Iis (4311602) - Problem Solver

Antigravity AI

June 4, 2026

Iis

First Edition: 2026

Author: Antigravity AI
Website: milav.in
YouTube: @milav.dabgar

To my students,
who constantly inspire me to find new analogies,
break down complex theories, and make learning
an engineered science.

Contents

Preface

About this Book

Welcome to **Computer Networks**! This textbook is meticulously designed to serve as a comprehensive problem solver and study companion. It provides a robust and intuitive foundation in the core principles of this subject, bridging the gap between theoretical models and practical applications. A unique feature of this book is its focus on decoding complex concepts using clear, step-by-step methodologies and real-world analogies.

Target Audience

This book is intended for students and professionals looking to master the concepts of Computer Networks. Whether you are revising for exams, seeking to solidify your fundamental understanding, or looking for practical examples to clarify abstract concepts, this book has been structured to support your learning journey. It serves as an accessible guide designed to remove the fear of complex topics, replacing it with an appreciation for the underlying mechanics.

Scope and Coverage

The coverage is divided logically to follow standard academic curriculums. Each chapter focuses on key topics, breaking them down into digestible sections:

- **Theoretical Insights** – Core concepts explained intuitively.
- **Method Blocks** – Standardized, step-by-step procedures for solving specific types of problems.
- **Worked Examples** – Fully solved problems that apply the methods in practice.

We hope this resource proves invaluable in your studies and helps you achieve excellence in **Computer Networks**. Happy learning!

Acknowledgments

Writing a comprehensive book that connects the fundamental forces of Chemistry with practical engineering applications requires more than just academic knowledge—it requires a community of support. I would like to express my sincere gratitude to everyone who contributed to the realization of this project.

Specially, I extend my heartfelt gratitude to the **Department of Technical Education (DTE), Government of Gujarat**, and **Government Polytechnic, Palanpur**, for providing an environment that fosters innovation, academic rigor, and continuous professional development.

I am deeply thankful to my family for their unwavering patience and support during the countless hours spent researching, formatting, and refining these chapters.

Finally, my deepest appreciation goes to my students. Your curiosity, challenging questions, and continuous feedback are the true driving force behind the unique analogies and streamlined structure of this book. This textbook was engineered for you.

Antigravity AI

About the Author

Milav Jayeshkumar Dabgar is an engineering educator and R&D professional with over 9 years of experience in electronics hardware development, embedded systems, AI/ML, and full-stack software engineering. He currently serves as a **Lecturer (Class - II) & Innovation Leader** at Government Polytechnic, Palanpur, under the Education Department of the Government of Gujarat.

Milav holds a Master of Engineering (ME) in Communication Systems from L.D. College of Engineering and a Bachelor of Engineering (BE) in Electronics & Communication. He is also currently pursuing a **BS in Data Science and Applications from IIT Madras**, where he has completed both the **Diploma in Programming** and the **Diploma in Data Science** with distinction.

Most recently, he completed the **AICTE QIP PG Certificate Program in Deep Learning from SVNIT Surat** with a **perfect 10/10 CGPA**, topping his batch.

A dedicated mentor, Milav has guided numerous student teams in securing over **INR 45 lakhs** in innovation funding, including INR 25 lakhs from Shark Tank India and INR 20 lakhs through iHub Gujarat, resulting in two student patents. He is recognized as an **NPTEL Evangelist** and **NPTEL Discipline Star** in Computer Science, reflecting his commitment to continuous learning and academic excellence.

His technical expertise spans from embedded systems (8051, STM32, Arduino) to modern web technologies (Next.js, Python, PostgreSQL) and Data Science (TensorFlow, PyTorch). Through this textbook, he aims to simplify complex Engineering Chemistry concepts by integrating relatable, real-world analogies drawn from modern tech and engineering landscapes.

Milav is also an active content creator, running a popular **YouTube channel** (@milav.dabgar) where he shares technical tutorials and academic resources. He maintains a personal blog and resource hub at milav.in and can be reached for professional collaborations on LinkedIn.

