLICY
3. Query Execution	Oracle dynamically appends predicate to SQL query
4. Transparent Process	User unaware of filtering

⚙️ Benefits

Security- Direct control at DB-level (not app^)
Simplicity- Single centralized policy
Flexibility- Different policies per operation

Design Tips

-Define business rules clearly(who,wht,when).
-Identify data to be protected data.
-Keep policies simple.
-Design contexts , predicates
-Test,optimize (use predicatecaching) if not policy execution is slw

🔐 Encryption

-Final layer of database security.
-Converts data into unreadable format; only decryptable with a key.
-Maintains confidentiality — *not* access control.

⚠️ Notes

-Encryption ≠ Access Control
Does not manage user privileges.
-Not for protecting from DBA users
Instead: protect & monitor DBA accounts, partition DB privileges (DDL, DML, DCL).

🔎 What to Encrypt

Option	Recommendation
All data in production DB	❌ Reduces performance, CPU cycle up,limits constraints, affects availability issues : poor keymanagmnt
Sensitive portions only	✅ Ideal:ex:credit card numbers, medical info

🧩 Encryption Layers

Type	Description
Database Encryption	Encrypts data before storage; managed automatically by DB,add-on, protect against administrators abuse, Happens **transparently** to users, Must still secure **administrative accounts**.
File/Folder Encryption	Encrypts at file system level
Application Encryption	App encrypts before writing data to DB/files applc^ encrypts when storing & applc^ decrypt before is disabled. progrmmr flexble for encrypt/decrypt ,(diffrnt keymngmnt,algrithm) need secure key management and segration of duties.

🔸 Transparent Data Encryption (TDE)

Aspect	Description
Purpose	Protects **data at rest** (on disk, backups). Not data in use/in transit.
Compliance	Meets standards like **PCI DSS**
Scope	Encrypts columns or full tablespaces
Algorithms	AES, 3DES
Advantage	Easy to implement; automatic encryption/decryption,quick

⚙️ How TDE Works

1.User inserts data → DB retrieves **master key** from **wallet(binary file defined in** sqlnet.ora**)** by using **wallet's PW**(when creating)
2.Master key decrypts table's encryption key
3.Table key encrypts user's data → stored in encrypted form
4.On retrieval, DB decrypts automatically

*** Advantages**

-Protects sensitive data if media stolen
-Helps meet compliance
-Don't need to create triggers or views to decrypt ,db do it
-Users not aware abt encryption , don't need any atten^
-No app modifications needed
-transparent to users

🗑️ Data Destruction (Data Sanitization)

Reference: NIST SP 800-88r1
Goal: Remove data remanence (residual data after deletion)to prevent from unauthorize recovery .
Cause : residual magnetic flux on HDD platters,left over electrons in SSD cells

Method	Description
Erasing	Deletes pointers only; data remains,simple deletion operation
Cleaning	Overwrites media with unclassified data (3 passes typical),reduce trad^ rcvry
Purging	Repeated cleaning + degaussing, remove all remnants bt not truestd ,use for prepare hard disks use in non sec env
Degaussing	Strong Magnetic field erases data (not effective on SSDs, CDs/DVDs)
Destruction	Physical — shredding, incineration, chemical dissolving

🧩 3. Data Masking

*- Trasform sensitive data into non-sensitive but realistic data.
*- Obfuscates (hiding) original values.
*- Used for **testing**, **auditing**, **training**.
*- Not reverse back ,
*- similar original structure
*- Table relationships preserved
*- Consistent masking on multiple runs

🧠 Masking Techniques

Technique	Description
Substitution	Replace value with another valid one
Redaction / Nulling	Replace with generic or null values (xxxx-xxxx)
Shuffling	Randomly reorder existing data within a column
Blurring	Slightly alter numeric values within a range
De-Identification	Remove identifiable info (names, IDs, etc.)

📋 Masking Constraints

Constraint	Requirement
Format Preservation	Masked data matches original format
Data Type Preservation	Maintain original types
Gender Preservation	Male/female names replaced correctly
Semantic Integrity	Logical values maintained (e.g., salary max)
Aggregate Value	Totals/averages approximately same
Uniqueness	Unique fields remain unique
Referential Integrity	Relationships not broken

🔐 Password Management

⚙️ Default Passwords

*- View users with defaults using:
*- For CDB/PDB, query from appropriate container.

