2M

UNIT 1

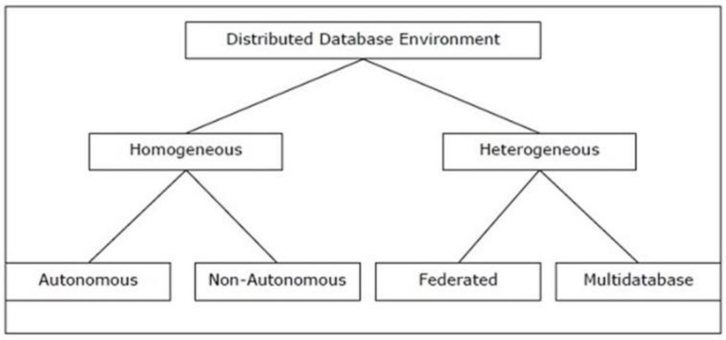
1. DEFINE DD

- **Definition:** A Distributed Database consists of a set of connected databases spread over several networked locations and controlled by a D-DBMS.
- **Characteristic:** Information is physically saved across sites yet retrieved as one unified logical database.
- **Benefit:** Offers increased dependability because the failure of a site does not halt the whole system.
- **Drawback:** Necessitates costly software to handle data synchronization and management.
- **Why Distributed Database:** Used to improve performance and availability through data distribution.

2. DEFINE DATA FRAGMENTATION

- **Definition:** Data Fragmentation is the process of dividing a table into smaller fragments to store data efficiently across multiple sites.
- **Method:** Fragmentation of a table can be done using horizontal, vertical, or hybrid fragmentation.
- **Advantage:** Improves database performance by storing data closer to the site where it is frequently used.
- **Disadvantage:** Access speed may be low when data is required from multiple fragments.
- **Rule:** Fragmentation must allow the original table to be reconstructed from its fragments.

3. GIVE THE TYPES OF DD



- Homogeneous Distributed Database: All sites use the same DBMS and operating system.
- Autonomous (Homogeneous): Each database works independently and controls its own operations.
- Non-Autonomous (Homogeneous): Databases are controlled by a central DBMS that manages all sites.
- Heterogeneous Distributed Database: Sites use different DBMSs, operating systems, or data models.
- Federated Database: Independent heterogeneous databases are integrated to work as one system.
- Multidatabase (Un-federated): Different databases are accessed through a central interface but remain loosely connected.

UNIT 2

1. DEFINE SERIALIZABILITY THEORY

Serializability theory is a fundamental concept in database concurrency control which states that a concurrent (parallel) schedule of transactions is correct if it produces the same effect as some serial execution of those transactions.

Key Points of Serializability Theory

- Ensures database consistency in concurrent transaction execution
- Allows parallel execution with the correctness of serial execution
- Forms the basis of concurrency control protocols like Two-Phase Locking
- Prevents anomalies such as lost updates, dirty reads, and inconsistent retrievals

Types of Serializability

Conflict Serializability: A schedule can be transformed into a serial schedule by swapping non-conflicting operations

View Serializability: A schedule produces the same views of data items as a serial schedule

2. GIVE THE DIFFERENR TYPES OF SECURITY

Cyber Security: Cyber security means securing our computers, electronic devices, networks, programs, systems from cyber-attacks. Cyber-attacks are those attacks that happen when our system is connected to the Internet.

Information Security: Information security means protecting our system's information from theft, illegal use and piracy from unauthorized use.

Application Security: Application security means securing our applications and data so that they don't get hacked and also the databases of the applications remain safe and private to the owner itself so that user's data remains confidential.

Network Security: Network security means securing a network and protecting the user's information about who is connected through that network

3. EXPLAIN THE TYPES OF CYBER ATTACKS AND THE SECURITY TECHNIQUES USED TO PROTECT DISTRIBUTED SYSTEMS

Cyber Attacks:

Cyber-attacks are malicious activities carried out to damage, disrupt, or gain unauthorized access to computer systems or networks. Examples include DoS attacks, phishing, malware, spoofing, and social engineering.

Security Techniques:

Security techniques are methods used to protect systems from attacks, such as using encryption, limiting secret key usage, securing networks, and minimizing the trusted computing base.

UNIT 3

1. EXPLAIN PASSWORD BASED AUTHENTICATION

Password-based authentication is a common method of verifying user identity using a shared secret password. Although it is simple and inexpensive, it is vulnerable to off-line dictionary attacks, where attackers try possible passwords.