CHAPTER 1

Basics of Information System

1.1 Information Concepts and Importance

Information Systems (IS) form the backbone of modern organizational and personal computing. The transition from raw data to actionable knowledge is a fundamental computational process.

Methodology:

Data to Knowledge Conversion Process The process of deriving meaning from raw observations involves the following hierarchical steps:

1. **Data Collection**: Gather raw, unorganized facts, numbers, or symbols.
2. **Data Processing**: Apply structuring, filtering, and aggregation to convert Data into **Information**.
3. **Knowledge Formation**: Apply context, rules, and experience to Information to create **Knowledge**.

Worked Example:

Classifying Data Information and Knowledge Given the following items, classify them into Data, Information, or Knowledge: 1. ‘38, 40, 39, 41‘ 2. "The daily high temperatures in Celsius for Ahmedabad over the last four days were 38, 40, 39, and 41." 3. "Since the temperature consistently exceeds 38 degrees Celsius, we should schedule outdoor work early in the morning."
Solution:

1. **Data**: The raw numbers (‘38, 40, 39, 41‘) lack context.
2. **Information**: The sentence provides context (temperatures in Ahmedabad), giving meaning to the numbers.
3. **Knowledge**: The actionable decision (scheduling outdoor work) based on the temperature trend demonstrates applied understanding.

Methodology:

Identifying Components of an Information System An Information System can be decomposed into five structural components. To analyze any IS, identify the following:

1. **Hardware**: Physical machinery and devices.
2. **Software**: Operating systems and application programs.
3. **Data**: The raw material processed by the system.
4. **Procedures**: The rules, policies, and methods for operating the system.
5. **People**: The end-users and administrators.

Worked Example:

Decomposing an ATM System Identify the five components of a Bank ATM (Automated Teller Machine) Information System.
Solution:

1. **Hardware**: The physical ATM machine, card reader, keypad, screen, cash dispenser, and network router.
2. **Software**: The ATM operating system and the banking application interface.
3. **Data**: Customer account numbers, PINs, transaction amounts, and account balances.
4. **Procedures**: The security protocols, PIN verification steps, and daily cash-loading rules.
5. **People**: The banking customers withdrawing cash and the bank technicians maintaining the machine.

1.2 Hardware Components of Computer System

Understanding hardware requires analytical skills in categorizing components based on their function (input, output, processing, storage) and calculating memory capacities.

Methodology:

Memory Capacity and Storage Calculation Primary memory (RAM, ROM) and secondary memory (HDD, SSD, Flash drives) capacities are measured in bytes. To calculate storage requirements:

1. Identify the base unit: 1 Byte = 8 bits.
2. Apply conversion factors:
   - 1 Kilobyte (KB) = 1024 Bytes
   - 1 Megabyte (MB) = 1024 KB
   - 1 Gigabyte (GB) = 1024 MB
   - 1 Terabyte (TB) = 1024 GB
3. **Calculation**: Total Files = $\frac{\text{Total Storage Capacity}}{\text{Size per File}}$

Worked Example:

Calculating Storage Space A student has a USB Flash Drive with an available capacity of 16 GB. They need to store video lectures, where each lecture file is exactly 400 MB. How many full lectures can be stored on the drive?
Solution:

1. Convert the total capacity from GB to MB to match the file unit:
$$16 \text{ GB} = 16 \times 1024 \text{ MB} = 16384 \text{ MB}$$
2. Divide the total capacity by the size of a single file:
$$\text{Number of lectures} = \frac{16384 \text{ MB}}{400 \text{ MB/lecture}}$$
3. Calculate the result:
$$16384/400 = 40.96$$
4. Since we can only store full lectures, take the integer part. The drive can store **40 full video lectures**.

Methodology:

Hardware Classification and Interconnection To systematically analyze a computer motherboard and its peripherals:

1. **Identify the Device**: Determine the physical component.
2. **Classify the Role**: Categorize as Input, Output, Processing (CPU), Primary Memory, or Secondary Storage.
3. **Determine the Interface**: Identify how it connects to the motherboard (e.g., PCIe, SATA, USB, HDMI, DIMM slots).