🔒 Account Locking

*- **Lock SYS and SYSTEM** accounts when not in use.
*- Assign temporary DBA rights when need via:

⏰ Password Aging & Expiration

*- Managed through **profiles**:

🔑 Password Verification & Complexity

-Enforced via verification function (example in utlpwdmg.sql).
-Custom rules can be added for:
Length, Upper/lowercase, Digits/special chars
ALTER PROFILE DEFAULT LIMIT

PASSWORD_VERIFY_FUNCTION verify_function;

🔠 Case-Sensitive Passwords

*- From Oracle 11g+, passwords are **case sensitive** by default.

👤 User-Specific Restrictions

*- Logging triggers
*- Audit login attempts

DB Security	Purpose	Example Command
User Creation	**Create DB user**	**CREATE USER kasun IDENTIFIED BY Pass@123;**
Privilege Grant	**Give permission**	**GRANT SELECT ON emp TO kasun;**
Role Assign	**Assign role to user**	**GRANT CONNECT, RESOURCE TO kasun;**
Audit Role	**Monitor activities**	**AUDIT ALL BY kasun;**
TDE	**Encrypt columns**	**ALTER TABLE emp MODIFY (salary ENCRYPT);**
VPD	**Row-level control**	**CREATE POLICY emp_policy...;**

🔒 Auditing**********

The process of monitoring access to systems, tracking object modifications, and retaining records for accountability and analysis. But when creating auditing (DB ekt wda lokui audit file).

What Can Be Audited
Valid / invalid transactions, SQL statements, Data modifications, Permissions & logins

🎯 Why Auditing is Used

Purpose	Description
Accountability	Track who did what
Deterrence	Deter misuse by users/intruders
Investigation	Trace suspicious actions
Compliance	Meet legal / policy standards
Monitoring	Observe critical DB activities
Non-repudiation	Prevent denial of actions

🧩 Oracle Auditing Modes

Mode	Description
Mixed Mode	Default in new installations; combines traditional & unified auditing
Pure Unified Auditing	Centralized; can apply to PDB or CDB levels

🧰 Best Practices for Auditing

** **Be Focused** – Audit only critical activities
** **Manage Growth** – Archive/purge audit trails
** **Periodic Review** – Check audit settings/logs
** **Control Access** – Restrict permissions to audit logs
** Maintain both **DB** & **OS-level** audit trails
** Before backup archives review and then archive

👥 Audit Roles

Role	Description
AUDIT_ADMIN	fullControl,Create/manage audit policies, run AUDIT/NOAUDIT, view audit data
AUDIT_VIEWER	View audit records, execute DBMS_AUDIT_UTIL

📁 Audit Trails

Type	Description
DB	Writes standard audit content to SYS.AUD$
DB, EXTENDED	Adds SQL text & bind variables
OS	Writes audit content to OS text files
XML	Writes audit + FGA logs in XML format
XML, EXTENDED	Includes SQL text + bind variables

⚙️ Default Auditing

**Commonly used SQL statements & privileges audited automatically.
** Controlled by:
AUDIT_TRAIL = DB;
Disable auditing:
AUDIT_TRAIL = NONE;

📋 Audit Options

Option	Description
Success / Failure	Audit successful, failed, or both
Frequency	Once per session (BY SESSION) or every time (BY ACCESS)
Scope	All users or specific users

🧠 Types of Auditing

1.Standard Auditing , 2.FGA Auditng

Category	Focus	Description	Scope /
Statement Auditing	**What SQL statements are executed**	**Audits execution of specific SQL statements or groups of statements affecting certain database objects.**	**General SQL command level**
Privilege Auditing	**Who performs database-level actions**	**Audits usage of system privileges (e.g., CREATE, DROP). Can apply to all users or specific users. If both statement and privilege auditing apply, only one audit record is generated.**	**User / System level**
Schema Object Auditing	**What operations occur on specific objects**	**Audits specific actions (e.g., SELECT, INSERT) performed on specific schema objects (tables, views, etc.). Applies to all users with privileges on that object.**	**Object level**

🔸 Audit Options

**BY = specify a list of users to limit scope
-BY SESSIONS (1 record for 1 session for 1 statmnt)
-BY ACCESS (many recrdr for 1 session for 1 statmnt)
**WHENEVER *(dmme ntnm auto sucss,fail dekm)*
-WHENEVER SUCCESSFULL
-WHENEVER NOT SUCCESSFUL
**AUDIT ANY system privlge require to audit another schema or not in default audit opt^

🔐 Protecting the Audit Trail

*Restrict DELETE ANY TABLE privilege
*Audit the audit trail itself:
AUDIT INSERT, UPDATE, DELETE ON SYS.AUD$ BY ACCESS;
*Audit SYSDBA & SYSOPER:
audit_sys_operations = TRUE
*Ensure **segregation of duties**
***Regular backups** of audit records

📄 Controlling Audit Size

*Disable unnecessary auditing
*Periodically archive & clean audit logs

🔍 Fine-Grained Auditing (FGA)

*Audits **specific column/row access** based on query content.
*Focuses on **security-relevant data** only.
*Stored in SYS.FGA_LOG$.