To prevent this, secure protocols like Encrypted Key Exchange (EKE) and Secure Remote Password (SRP) are used to safely generate a session key without revealing the password.

Example:

When a user logs into an email account by entering a username and password, the system verifies the password before allowing access. Secure protocols like SRP ensure that the password is not sent directly over the network, protecting it from attackers.

2. DIFFERENTIATE INTERNET GRAPHS AND GENERALIZED RANDOM GRAPH NETWORKS

Internet Graphs	Generalized Random Graph Networks
Represent real-world networks like the Web and Internet	Are theoretical / simulated models
Degree distribution follows Power law, Zipf's law, Pareto law	Degree distribution is given as input, edges are random
Contain hubs with very high connectivity	Random edge distribution, clustering tends to zero
Example: Popular websites like Google or YouTube with millions of links	Example: Computer-generated network with power-law degree distribution

3. DIFFERENTIATE BETWEEN PROTOCOLS BASED ON SYMMETRIC CRYPTOSYSTEMS AND PROTOCOLS BASED ON ASYMMETRIC CRYPTOSYSTEMS

Protocols Based on Symmetric Cryptosystems	Protocols Based on Asymmetric Cryptosystems
Use one shared secret key for encryption and decryption	Use a pair of keys: public key and private key
Same key is known to both communicating parties	Public key is shared openly; private key is kept secret
Faster and computationally efficient	Slower compared to symmetric protocols
Key distribution and storage is difficult	Key distribution is easier using public keys
Scalability is limited as each pair needs a unique key	Highly scalable for large networks
Vulnerable to key compromise and replay attacks	More resistant to replay and impersonation attacks
Example: Wide-Mouth Frog protocol, Needham–Schroeder (symmetric)	Example: RSA-based authentication, Needham–Schroeder (public key)

3. EXPLAIN TAPESTRY

Tapestry is a structured peer-to-peer overlay network that provides scalable, location-independent object lookup using prefix-based routing. Nodes and objects are mapped to a common identifier space using hashing, and routing is performed by progressively matching prefixes to reach the object's root or surrogate root, ensuring efficient lookup and fault tolerance even under node churn.

12M

UNIT 1

1. EXPLAIN THE REPLICATION AND ALLOCATION TECHNIQUES FOR DDBMS

In a Distributed Database Management System (DDBMS), data must be placed efficiently across multiple sites to improve performance, availability, and reliability. This is achieved using data replication and data allocation techniques.

1. Data Replication

Data Replication is the process of storing multiple copies of the same data at different sites in a distributed database system to ensure fault tolerance and fast data access.

Need for Replication

- Eliminates single point of failure
- Improves data availability
- Reduces query response time

Types / Techniques of Data Replication

Snapshot Replication

- Data is copied periodically at fixed intervals.
- Suitable when data changes infrequently.

Near Real-Time Replication

- Data changes are propagated almost immediately.
- Ensures high data freshness.

Pull Replication

- Target site requests updates from the source site when needed.
- Useful when target does not require continuous updates.

Full Replication

- Entire database is replicated at all sites.
- High availability but high storage and update cost.

Partial Replication

- Only selected fragments are replicated.
- Reduces storage and synchronization overhead.

Advantages of Data Replication

- High Reliability
- Reduced Network Load

Disadvantages of Data Replication

- Increased Storage Cost
- Complex Update Mechanisms

2. Data Allocation

Data Allocation refers to deciding where data fragments and replicas should be stored in a distributed database system.

Types of Data Allocation Techniques

Centralized Allocation

- All data stored at a single site.
- Simple but poor reliability.

Fragmented Allocation

- Data fragments stored at different sites.
- Improves locality of access.

Replicated Allocation

- Multiple copies of data fragments stored at multiple sites.
- Improves availability.

Hybrid Allocation

- Combination of fragmentation and replication.
- Most commonly used approach.

Dynamic Allocation

- Data placement changes based on access patterns.
- Improves adaptability.

Factors Affecting Allocation Decisions

- Query access frequency
- Communication cost
- Storage availability
- Reliability requirements
- Update frequency

2. EXPLAIN THE ARCHITECTURE FOR DDBMS

A Distributed DBMS architecture defines how data, control, and processing are organized across multiple sites in a distributed environment.