Worked Example:

Categorizing Computer Peripherals Classify the following components according to their role and standard motherboard interface: (1) RAM, (2) Solid State Drive (SSD), (3) Mechanical Keyboard, (4) Monitor.
Solution:

- **RAM**: Role → Primary Memory. Interface → DIMM slot on the motherboard.
- **Solid State Drive (SSD)**: Role → Secondary Storage. Interface → SATA port or M.2 NVMe slot.
- **Mechanical Keyboard**: Role → Input Peripheral. Interface → USB port (I/O panel).
- **Monitor**: Role → Output Peripheral. Interface → HDMI or DisplayPort (on GPU or I/O panel).

1.3 Search Engines and Digital Platforms

Search engines and digital platforms involve query formulation and logical workflows to retrieve information or execute digital transactions.

Methodology:

Advanced Google Search Query Formulation To filter and refine search results systematically, construct search queries using specific search operators:

1. **Exact Match**: Wrap terms in double quotes to find exact phrases. (e.g., `"Information System"`)
2. **Site Restriction**: Use `site:[domain]` to search only within a specific website or top-level domain. (e.g., `site:edu` or `site:gov.in`)
3. **File Type**: Use `filetype:[extension]` to search for specific document formats like PDF, PPT, or DOCX. (e.g., `filetype:pdf`)
4. **Exclusion**: Use a hyphen `-` immediately before a word to exclude it from the results. (e.g., `-Wikipedia`)
5. **Logical OR**: Use `OR` (capitalized) to search for pages that contain at least one of multiple terms.

Worked Example:

Constructing a Complex Search Query Formulate a Google search query to find PDF presentations about either "Motherboard" or "Peripherals", specifically from academic institutions (using the '.edu' domain), ensuring that results from "Amazon" are excluded.
Solution:

1. **Identify Keywords and OR logic**: `Motherboard OR Peripherals`
2. **Apply File Type operator**: `filetype:pdf`

3. **Apply Site Restriction**: `site:edu`
4. **Apply Exclusion operator**: `-Amazon`
5. **Combine the query**: `(Motherboard OR Peripherals) filetype:pdf site:edu -Amazon`

Methodology:

Digital Platform Selection and Workflow India's digital infrastructure utilizes various platforms for specific use cases. To solve a user's service requirement:

1. **Analyze the Requirement**: Determine if the need is financial, document storage, transport-related, identity-related, or e-governance.
2. **Select the Platform**:
   - **BHIM / e-RUPI**: For secure UPI-based monetary transactions and digital vouchers.
   - **DigiLocker**: For fetching and storing authentic digital documents (Mark sheets, Aadhar).
   - **m-Parivahan**: For driving licenses (DL) and vehicle registration certificates (RC).
   - **NSDL**: For PAN card issuance and income tax services.
   - **Digital Gujarat**: For state-level domicile certificates, scholarships, and e-governance.
   - **Passport Seva**: For scheduling appointments and processing passport applications.
3. **Define the Workflow**: Authentication (e.g., Aadhar OTP) → Service Request → Verification → Delivery.

Worked Example:

Selecting the Appropriate Digital Platform A student in Gujarat needs to apply for a state scholarship, securely store their 10th-grade mark sheet, and transfer application fees to a friend's bank account instantly. Identify the correct digital platforms for these tasks.
Solution:

1. **State Scholarship Application**: The student should use the **Digital Gujarat** portal, which handles state-specific scholarships and domicile certificates.
2. **Secure Document Storage**: The student should use **DigiLocker** to fetch and securely store the official digital copy of their 10th-grade mark sheet directly from the education board.
3. **Instant Fee Transfer**: The student should use **BHIM** (or any UPI application) to instantly transfer the application fees directly to their friend's bank account using a UPI ID or phone number.

CHAPTER 2
Digital Logic

2.1 Introduction to Digital Computers and Number Systems

Digital computers use the binary number system (base 2) for data representation and processing, as it naturally maps to the two states of electrical switches (ON and OFF). Understanding conversions between decimal (base 10), binary (base 2), octal (base 8), and hexadecimal (base 16) is essential for computational problem-solving in digital logic.