🕸️ Requirements

*EXECUTE privilege on DBMS_FGA
*Owned by SYS user

🧩 FGA Parameters

Parameter	Description
object_schema, object_name	Target table or view
policy_name	Unique policy identifier
audit_condition	Boolean condition for auditing
audit_column	Columns to audit
audit_trail	Location (DB, DB+EXTENDED, XML, etc.)
statement_types	SELECT, INSERT, UPDATE, DELETE
audit_column_opts	ANY_COLUMNS / ALL_COLUMNS (default: ALL)

📝 Note:*FGA policies **cannot be modified**. Must drop & recreate.
*FGA-enabled colmns **cnt be encrypted/decrypted** unls policyIs disabled.

🛡️ Advantages of FGA

*Granular auditing (column-level)
*Triggers alerts for suspicious activity
*No initial parameter setup needed
*User-defined conditions
*Supports compliance for sensitive data
Ex: Audit if finance clerks view salary > 5000 during off-hours or from unauthorized IP.

✅ Quick Summary

Area	Key Focus
Auditing	Accountability, compliance, monitoring
Types	Standard (DDL/DML/Privileges), Fine-Grained
Roles	AUDIT_ADMIN, AUDIT_VIEWER
Storage	SYS.AUD$, SYS.FGA_LOG$, XML logs
Protection	Restrict privileges, backup logs, audit SYS operations
FGA	Row/column-based audit using DBMS_FGA.ADD_POLICY

✅ Traditional Database Security

**Focus: Protecting data at rest
**Uses proven mechanisms based on CIA Triad:
**Techniques:Authentication,Access control,Encryption,Hashing,Auditing & Accounting
**Privacy: Newer concern in databases

🐧 Linux

Feature	Description
Type	Free, open-source OS under GPL
Rights	Run, study, modify, redistribute
Nature	Multi-user, multitasking, multiprocessing, multithreading

Feature	Description
Compatibility	Coexists with other OS; runs on multiple platforms (x86, AMD64,.)

Operating System (OS)

Term	Description
Definition	Software that allows hardware and software to communicate
Function	Manages files, memory, devices, processes, and system operations

Open Source (OSS)

*Publicly accessible source code (view, modify, share)
*Built collaboratively through community contribution
*Encourages transparency and innovation

Why Use Linux

Advantage	Description
Open Source	Free to use and modify
High Security	Less prone to viruses/malware
High Stability	Rarely crashes; long-term performance
Runs on Any Hardware	Efficient resource use; customizable,fast,low overhead
Applications	Thousands of free open-source apps
Community Support	Strong documentation, forums, live help

Linux Distributions

Distribution	Description
Ubuntu	Based on Debian; popular desktop/server OS
Debian	Community-supported, free and open-source
Fedora	Community-supported; sponsored by Red Hat
Red Hat Enterprise Linux (RHEL)	Commercial version by Red Hat
OpenSUSE	Community-driven, supported by SUSE Project

OS Architecture Overview

Layer	Description / Example
Hardware	CPU, RAM, Disk, Motherboard
Kernel	Core of OS; manages memory, CPU, I/O
Shell	User interface that sends commands to kernel
Utilities	Text editors, browsers, etc. (run on Shell)

Core Components of Linux

Component	Description
Kernel	Heart of OS; manages hardware, processes, memory, devices, files
Boot Loader	Loads OS into memory (e.g., GRUB, LILO)
Shell	Command interpreter (bash, sh, csh, ksh)
File System	Organizes and stores data (Ext4, XFS, BtrFS, etc.)
Desktop Envrmnt	GUI interface (GNOME, KDE, Cinnamon, Xfce)
Daemons	Background services (systemd, httpd, vsftpd)
Graphical Server	Manages display output (X Server)

Kernel Details

Function	Description
Memory Management	Tracks usage and allocation
Process Management	Controls CPU scheduling and execution
Device Drivers	Interface between hardware & processes
File Management	Organizes files/folders

Boot Loader

Boot Loader	Key Features
LILO	Old loader, no CLI, no multiboot
GRUB 2	Password support, encryption, multiboot, modern default

Shell

Shell Type	Description
sh	Bourne Shell
bash	Bourne Again Shell (default in most distros)
ksh	Korn Shell
csh	C Shell

Role: Takes user commands → sends to kernel for execution.