Parameters of DDBMS Architecture

DDBMS architectures are designed based on the following three parameters:

- Distribution – Describes how data is physically distributed among different sites.
- Autonomy – Indicates the degree of independence each local DBMS has in control and operation.
- Heterogeneity – Refers to similarity or dissimilarity of DBMSs, data models, and operating systems.

Architectural Models of DDBMS

1. Client–Server Architecture for DDBMS

This is a two-tier architecture where responsibilities are divided between clients and servers.

Characteristics

- Server performs data storage, query processing, optimization, and transaction management.
- Client provides user interface and handles query submission.
- Some functions like consistency checking may be shared.

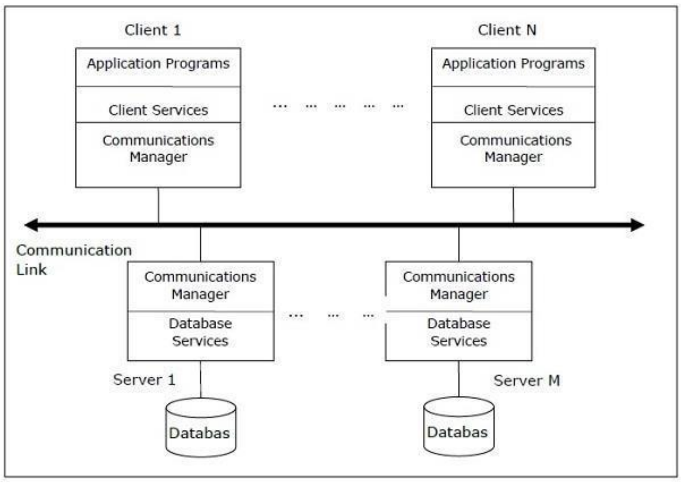
Types

Single Server – Multiple Clients

One server handles requests from multiple clients.

Multiple Server – Multiple Clients

Several servers store data and multiple clients access them.



2. Peer-to-Peer Architecture for DDBMS

In this architecture, each site acts as both client and server.

Characteristics

- No centralized control.
- All peers share resources and cooperate in query processing.
- Suitable for highly distributed and autonomous environments.

Schema Levels

- Local Internal Schema – Physical storage at each site
- Local Conceptual Schema – Logical organization at each site
- Global Conceptual Schema – Enterprise-wide integrated view
- External Schema – User views and application access

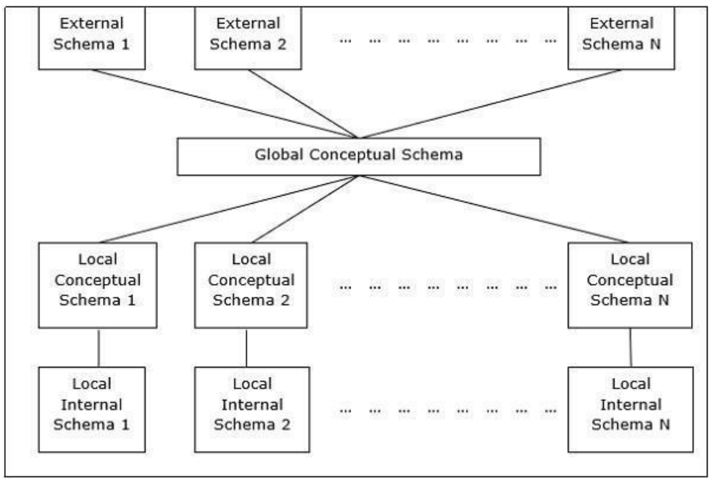
Major Components

User Processor

- User interface handler
- Semantic data controller
- Global query optimizer and decomposer
- Distributed execution manager