Methodology:

Decimal to Any Base Conversion To convert a decimal integer to any target base (binary octal or hexadecimal), use the repeated division method:

1. Divide the decimal number by the target base (2 for binary, 8 for octal, 16 for hexadecimal).
2. Record the integer quotient and the remainder.
3. Replace the original number with the quotient from step 1.
4. Repeat the process until the quotient becomes 0.
5. The converted number is obtained by reading the remainders in reverse order (from bottom to top).

For hexadecimal, represent remainders 10 through 15 as A through F respectively.

Worked Example:

Convert 156 Decimal to Binary Octal and Hexadecimal **1. Decimal to Binary (Base 2):**

Division	Quotient	Remainder
156/2	78	0 (LSB)
78/2	39	0
39/2	19	1
19/2	9	1
9/2	4	1
4/2	2	0
2/2	1	0
1/2	0	1 (MSB)

Reading bottom to top: $(156)_{10} = (10011100)_2$.
2. Decimal to Octal (Base 8):

Division	Quotient	Remainder
156/8	19	4 (LSB)
19/8	2	3
2/8	0	2 (MSB)

Reading bottom to top: $(156)_{10} = (234)_8$.

3. Decimal to Hexadecimal (Base 16):

Division	Quotient	Remainder
156/16	9	$12 \rightarrow C$ (LSB)
9/16	0	9 (MSB)

Reading bottom to top: $(156)_{10} = (9C)_{16}$.

Methodology:

Any Base to Decimal Conversion To convert a number from any base to decimal, use the positional weight method:

1. Assign a positional weight to each digit of the number, starting from 0 at the rightmost (least significant) digit and increasing by 1 moving left.
2. Multiply each digit by the base raised to the power of its positional weight.
3. Sum all the resulting products. The sum is the decimal equivalent.

Worked Example:

Convert Binary and Hexadecimal to Decimal **1. Convert $(11010)_2$ to Decimal:**

$$\begin{aligned}(11010)_2 &= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\ &= 16 + 8 + 0 + 2 + 0 \\ &= (26)_{10}\end{aligned}$$

2. Convert $(2F5)_{16}$ to Decimal:

$$\begin{aligned}(2F5)_{16} &= 2 \times 16^2 + 15 \times 16^1 + 5 \times 16^0 \\ &= 2 \times 256 + 15 \times 16 + 5 \times 1 \\ &= 512 + 240 + 5 \\ &= (757)_{10}\end{aligned}$$

Methodology:

Conversions Between Binary Octal and Hexadecimal **Binary to Octal:** Group the binary bits into sets of 3, starting from the right. Pad with leading zeros if necessary. Convert each 3-bit group to its decimal/octal equivalent.
Binary to Hexadecimal: Group the binary bits into sets of 4, starting from the right. Pad with leading zeros if necessary. Convert each 4-bit group to its hex equivalent.
Octal/Hexadecimal to Binary: Replace each octal digit with its 3-bit binary equivalent, or each hex digit with its 4-bit binary equivalent.

Worked Example:

Convert between Binary and Hexadecimal **1. Convert $(101101101)_2$ to Octal and Hexadecimal:**
To Octal (3-bit groups):

$$101 \quad 101 \quad 101 \rightarrow 5 \quad 5 \quad 5 \rightarrow (555)_8$$

To Hexadecimal (4-bit groups, pad with zeros):

$$0001 \quad 0110 \quad 1101 \rightarrow 1 \quad 6 \quad 13(D) \rightarrow (16D)_{16}$$

2. Convert $(A3)_{16}$ to Binary:

$$A \rightarrow 1010, \quad 3 \rightarrow 0011 \rightarrow (10100011)_2$$

2.2 Working of Logic Gates

Logic gates are the fundamental building blocks of digital circuits. They perform basic logical functions based on boolean algebra.