File System

Concept	Description
Definition	Defines how files are stored, named, and retrieved
Types	Ext2, Ext3, Ext4, XFS, BtrFS, GlusterFS
Default (RHEL7)	XFS – scalable and high performance

Desktop Environments

Environment	Description
GNOME 3	Default in many distros
KDE Plasma 5	Highly customizable
Cinnamon	User-friendly
Xfce	Lightweight and fast

Daemons & Graphical Server

Type	Description
Daemons	Background services ending with “d” (e.g., systemd, httpd)
Graphical Server (X Server)	Manages graphical display; not directly interacted with

What is OS Hardening

Concept	Description
Definition	Process of securing an OS (windws/Linux/macOS) by applying patches, tools, configurations to reduce vulnrbilties
Goal	Protect sensitive systems by reducing attack surface and eliminating potential attack vectors.minmn vulnrblies
Includes	Tools, techniques, and best practices for OS, applications, firmware, and infrastructure.

Types of Hardening

Type	Focus
Operating System Hardening	Secure OS-level components and configurations.
Network Hardening	Secure network devices and protocols.
Server Hardening	Protect server configurations and access.
Application Hardening	Secure application code and runtime behavior.

Core OS Hardening Practices

Category	Key Actions
Account & Passwords	- Create complex passwords - Change passwords regularly - Disable guest/unnecessary accounts - Create non-admin accounts - Disable automatic login
Access & Lock	- Configure screen lock timeout - Enable screensaver lock

Category	Key Actions
	- Set lockout policy after failed attempts
System Configuration	- Disable unnecessary services - Disable start menu options - Enable Windows Firewall - Enable BitLocker encryption - Use NTFS file system
Updates & Monitoring	- Enable automatic Windows updates - Check security patches - Review Event Viewer logs
Privacy & Remote Access	- Disable remote access - Review privacy settings regularly

Hardening Step	Description
Disable Unused Services	Reduces attack surface
Patch Regularly	Fix vulnerabilities
Enforce Strong Password Policy	Prevent unauthorized logins
Use Firewall (UFW/iptables)	Block unwanted traffic
Enable Audit Logging	Detect suspicious activity
Limit Root Access	Use sudo for admin tasks

Detailed Hardening Tasks

Disabling Unnecessary Services

** Many services (e.g., Windows XP had ~90 by default).
** Each service can be a potential risk.
** Research before disabling (not always obvious).
** Use testing and monitoring after changes.

Protect Management Interfaces

** Change default credentials.
** Add authentication layers (multi-factor, third-party tools).
** Restrict access to management consoles or admin panels.

Password Protection

Aspect	Description
Weak Passwords	Vulnerable to brute-force or offline attacks.
Strong Passwords	Use complexity + frequent refresh.
Example Breach	*Rockyou.com (2009)* – SQL Injection led to 32M plaintext passwords leaked.

Account Management

Task	Purpose
Disable unnecessary accounts	Guest, root, mail, etc.
Disable interactive logins	Restrict login privileges.
Create non-admin users	Use standard accounts for daily use.
Disable auto-login	Require credentials at each startup.

Screen Lock & Timeout

Step	Action
1	Open **Settings → Personalization → Lock screen**
2	Select **Screen saver settings**
3	Set **wait time** & enable **"On resume, display logon screen"**

Use a Password Manager

**Generates complex passwords.
**Eliminates need to memorize many passwords.
**Simplifies password updates.

Turn On Windows Firewall

Steps
Control Panel → System & Security → Windows Firewall → Turn On Firewall
Select "Turn on Windows Firewall" under both private & public networks.

Enable Full Disk Encryption (BitLocker)

Step	Description
1	Right-click drive → "Turn on BitLocker"
2	Choose **password** and **recovery key**
3	Select **Encrypt Entire Drive → New Encryption Mode**
4	Start encryption

Enable Automatic Updates

Step	Description
1	Settings → Update & Security → Windows Update
2	Advanced Options → Choose "Automatic (Recommended)"

Disable Remote Access

Step	Description
1	Search "Remote settings" → "Allow Remote Access"
2	Check **"Don't allow remote connections"**

Review Security Settings

Step	Description
1	Start → Settings → Privacy
2	Review permissions for apps, location, camera, microphone, etc.