Data Processor

- Local query optimizer
- Local recovery manager
- Run-time support processor



3. Multi-DBMS Architecture

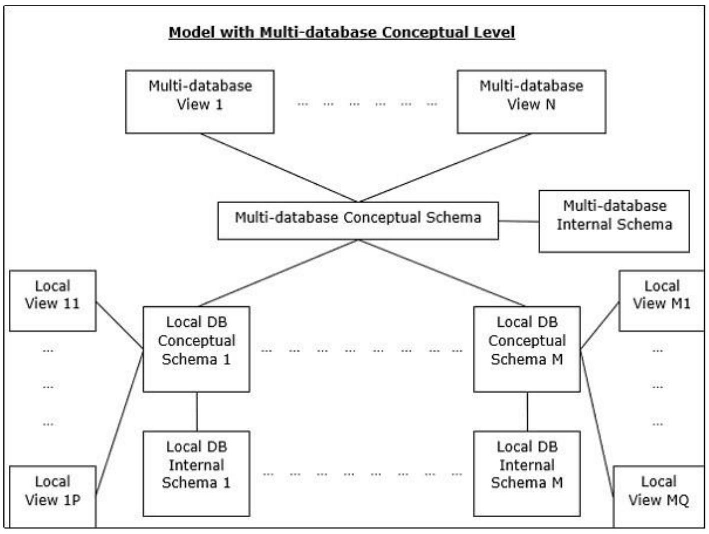
A Multi-DBMS integrates multiple autonomous database systems that may be heterogeneous.

Schema Levels

- Multi-database View Level
- Multi-database Conceptual Level
- Multi-database Internal Level
- Local Database View Level
- Local Database Conceptual Level
- Local Database Internal Level

Design Alternatives

- With multi-database conceptual level – Global integrated schema exists.
- Without multi-database conceptual level – Databases interact without a global schema.


Model with Multi-database Conceptual Level
Multi-database View 1
Multi-database View N
Multi-database Conceptual Schema
Multi-database Internal Schema
Local View 11
Local View 1P
Local DB Conceptual Schema 1
Local DB Internal Schema 1
Local DB Conceptual Schema M
Local DB Internal Schema M
Local View M1
Local View MQ

3. EXPLAIN PARALLEL DATABASE AND QUERY PROCESSING

Parallel Databases

A Parallel Database System improves performance by using multiple CPUs, disks, and memory units simultaneously to execute database operations.

Key Features

- Data loading and query processing are done in parallel.
- Large tasks are divided into smaller subtasks executed concurrently.
- Suitable for OLTP and Decision Support Systems (DSS).
- Provides faster response time than centralized or client–server DBMS.
- Uses parallel processing to enhance throughput.

Architectural Models of Parallel Databases

1. Shared Memory Architecture (SMP)

In this architecture, multiple CPUs share a single global memory and disk system.

Characteristics

- Single copy of OS and DBMS.
- CPUs communicate through shared memory.
- Tightly coupled system.

Advantages

- Fast data access.
- Efficient communication between processors.

Disadvantages

- Limited scalability (about 80–100 CPUs).
- Interconnection network becomes a bottleneck.

2. Shared Disk Architecture

Each CPU has its own memory, but all CPUs share common disk storage.

Characteristics

- Each node runs its own OS and DBMS.
- Loosely coupled system.
- Also called cluster architecture.

Advantages

- Better fault tolerance.
- Load balancing is easier.

Disadvantages

- Disk contention increases with more CPUs.
- Scalability is limited.

3. Shared Nothing Architecture (MPP)

Each node has its own CPU, memory, and disk, and no resources are shared.

Characteristics

- Nodes communicate via network.
- Known as Massively Parallel Processing (MPP).
- Best scalability.

Advantages

- High scalability.
- Easy to add more nodes.

Disadvantages

- High communication cost.
- Non-local data access is expensive.

2. Query Processing in Distributed DBMS

Overview

- Query processing in DDBMS involves data transmission between sites, making it more complex than centralized DBMS.
- Queries may need data from multiple sites.
- Communication cost plays a crucial role.
- Query optimization aims to minimize data transfer.

Cost of Distributed Query Processing

The major cost is data transfer cost.

Data Transfer Cost Formula

Data Transfer Cost=Data Transfer Cost=$C \times Size$

Where:

C = cost per byte

Size = number of bytes transferred

Query Execution Strategies

- Transfer data from Site S2 to S1.
- Transfer data from Site S1 to S2.
- Transfer data from both sites to a third site.

The optimal strategy depends on:

- Relation size
- Result size
- Communication cost
- Location where result is needed

Semi-Join Technique in Distributed Query Processing

A Semi-Join is an optimization technique used to reduce the number of tuples transmitted between sites.

Working

- Transfer only the join attribute of one relation.
- Perform filtering at the remote site.
- Send back only matching tuples.

Advantages

- Reduces data transfer size.
- Minimizes network cost.
- Improves overall query performance.