Methodology:

Basic Logic Gates AND OR and INVERTER

1. **AND Gate:** Output is HIGH (1) only if ALL inputs are HIGH (1).
   Boolean Equation: $Y = A \cdot B$
2. **OR Gate:** Output is HIGH (1) if AT LEAST ONE input is HIGH (1).
   Boolean Equation: $Y = A + B$
3. **INVERTER (NOT Gate):** Outputs the complement of the input.
   Boolean Equation: $Y = \overline{A}$

A	B	AND $(A \cdot B)$	OR $(A + B)$	NOT $(\overline{A})$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Methodology:

Exclusive Logic Gates XOR and XNOR

1. **XOR Gate (Exclusive-OR):** Output is HIGH (1) if the inputs are DIFFERENT (odd number of 1s).
   Boolean Equation: $Y = A \oplus B = A\overline{B} + \overline{A}B$
2. **XNOR Gate (Exclusive-NOR):** Output is HIGH (1) if the inputs are the SAME (even number of 1s). It is the complement of XOR.
   Boolean Equation: $Y = A \odot B = AB + \overline{A}\overline{B}$

A	B	XOR $(A \oplus B)$	XNOR $(A \odot B)$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Worked Example:

Evaluate Logic Gate Expressions Given inputs $A = 1$, $B = 0$, $C = 1$, evaluate the output of the boolean expression: $Y = (A \oplus B) \cdot \overline{(B + C)}$

Step 1: Substitute the given values into the expression.

$$Y = (1 \oplus 0) \cdot \overline{(0 + 1)}$$

Step 2: Evaluate the XOR operation.

$$1 \oplus 0 = 1$$

Step 3: Evaluate the OR operation inside the complement.

$$0 + 1 = 1$$

Step 4: Evaluate the NOT operation.

$$\overline{1} = 0$$

Step 5: Evaluate the final AND operation.

$$Y = 1 \cdot 0 = 0$$

Final Output: $Y = 0$

2.3 Working of Universal Gates

NAND and NOR gates are known as Universal Gates because any boolean function or logic gate can be implemented using only NAND gates or only NOR gates.

Methodology:

NAND and NOR Universal Gates

1. **NAND Gate (NOT-AND):** Output is LOW (0) only if ALL inputs are HIGH (1). It is an AND gate followed by a NOT gate.
   Boolean Equation: $Y = \overline{A \cdot B}$
2. **NOR Gate (NOT-OR):** Output is HIGH (1) only if ALL inputs are LOW (0). It is an OR gate followed by a NOT gate.
   Boolean Equation: $Y = \overline{A + B}$

A	B	NAND ($\overline{A \cdot B}$)	NOR ($\overline{A + B}$)
0	0	1	1
0	1	1	0
1	0	1	0
1	1	0	0

2.4 Simplification of Basic Logic Gates Using Universal Gates

Universal gates can be interconnected to realize the basic logic functions (NOT, AND, OR). This process requires the application of De Morgan's Laws: $\overline{A \cdot B} = \overline{A} + \overline{B}$ and $\overline{A + B} = \overline{A} \cdot \overline{B}$.

Methodology:

Realization using NAND Gates

1. **NOT Gate:** Tie all inputs of a NAND gate together.
   Equation: $Y = \overline{A \cdot A} = \overline{A}$
2. **AND Gate:** Use a NAND gate followed by a NOT gate (realized by another NAND gate).
   Equation: $Y = \overline{\overline{A \cdot B}} = A \cdot B$
3. **OR Gate:** Invert the inputs using NAND gates, then pass them through another NAND gate.
   Equation: $Y = \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A}} + \overline{\overline{B}} = A + B$

Worked Example:

Derive OR Gate from NAND Gates Show step-by-step how an OR gate function $(A + B)$ is derived using only NAND gates.
Step 1: Invert Input A. Connect input A to both terminals of a NAND gate. Output $Y_1 = \overline{A \cdot A} = \overline{A}$.
Step 2: Invert Input B. Connect input B to both terminals of a second NAND gate. Output $Y_2 = \overline{B \cdot B} = \overline{B}$.

Step 3: Apply NAND to Inverted Inputs. Connect Y_1 and Y_2 to the inputs of a third NAND gate. Final Output $Y = \overline{Y_1 \cdot Y_2} = \overline{\overline{A} \cdot \overline{B}}$.
Step 4: Apply De Morgan's Law. $Y = \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A}} + \overline{\overline{B}} = A + B$.
This proves that three NAND gates can be used to construct a functioning OR gate.