NTFS File System

**Default Windows file system.
**Supports permissions, encryption, compression.
**More secure than FAT32.

Windows Event Viewer

**Monitors system, security, and application logs.
**Helps identify unusual activities.

LinuxHardening Process

-Linux is secure by design but needs **extra configuration** for better protection.
-Focus on **updates, password security, access control**, and **monitoring**.

Patching and Updates

Step	Description
Purpose	Fix vulnerabilities & mitigate threats.
Bst prctce	Automate updates and testing; keep snapshots for rollback.
Risk	Patches can break functions → test before deploying.

OS-Specific Commands:

OS	Commands
Fedora / RedHat	rpm -Uvh package.rpm / yum update package
SuSE	yast --install package / zypper patch package
Solaris	smpatch update package / pkg update --be-name updateBEname package
Ubuntu	dpkg -i package.deb / apt-get update package

Passwords & Root Privileges

Topic	Key Points
Passwrd strge	Hashes stored only in /etc/shadow.
Root User	UID 0; only one account should have this. Multiple UID 0 = serious threat.
UIDs	Prevent duplication; file ownership tied to UID.
Shared IDs	Security risk; use unique IDs.

Password Security

Aspect	Recommendations
User Awareness	Educate users on password policies & social engineering.
Strength	Use long, complex passwords; add **salt**.
Recycling	Disallow reuse of old passwords.
Lifespan	90–120 days (shorter if <8 characters).

PAM (Pluggable Authentication Module)

**Config file: /etc/pam.d/
**Controls: -Minimum length -Complexity rule -Reuse prevention

Security Policy

Key Questions	Examples
What to protect?	Computers, data, applications

Key Questions	Examples
From what?	Unauthorized access, malware, data theft
From whom?	Insiders, external attackers

⚖️ **Balance security and prfrmnce** moreSecurity=less performance.

👥 **Default Accounts**

Task	Action
Remove unused default accounts	Reduces attack vectors
Keep essential system accounts	Needed for operations
Unsure about an account?	Lock it instead of deleting (/etc/shadow invalid hash)
Logging	Record all account modifications

🧩 **Attack Surface**

-Total points exposed to attack. To prevnt: Restrict access to essential network services(a single gateway instead of multiple endpoints)

💻 **9. Security Measures**

🔸 **Servers**

-* Secure ports using **firewall**
-* Enable only required services
-* Change default passwords
-* Use **2FA**
-* Use **Fail2Ban** for brute-force protection
-* Use **HTTPS** instead of HTTP

🔸 **Laptops.**

-* Protect from theft and open Wi-Fi risks
-* Enable **Full Disk Encryption**
-* Use **firewall**
-* Use **VPN** for remote connections

🔸 **Networks**

-* Disable unused services
-* Use **firewalls** to block public access
-* Ensure proper authentication for all running services

🔐 **OpenSSH Security**

Setting	Description
Purpose	Encrypt communication; replace Telnet/TFTP
Server Config	/etc/ssh/sshd_config
Client Config	/etc/ssh/ssh_config or ~/.ssh/config
Important Parameters	Port, ListenAddress, Protocol (use SSHv2), LogLevel, StrictModes
Commands	sshd -d (debug), sshd -f (specify config file)

🧱 **Firewalls**

Type	Description
Filtering Gateway	Router-based firewalls (e.g., **pfSense**, **OpenSense**)
Host-based	Linux kernel firewalls (e.g., **Netfilter**, **ipfilter**)

📊 **Monitoring and Logging**

Tool	Purpose
Logcheck	Detect unusual system log entries
top / htop / glances	Monitor real-time CPU & network activity

Modes:-*Paranoid*: very detailed (use for critical systems).
-*Server / Workstation*: moderate verbosity.

🧩 **Detecting Changes**

Tool/Command	Function
dpkg-verify	Check for altered or unauthorized files (Debian)
rpm -V -a	Verify file integrity (RedHat)
AIDE	Advanced Intrusion Detection Environment – monitors file integrity

Tool/Command	Function
apropos <keyword>	Search man pages for related commands

🕵️ **Auditing – AIDE**

Feature	Description
Purpose	File integrity checking & intrusion detection
Deployment	Runs on most distros; can be centralized
Automation	Schedule checks and send alerts

✅ **Quick Exam Notes**

** Always **patch first** and **backup**.
** **Lock, don't delete** unknown users.
** **SSHv2 + Fail2Ban + Firewall = base security combo**.
** **Logs + AIDE = best detection method**.
** **Smaller attack surface → stronger defense.**

💻🧮 **Big Data** 🧬🧬🧬🧬🧬

🧯 **Drawbacks of Relational Database Systems (RDBMS)**

Drawback	Explanation
Difficult to scale	Vertical scaling is expensive and limited
Maintenance & configuration	Complex and costly for large datasets
Peak provisioning	Wastes resources during low usage
System diversification	Makes choosing the right system harder

Solution: Use **NoSQL databases** for Big Data.