Example for query processing:

Scenario:

EMPLOYEE table at Site S1

DEPARTMENT table at Site S2

Query:

SELECT E.NAME, D.DNAME

FROM EMPLOYEE E, DEPARTMENT D

WHERE E.DID = D.DID;

Processing:

- Data is required from two different sites.
- One table must be transferred to the other site for join.

Example for pdbms:

A company has a SALES table with 10 million records.

Query:

SELECT SUM(amount) FROM SALES;

Parallel Execution:

- The SALES table is divided into 4 partitions.
- Each partition is processed by a different CPU simultaneously.
- Each CPU calculates a partial sum.
- Final result = sum of all partial sums.

Result:

Query executes much faster than sequential processing.

UNIT 2

1. EXPLAIN THE CRYPTOGRAPHIC ALGORITHMS

- Cryptography is the technique of securing information by converting it into an unreadable form.
- Encryption encodes a message so that only authorized users can read it, while decryption converts it back to readable form.
- Modern cryptographic algorithms use keys, which are secret parameters that control encryption and decryption.
- A cryptographic key is essential because without it, encrypted data cannot be decoded.

Types of Cryptographic Algorithms

1. Symmetric Key Cryptography

In symmetric key cryptography, the same secret key is used for both encryption and decryption by the sender and receiver.

Working

- Sender encrypts data using a shared secret key.
- Receiver decrypts data using the same key.

Advantages

- Fast and efficient.
- Suitable for large volumes of data.

Disadvantages

- Secure key distribution is difficult.
- If the key is compromised, security is lost.

Examples

- DES (Data Encryption Standard)
- AES (Advanced Encryption Standard)

2. Hash Functions

Definition

- Hash functions convert data into a fixed-length hash value without using any key.
  The original data cannot be recovered from the hash value.

Working

- Input data → hash algorithm → fixed-size hash value.
- Same input always produces the same hash.

Advantages

- Ensures data integrity.
- Fast computation.

Disadvantages

- No encryption or decryption possible.
- Cannot retrieve original data.

Examples

- MD5
- SHA-256

3. Asymmetric Key Cryptography

Definition

Asymmetric cryptography uses two different keys:

Public Key – used for encryption

Private Key – used for decryption

Working

- Sender encrypts message using receiver’s public key.
- Receiver decrypts it using private key.

Advantages

- Secure key exchange.
- Supports authentication and digital signatures.

Disadvantages

- Slower than symmetric encryption.
- Requires more computation.

Examples

- RSA
- Diffie–Hellman

Uses of Cryptography

- Confidentiality – Protects data from unauthorized access.
- Integrity – Ensures data is not altered during transmission.
- Authentication – Verifies identity of users.
- Non-repudiation – Prevents denial of sent messages.
- Secure Communication – Used in SSL/TLS, emails, and online transactions.

2. EPLAIN THE DIGITAL SIGNATURE AND THE DISTRIBUTED DEADLOCKS

A digital signature is a cryptographic technique used to verify the authenticity, integrity, and non-repudiation of digital messages or documents. It assures the receiver that the message was sent by a known sender and was not altered during transmission.

Working of Digital Signature

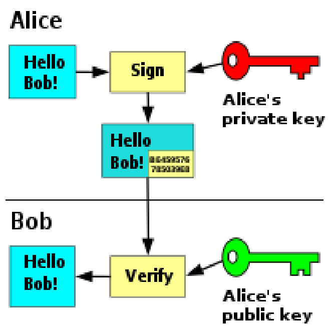
- Digital signatures use asymmetric cryptography.
- The sender creates a signature using their private key and the message.
- The receiver verifies the signature using the sender's public key.
- If verification is successful, the message is authentic and unmodified.

Features of Digital Signatures

- Ensures authentication of the sender.
- Provides data integrity.
- Supports non-repudiation (sender cannot deny sending).
- Difficult to forge compared to handwritten signatures.
- Can include timestamping for added security.

Applications of Digital Signatures

- Software distribution
- Online financial transactions
- Electronic contracts
- Secure e-mail communication
- E-commerce systems



A digital signature works by allowing the sender (Alice) to sign a message using her private key. The signed message is then sent to the receiver (Bob).

Bob uses Alice's public key to verify the signature. If the verification is successful, Bob is sure that the message was sent by Alice and that it was not altered during transmission, thus providing authentication, integrity, and non-repudiation.