Methodology:

Realization using NOR Gates

1. **NOT Gate:** Tie all inputs of a NOR gate together.
   Equation: $Y = \overline{A + A} = \overline{A}$
2. **OR Gate:** Use a NOR gate followed by a NOT gate (realized by another NOR gate).
   Equation: $Y = \overline{\overline{A + B}} = A + B$
3. **AND Gate:** Invert the inputs using NOR gates, then pass them through another NOR gate.
   Equation: $Y = \overline{\overline{A} + \overline{B}} = \overline{\overline{A}} \cdot \overline{\overline{B}} = A \cdot B$

Worked Example:

Derive AND Gate from NOR Gates Show step-by-step how an AND gate function $(A \cdot B)$ is derived using only NOR gates.
Step 1: Invert Input A. Connect input A to both terminals of a NOR gate. Output $Y_1 = \overline{A + A} = \overline{A}$.
Step 2: Invert Input B. Connect input B to both terminals of a second NOR gate. Output $Y_2 = \overline{B + B} = \overline{B}$.
Step 3: Apply NOR to Inverted Inputs. Connect Y_1 and Y_2 to the inputs of a third NOR gate. Final Output $Y = \overline{Y_1 + Y_2} = \overline{\overline{A} + \overline{B}}$.
Step 4: Apply De Morgan's Law. $Y = \overline{\overline{A} + \overline{B}} = \overline{\overline{A}} \cdot \overline{\overline{B}} = A \cdot B$.
This proves that three NOR gates can be used to construct a functioning AND gate.

CHAPTER 3
Operating System

3.1 General features of OS

Methodology:

Analyzing OS Functions and Services An Operating System (OS) acts as an intermediary between the user and the computer hardware. The fundamental algorithm for OS operation involves managing resources systematically through these core functions:

1. **Process Management:** Allocate CPU time to processes using scheduling algorithms.
2. **Memory Management:** Track memory allocation and deallocation for active programs.
3. **File System Management:** Map logical files to physical disk blocks.
4. **Device Management:** Handle Input/Output (I/O) requests via device drivers.
5. **Security and Protection:** Enforce access controls and user authentication algorithms.

Worked Example:

Mapping User Actions to OS Services **Problem:** A user double-clicks a text file to edit it and then prints the document. Map these sequential actions to the specific OS services involved.
Solution:

1. **File Execution:** The OS intercepts the double-click and uses *Process Management* to create a new process for the text editor application.
2. **Loading to RAM:** The OS uses *Memory Management* to allocate RAM and load the editor's binary instructions from the disk into memory.
3. **Opening File:** The OS invokes *File System Management* to locate the text file on the hard drive and stream its contents into the allocated application memory.
4. **Printing:** When the print command is issued the OS uses *Device Management* to communicate with the printer hardware via drivers and spool the data stream.

3.2 Types of OS

Methodology:

Classifying OS based on Application Requirements Different computational requirements demand specific OS architectures. The standard classification method involves evaluating response time user interaction and resource distribution:

1. **Batch OS:** Processes jobs in bulk without user interaction. Optimized for throughput.

2. **Multitasking Time-Sharing OS:** Divides CPU time among multiple users/tasks sequentially to provide fast interactive response times.
3. **Multiprocessing OS:** Utilizes multiple physical CPUs sharing a common memory space to execute parallel computational tasks.
4. **Real-Time OS (RTOS):** Adheres to strict time constraints. Computations must finish within a defined deadline to prevent system failure.
5. **Distributed OS:** Manages a group of independent network nodes and makes them appear as a single computational system.
6. **Network OS:** Runs on a dedicated server to manage data security and local network routing.
7. **Mobile OS:** Optimized for portable devices prioritizing battery efficiency and wireless connectivity.