🚀 **NoSQL Systems**

Feature	Details
Advantages	Elastic scaling, less admin, cost-efficient, flexible data models
Limitations	Limited access control, indexing, functionality; requires knowledge of data model

Feature	Details
Goal	Store massive data in clustered environments efficiently

🔭 **Distributed File Systems (DFS)**

** Designed for **cluster computing**.
** Most analytics (ML, stats, data mining) require access to **all data**.
** DFS enables **parallelized analytics** across clusters.

Challenges in Data Processing

****Storage:** Node failure → data unavailable. Need file organization & search.
****Parallelization:** Assign tasks, aggregate results, handle worker failures.
****Concurrency:** Deadlocks, resource starvation.
Solution: Execution framework ("runtime") + developer-specified computations.

👓 **Hadoop Solutions**

Challenge	Hadoop Feature
Storage	HDFS – redundant, fault-tolerant
Parallel processing	MapReduce – divide-and-conquer tasks
Concurrency	YARN – job coordination & resource allocation
Scaling	Scale-out on commodity hardware, move processing to data

Key Points

****Process data sequentially to avoid expensive seeks.
****Cluster composed of **MasterNodes**, **WorkerNodes**, **Client Nodes**.
****Master: NameNode, Secondary NameNode, Job Tracker
****Workers: DataNode, TaskTracker

🌐 **Hadoop Ecosystem**

MapReduce

**Job Tracker(Master):Schedules tasks, monitors jobs, handles failures
**Task Tracker (Worker): Executes tasks, reports progress
**Limitation: Single task at a time, no real-time or ad-hoc analysis

YARN (MapReduce 2.0)

****Resource Manager (Master)**: Allocates resources
****Node Manager (Worker)**: Executes tasks
****Application Master**: Requests containers, monitors tasks
YARN Container: Collection of CPU, RAM, disk resources for job

General Workflow

1.Client submits job → Resource Manager
2.Resource Manager launches Application Master
3.Application Master requests containers
4.Resource Manager allocates resources on Worker nodes

Apache Spark

Feature	Description
Engine	Unified analytics engine for large-scale data processing
Speed	Extremely fast due to in-memory caching
APIs	Scala, Java, Python, R
Libraries	MLlib (Machine Learning), GraphX, Spark Streaming, Spark SQL
Advantages	Implicit data parallelism, fault tolerance, easier programming than MapReduce

Requirements

****Cluster manager:** Standalone, Hadoop YARN, Mesos, Kubernetes
****Distributed storage:** HDFS, Alluxio, S3, Cassandra, Lustre, etc.
****Architecture:** Master-Worker

Performance:

****Spark can sort 100TB of data **3X faster** than Hadoop MapReduce using **10X fewer machines**.

****Limitations:**
-Not ideal for simple tasks (MapReduce may be better).
-Not a multi-user environment.
-Requires careful memory planning per dataset.

Security Considerations for Big Data

****Ensure **data confidentiality, integrity, and availability**.
****Access control for distributed systems.
****Encryption in transit and at rest.
****Monitoring for suspicious activity across clusters.

IoT Security Cheat Sheet

Threat	Impact	Mitigation
Data Interception	Data theft	TLS/SSL encryption
Device Spoofing	Fake data, wrong actions	Digital certificates
Firmware Malware	Device compromise	Firmware signing, updates
Privacy Violation	Identity exposure	Data anonymization
Unauthorized Access	Data leaks	Role-based access, authentication

Quick Exam Reminders

****RDBMS → Hard to scale → NoSQL preferred for Big Data
****Hadoop = HDFS + MapReduce + YARN
****Spark = Fast, in-memory, supports ML, Graphs, Streaming, SQL
****YARN manages resources;Containers hold CPU/RAM/disk for tasks
****Security = holistic approach for distributed environment

2. Why Big Data Security is Different

Aspect	Traditional DB	Big Data System
Scale	Small to moderate	Petabytes to exabytes
Data Sources	Controlled	Heterogeneous, distributed
Analysis	Simple queries	Complex analytics, ML, streaming
Security Approach	Standardized audits	Challenging due to scale & distribution

Implication: Traditional DBsecurity tchniqs **cant fully apply** to big data.