Distributed Deadlocks

A distributed deadlock occurs in a Distributed DBMS when transactions executing at different sites wait indefinitely for each other to release locks on data items.

Distributed Deadlock Detection Techniques

1. Centralized Deadlock Detection

- One site act as a central deadlock detector.
- All sites periodically send their LWFG updates.
- The detector constructs the GWFG and checks for cycles.

Advantages:

- Simple design
- Easy cycle detection

Disadvantages:

- High communication overhead
- Single point of failure

2. Hierarchical Deadlock Detection

- Deadlock detectors are organized in levels.
- Local deadlocks are detected at local sites.
- Global deadlocks are detected at higher-level detectors.

Advantages:

- Reduced communication cost
- No single central site dependency

Disadvantages:

- Complex implementation
- Requires modification of transaction managers

3. Distributed Deadlock Detection

- Each site maintains its own LWFG.
- Sites exchange information about waiting transactions.
- Deadlocks are detected cooperatively.

Advantages:

- No single point of failure
- Better scalability

Disadvantages:

- Complex coordination
- Higher algorithmic complexity

3. EXPLAIN THE DISTRIBUTED CONCURRENCY CONTROL

Distributed Concurrency Control ensures that multiple transactions executing at different sites of a distributed database system can run simultaneously without violating ACID properties, especially isolation and consistency.

Its main goal is to guarantee serializability while resolving read–write and write–write conflicts among transactions.

Need for Concurrency Control

- Concurrency control is required to:
- Enforce isolation among transactions
- Preserve database consistency
- Resolve read–write and write–write conflicts

Locking-Based Concurrency Control

- In locking-based protocols, a lock is associated with each data item to control access.
- Read lock (shared lock): Compatible with another read lock
- Write lock (exclusive lock): Not compatible with read or write locks
- Lock compatibility is determined using a lock compatibility matrix

One-Phase Locking Protocol

- A transaction locks a data item before use and releases it immediately after use
- Provides high concurrency
- Does not guarantee serializability

Two-Phase Locking (2PL) Protocol

In 2PL, a transaction has two phases:

Growing (Expanding) Phase

Transaction acquires all required locks

No locks are released

Shrinking Phase

- Transaction releases locks
- No new locks can be acquired
- 2PL ensures conflict serializability.

Centralized Two-Phase Locking (C2PL)

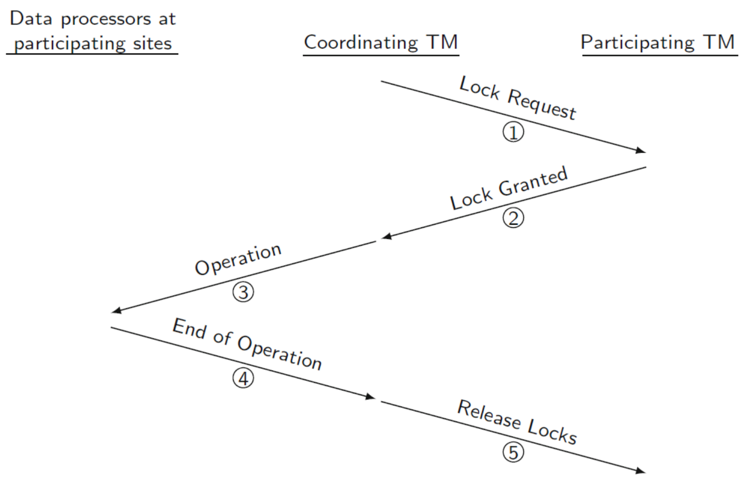
- A single site acts as the central lock manager
- All lock requests from other sites are sent to this site
- Also known as Primary Site 2PL

Advantages:

- Simple design
- Easy deadlock detection

Disadvantages:

- Single point of failure
- Communication overhead



Distributed Two-Phase Locking (D2PL)

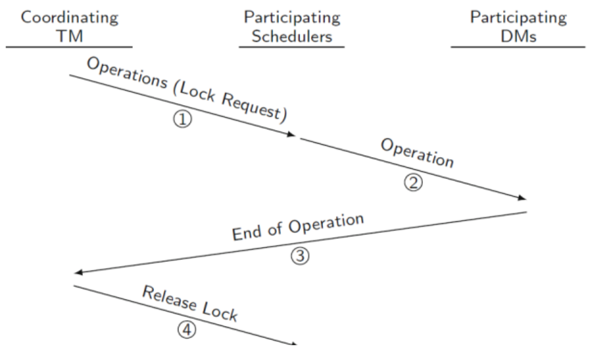
- Each site has its own local lock manager
- Lock requests are sent to all participating sites
- Lock managers directly communicate with data processors