Worked Example:

Selecting OS Architecture for Specific Scenarios **Problem:** Determine the most appropriate OS architecture for the following computational systems: (A) Anti-lock Braking System (ABS) in an automobile. (B) A university web server handling thousands of concurrent student requests. (C) Generating monthly bank billing statements for 100000 customers.
Solution:

1. **System A (ABS):** Requires strict algorithmic execution within milliseconds. A delay results in physical failure. **Selection: Real-Time OS (Hard RTOS)**.
2. **System B (Web Server):** Requires handling multiple independent tasks simultaneously while sharing CPU and memory resources evenly. **Selection: Multitasking Time-Sharing OS**.
3. **System C (Bank Statements):** Requires heavy processing of identical jobs without user intervention. Optimization for throughput rather than immediate response is required. **Selection: Batch OS**.

3.3 Windows and Linux Operating System

Methodology:

Comparing Architecture of Windows and Linux **Microsoft Windows OS (Features and State):**

1. **Evolution:** Transitioned from a command-line MS-DOS interface to a monolithic-hybrid NT architecture.
2. **Current State:** Windows features virtualization-based security GUI-centric management and extensive backwards compatibility.

Linux Operating System Components:

1. **Hardware Layer:** Physical computational components (CPU RAM Storage).
2. **Kernel:** The core interface interacting directly with hardware. Executes scheduling and memory mapping algorithms.
3. **Shell:** Command-line interpreter acting as the interface between the user and the kernel.
4. **System Libraries:** Pre-compiled code modules used by user-space applications to invoke kernel functions.

Worked Example:

Tracing Execution Kernel Mode vs User Mode **Problem:** Explain the dual-mode architectural sequence when a C program executes a `read()` system call to fetch data from the hard disk in Linux.
Solution:

1. **User Mode Execution:** The application executes instructions normally in User Mode with restricted hardware access. It triggers the `read()` system call function.
2. **Mode Switch (Context Switch):** The system call generates a software interrupt (trap). The CPU switches its execution mode bit from User Mode (1) to Kernel Mode (0).
3. **Kernel Mode Execution:** The Linux Kernel assumes control validates the memory addresses accesses the disk controller directly and retrieves the block data into kernel buffer space.
4. **Return to User Mode:** The Kernel copies the data from its buffer to the user application's memory space switches the CPU mode bit back to User Mode (1) and returns control to the application's next instruction.

3.4 Proprietary and Open-source software

Methodology:

Evaluating Software Licensing Models The method for selecting a software ecosystem depends heavily on code accessibility and legal redistribution rights:

1. **Proprietary Software:** Source code is hidden encrypted and legally owned by an entity. Users purchase a license to execute the binary. Algorithmic modifications are strictly restricted.
2. **Open-source Software (OSS):** Source code is publicly accessible. The community can inspect modify recompile and distribute the software under licenses like GPL MIT or Apache.

Worked Example:

Cost and Customization Analysis **Problem:** A startup needs to deploy a highly customized database engine for a unique data-mining algorithm. Compare the implementation steps using Proprietary Software vs Open-source Software.
Solution:

1. **Step 1: Algorithmic Customization**
   Proprietary: Developers cannot modify the core database engine algorithms. They must build inefficient wrappers around the provided API.
   Open-source: Developers can rewrite core engine components (like query optimizers) directly in C/C++ to integrate their algorithm.
2. **Step 2: Cost Calculation**
   Proprietary: Incurs upfront licensing fees per CPU core plus recurring support contracts.
   Open-source: Zero software licensing fees. Financial resources are shifted entirely to internal developer salaries and cloud hosting compute costs.
3. **Step 3: Security Auditing**
   Proprietary: Black-box testing only. Must trust the vendor to patch vulnerabilities.
   Open-source: Internal security engineers can perform white-box source code audits to verify the absence of backdoors.

CHAPTER 4

Information Communication System

4.1 Basic Terminology of Information Communication

The foundation of information communication involves the structured transfer of data between devices. A basic communication system consists of a Source, Transmitter, Transmission Medium, Receiver, and Destination.

4.1.1 Transmission Modes

The directionality of data flow is governed by three primary modes:

1. **Simplex:** Unidirectional communication (e.g. keyboard to monitor).
2. **Half-duplex:** Bidirectional communication but only one direction at a time (e.g. walkie-talkie).
3. **Full-duplex:** Simultaneous bidirectional communication (e.g. telephone network).