Key Security Challenges in Big Data

1.Data Validity & Authenticity: Logs and event data from endpoints may be tampered.
2.Open-source Frameworks: Vulnerabilities in Hadoop, Spark, etc.
3.Distributed Computing: Hard to secure multiple nodes and clusters.
4.Non-Relational Databases: NoSQL systems hv limtd access contrls.
5.Insider Threats: Hard to monitor access to data mining tools.
6.Auditing&Compliance:stndrd secrty audtng is difficult at lrge scale.

Big Data Security Stages

Stage	Security Focus
Data Acquisition	Secure transit from source to storage / ingestion
Data Storage	Protect data in storage layers (e.g., HDFS, cloud)
Data Analysis	Ensure confidentiality of analytics outputs (reports, dashboards)

5Vs of Big Data

V	Meaning	Example / Context
Volume	**Massive amount of data generated (terabytes, petabytes).**	**IoT sensors, CCTV footage.**
Velocity	**Speed of data generation and processing.**	**Real-time streaming from IoT devices.**
Variety	**Different data types (structured, unstructured, multimedia).**	**Text, audio, video, sensor logs.**
Veracity	**Accuracy and reliability of data.**	**Filtering out false or corrupted sensor data.**
Value	**Useful insights or benefits extracted from data.**	**AI-based decisions, smart city analytics.**

Security Techniques & Best Practices

***Authentication & Access Control** → Restrict who can access data & analytics tools
***Encryption** → Data at rest and in transit
***Monitoring & Auditing** → Detect anomalies, suspicious behavior
***Use of Secure Frameworks** → Harden Hadoop/Spark clusters
***Data Masking & Privacy** → Protect sensitive information in outputs
***Incident Response** → Plan for breaches and insider threats

Quick Exam Takeaways

*Big data differs from trdt^nl DBs → need new security approaches
*Secure all stages: **Acquisition → Storage → Analysis**
*Focus on distributed systems,NoSQL,insider threats,&analytics outpt
*combne **tradi^nl techniques**(CIA Triad)with**bigdataspecific strategies**

Queries :

VIEW MANAGEMENT

CREATE [OR REPLACE] [FORCE | NOFORCE] VIEW view_name [(col_alias,...)] AS <subquery>
[WITH [CHECK OPTION] [READ ONLY] [CONSTRAINT]];
-- Create or replace view

```
CREATE OR REPLACE VIEW
emp_rec AS
SELECT employee_id, first_name,
last_name, salary
FROM hr.employees
WHERE salary < 50000
WITH READ ONLY;

-- Grant and revoke privileges
GRANT CREATE VIEW TO
user1;
GRANT SELECT ON emp_rec TO
scotte;
GRANT SELECT ON view_name
TO user1;
REVOKE SELECT ON
view_name FROM user1;

-- Drop a view
DROP VIEW view_name;
```

🔹 **PRIVILEGES & ACCESS CONTROL**

```
-- Object-level grants
GRANT SELECT, INSERT,
DELETE ON Employee TO Nial,
Kasun;
GRANT INSERT(Empid, Ename)
ON Employee TO Kasun;

-- Grant to all users
GRANT SELECT ON emp_view
TO PUBLIC;

-- Grant with grant option
GRANT UPDATE ON table_name
TO HR WITH GRANT OPTION;

-- System-level grants
GRANT CREATE ANY TABLE,
SELECT ANY TABLE TO user1;

-- DBA role
GRANT DBA TO chamantha;
```

🔹 **ACCOUNT MANAGEMENT**

```
-- Lock / unlock accounts
ALTER USER scott ACCOUNT
LOCK;
ALTER USER scott ACCOUNT
UNLOCK;

-- Check default password users
SELECT * FROM
DBA_USERS_WITH_DEFPWD;
```

🔹 **PASSWORD PROFILE SETTINGS**

```
PASSWORD_REUSE_TIME   120
PASSWORD_LIFE_TIME    30
PASSWORD_LOCK_TIME    5
PASSWORD_GRACE_TIME   3
FAILED_LOGIN_ATTEMPTS  3
```