Advantages:

- No single point of failure
- Better scalability

Disadvantages:

- Increased communication cost
- More complex implementation



UNIT 3

1. EXPLAIN THE AUTHENTICATION IN DISTRIBUTED DATABASES

Authentication in a distributed database/system is the process of verifying the identity of a principal (user, process, client, or server) before allowing access to data or services.

Need for Authentication

- Distributed systems are open to intruders and impersonation attacks.
- It is required for authorization (what a user can do) and accounting (tracking usage).
- A principal may try to impersonate another principal, so identity verification is essential.

What is Authentication?

- Authentication = Identification + Verification
- Identification: A principal claims an identity (e.g., username).
- Verification: The system checks whether the claim is valid (e.g., password check).
- Authentication is based on secret information like passwords, keys, or cryptographic credentials.

Types of Authentication

One-way authentication

Only one principal verifies the identity of the other (e.g., user login).

Mutual authentication

Both parties verify each other’s identity (e.g., client and server).

Types of Authentication in Distributed Systems

Message Content Authentication

- Ensures the message is not altered in transit.
- Done using Message Authentication Codes (MACs).

Message Origin Authentication

Verifies that the message is sent by the claimed sender.

General Identity Authentication (GIA)

- Verifies that a principal’s identity is genuine.
- Used for authorization and accounting.

Types of Principals

- Hosts: Network-level entities identified by IP address or hostname.
- Users: Humans or accounts responsible for system activities.
- Processes: Programs acting on behalf of users inside the system.

Authentication Protocols

- Authentication is carried out using authentication protocols, which involve message exchanges between principals.
- These protocols use cryptography to securely prove identity.

Classification of Authentication Protocols

- Authentication protocols can be classified based on:
- Cryptography used: Symmetric or Asymmetric
- Authentication type: One-way or Mutual

Key exchange method

- Third-party involvement: Online or Offline
- Trust and security guarantees
- Storage of secrets

Cryptographic Techniques Used

Symmetric Key Cryptography

- Same secret key used for encryption and decryption.
- Fast but has key distribution problems.

Asymmetric Key Cryptography

- Uses public key and private key.
- Public key encrypts; private key decrypts.
- More secure and scalable.
- Example: Login Authentication
- User sends username and password (or hashed password).
- System verifies using a one-way function.
- If verified, access is granted.

$$\begin{array}{rl}
U \rightarrow H: & U \\
H \rightarrow U: & \text{"Please enter password"} \\
U \rightarrow H: & p \\
H: & \text{compute } y = f(p) \\
: & \text{Retrieve user record } (U, f(\text{password}_U)) \text{ from the database} \\
: & \text{If y } = f(\text{password}_U)\text{, then accept; otherwise reject}
\end{array}$$

2. EXPLAIN THE CHORD DISTRIBUTED HASH TABLE AND CAN

Chord Distributed Hash Table (DHT)

Chord is a structured Peer-to-Peer (P2P) protocol that efficiently maps keys (data objects) to nodes using a flat identifier space and consistent hashing.

Key Ideas of Chord

- Both nodes and data objects are hashed into the same m-bit identifier space using a hash function.
- Identifiers range from 0 to $2^m - 1$, arranged in a logical ring.
- Each data key k is stored at the node with identifier succ(k) (the first node whose ID $\geq$ k).

Chord Lookup Operation

- Chord supports one main operation:
- lookup(key) → finds the node responsible for that key.

Simple Lookup

- Each node stores only its successor.
- Query is forwarded node by node around the ring.

Cost:

- Space: O(1)
- Time: O(n) hops (slow)
- Scalable Lookup using Finger Table
- Each node maintains a finger table with up to m = O(log n) entries.

Finger table entry:

- finger[i] = succ($n + 2^{i-1}$)
- Lookup jumps closer to the destination key each time.