🛡️ **DATABASE AUDITING**

a) Statement Auditing

```
AUDIT TABLE;
AUDIT SELECT TABLE;
```

b) Privilege Auditing

```
AUDIT SELECT ANY TABLE;
```

c) Schema Object Auditing

```
AUDIT SELECT ON
hr.employees;
```

Examples:

```
AUDIT DELETE ON hr.employee
BY ACCESS;
AUDIT EXECUTION ON
hr.employee BY ACCESS;  --
PL/SQL e
AUDIT SELECT, INSERT,
DELETE ON hr.department BY
ACCESS WHENEVER
SUCCESSFUL;
AUDIT SELECT ON DEFAULT
BY ACCESS WHENEVER NOT
SUCCESSFUL; -- new objects also
audited
```

Remove Auditing:

```
NOAUDIT DELETE ON emp;
NOAUDIT ALL ON DEFAULT;
```

Fine-Grained Auditing (FGA)

Enable FGA:

```
BEGIN
 DBMS_FGA.ADD_POLICY(
   object_schema   => 'SCOTT',
   object_name     =>
'EMPLOYEES',
   policy_name     =>
'SALARY_CHK_AUDIT',
   audit_condition => 'SALARY >
50000',
   audit_column    => 'SALARY'
 );
END;
/
```

View & Test:

```
SELECT POLICY_NAME FROM
DBA_AUDIT_POLICIES;
SELECT * FROM
DBA_FGA_AUDIT_TRAIL;
SELECT * FROM
DBA_COMMON_AUDIT_TRAIL
;
```

Manage Policies:

```
BEGIN
DBMS_FGA.DISABLE_POLICY(
'HR','EMPLOYEES','CHK_HR_E
MPLOYEES');
DBMS_FGA.DROP_POLICY('HR
','EMPLOYEES','CHK_HR_EMPL
OYEES');
END;
/
```

📌 **VPD (Virtual Private Database)**

VPD Function:

```
CREATE OR REPLACE
FUNCTION set_auth_user_privs(
  schema_p IN VARCHAR2,
table_p IN VARCHAR2
) RETURN VARCHAR2 AS
BEGIN
  RETURN '"UID" =
SYS_CONTEXT("hr_AUTH_detai
ls", "emp_no")';
END;
/
```

Attach Policy:

```
BEGIN
 DBMS_RLS.ADD_POLICY(
   object_schema   => 'hr',
   object_name     =>
'SYSTEM_AUTH',
   policy_name     =>
'USER_AUTH_POLICY',
   function_schema => 'system',
   policy_function =>
'set_auth_user_privs',
   statement_types => 'SELECT'
 );
END;
/
```

Example VPD Function (Static Policy):

```
CREATE OR REPLACE
FUNCTION
manager_policy_fn_static (
   schema_name IN VARCHAR2,
   table_name  IN VARCHAR2
) RETURN VARCHAR2 IS
BEGIN
  RETURN 'manager_ID =
"S00001"';
END;
/
```

Apply VPD to Customer Table:

```
BEGIN
 DBMS_RLS.ADD_POLICY(
   object_schema   =>
'C##SysAdmin',
   object_name     => 'Customer',
   policy_name     =>
'MANAGER_POLICY_STATIC',
   function_schema =>
'C##SysAdmin',
   policy_function =>
'manager_policy_fn_static',
   statement_types => 'SELECT'
 );
END;
/
```

🧰 **DATA MASKING & REDACTION**

Create Masking Definition:

```
BEGIN

DBMS_REDACT.ADD_POLICY(
   object_schema =>
'C##SYSADMIN',
   object_name   =>
'CUSTOMER_PHONE',
   column_name   => 'PHONE',
   policy_name   =>
'MASK_PHONE',
   function_type =>
DBMS_REDACT.FULL,  -- full
masking
   expression    => '1=1'
 );
END;
/
```

🔐 **DATA ENCRYPTION**

Encrypting NIC Column (using DBMS_CRYPTO):

```
ALTER TABLE Customer ADD
(NIC_ENC RAW(2000));

BEGIN
 FOR r IN (SELECT customer_ID,
NIC FROM Customer) LOOP
   UPDATE Customer
   SET NIC_ENC =
DBMS_CRYPTO.ENCRYPT(

UTL_RAW.CAST_TO_RAW(r.NI
C),

DBMS_CRYPTO.DES_CBC_PK
CS5,
```

```
UTL_RAW.CAST_TO_RAW('DO
SS2025')
    )
    WHERE customer_ID =
r.customer_ID;
  END LOOP;
  COMMIT;
END;
/
```