Cost:

- Space: O(log n)
- Lookup time: O(log n) hops

Advantages of Chord

- Highly scalable
- Efficient key lookup
- Low overhead when nodes join or leave (only O(1/n) keys move)

Limitations

- Single logical ring
- Limited fault tolerance compared to CAN

CAN (Content Addressable Network)

CAN is a distributed indexing and lookup system that maps keys to nodes using a multi-dimensional virtual coordinate space.

Key Ideas of CAN

- Uses a d-dimensional Cartesian space (e.g., 2D, 3D).
- The space is organized as a d-torus (wrap-around edges).
- The space is partitioned into regions, each owned by a node.
- A key is hashed to a point in the coordinate space.
- The node owning that region stores the key-value pair.

Core Operations

- Insert
- Search
- Delete

CAN Initialization (Node Join)

- Node contacts a bootstrap node.
- Picks a random point in the coordinate space.
- Routes request to the node owning that region.
- That node splits its region and gives half to the joiner.
- Neighbor tables are updated.
- Only neighbor nodes are involved → low overhead.

CAN Routing

- Uses greedy routing in Euclidean space.
- Each node maintains neighbors that share a boundary.
- Message is forwarded to the neighbor closest to destination point.

CAN Performance

- Average neighbors per node: O(d)
- Average path length: $O(n^{1/d})$
- Join cost: O(d)
- Fault tolerance is high due to multiple routing paths

Advantages of CAN

- Good fault tolerance
- Load evenly distributed
- Neighbor state independent of total network size

Limitations

- Lookup slower than Chord for low dimensions
- More complex routing logic

3. EXPLAIN THE GRAPH STRUCTURES OF COMPLEX NETWORK

Peer-to-Peer (P2P) overlay networks are large-scale distributed systems that grow dynamically without centralized control. Understanding their graph structure is important to analyse scalability, robustness, routing efficiency, and fault tolerance. Such networks belong to a broader class of complex networks, which include the World Wide Web, Internet topology, social networks, biological networks, and power grids.

1. Random Graph Model (Erdős–Rényi Model)

- The first attempt to model large uncontrolled networks was the random graph model, where:
- The network has n nodes.
- Each pair of nodes is connected with probability p.
- The expected number of edges is $n(n-1)p / 2$.

Properties:

- Average path length grows as $O(\log n)$ → small-world behaviour.
- Degree distribution follows a Poisson distribution.
- Clustering coefficient is low (equal to p).

Limitation:

- Real networks show high clustering and hubs, which random graphs cannot explain.
- Hence, random graphs are insufficient to model P2P overlays.

2. Small-World Networks

- A key property observed in many real networks is the small-world phenomenon, where:
- Any two nodes can be reached using a small number of hops, even in large networks.
- Popularized by Milgram's "six degrees of separation".

Characteristics:

- Short average path length (similar to random graphs).
- Much higher clustering than random graphs.

Examples:

- Social networks
- World Wide Web
- Internet graphs
- P2P overlays

3. Clustering in Networks

- Clustering refers to the tendency of nodes to form tightly connected groups or cliques.
- Measured using the clustering coefficient.
- High clustering means neighbours of a node are likely connected.

Observations:

- Real networks (social, P2P, biological) have high clustering.
- Random graphs have very low clustering, making them unrealistic models.

4. Degree Distribution and Scale-Free Networks

Let P(k) denote the probability that a node has k connections.

In many real networks:

- P(k) follows a power-law distribution.
- Few nodes (hubs) have very high degree.
- Most nodes have small degree.
- Such networks are called scale-free networks because:
- The degree distribution is independent of network size.
- No characteristic scale exists.

Examples:

- WWW
- Internet Autonomous Systems (AS)
- Social networks
- P2P networks

Advantages:

Robust against random node failures.

Disadvantage:

Vulnerable to targeted attacks on hubs.

5. Empirical Observations of Real Networks

Different real-world networks show combinations of these properties:

Network	Degree Distribution	Small World	Clustering
WWW	Power law (in & out)	Yes	High
Internet (AS)	Power law	Yes	High
Social networks	Power law	Yes	High
Protein networks	Power law with cutoff	Yes	Moderate
Power grid	Exponential	No	Low

6. Implications for P2P Overlay Design

Understanding graph structures helps in:

- Designing efficient routing protocols
- Improving scalability
- Enhancing fault tolerance
- Handling node churn
- Structured overlays like Chord, CAN, Pastry, and Tapestry are designed to achieve:
- Small path lengths
- Controlled degree
- Balanced load distribution