4.1.2 Synchronization and Transmission Types

Data can be transmitted sequentially or concurrently:

- **Serial Communication:** One bit is sent at a time over a single wire. It requires less cabling and is better for long distances.
- **Parallel Communication:** Multiple bits are sent simultaneously over multiple wires. It is faster but prone to crosstalk over long distances.
- **Asynchronous Transmission:** Data is sent byte-by-byte. Each byte is framed with a start bit and a stop bit.
- **Synchronous Transmission:** Data is sent as a continuous stream of blocks or frames. The sender and receiver are synchronized by a common clock.

Methodology:

Data Transmission Time Algorithm Use the following steps to compute the time required to transmit data across a network medium.

1. Identify the file size S and convert it into bits (1 Byte = 8 bits).
2. Identify the channel capacity or data rate R in bits per second (bps).
3. For Asynchronous Transmission overhead calculate the effective number of bits S_{eff}. If 1 start bit and 1 stop bit are added per 8-bit character the total bits per character is 10.
4. Calculate the total transmission time $T = \frac{S_{eff}}{R}$.

Worked Example:

Calculating Transmission Time in Asynchronous Serial Link A device is sending a 5 KB text file over a 9600 bps serial link. The communication is asynchronous with 1 start bit and 1 stop bit per 8-bit character. Calculate the total transmission time.

Solution:

1. File size $S = 5$ KB $= 5 \times 1024$ Bytes $= 5120$ Bytes.
2. Data rate $R = 9600$ bps.
3. In asynchronous mode each 8-bit byte requires 1 start bit and 1 stop bit. Thus 1 Byte transmits as 10 bits.
4. Effective data size in bits $S_{eff} = 5120 \times 10$ bits $= 51200$ bits.
5. Transmission time $T = \frac{S_{eff}}{R} = \frac{51200}{9600} \approx 5.33$ seconds.

4.2 Modulation and Multiplexing

4.2.1 Modulation Concept and Types

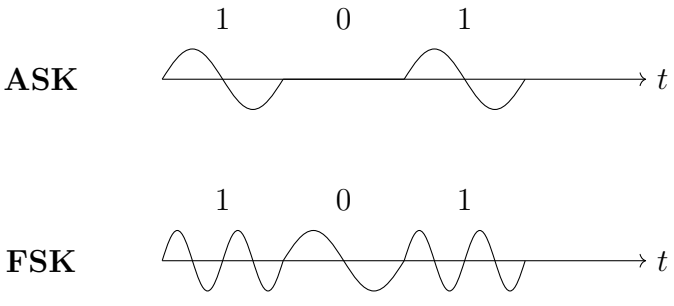
Modulation is the process of varying one or more properties of a periodic waveform (carrier signal) with a modulating signal that contains information to be transmitted. It is required to reduce antenna size minimize interference and allow multiplexing over long distances.

Types of Analog Modulation:

- **Amplitude Modulation (AM):** The amplitude of the carrier wave varies in accordance with the baseband signal.
- **Frequency Modulation (FM):** The frequency of the carrier wave varies in accordance with the baseband signal.
- **Phase Modulation (PM):** The phase of the carrier wave varies in accordance with the baseband signal.

Types of Digital Modulation:

- **Amplitude Shift Keying (ASK):** Binary values are represented by two different amplitudes of the carrier frequency.
- **Frequency Shift Keying (FSK):** Binary values are represented by two different frequencies.
- **Phase Shift Keying (PSK):** Binary values are represented by phase shifts in the carrier signal.
- **Quadrature Amplitude Modulation (QAM):** Combines ASK and PSK to increase data throughput.



4.2.2 Multiplexing

Multiplexing is the set of techniques that allows the simultaneous transmission of multiple signals across a single data link.

- **FDM (Frequency Division Multiplexing):** Divides the total bandwidth into non-overlapping frequency bands.
- **TDM (Time Division Multiplexing):** Divides the transmission time into discrete slots and allocates a slot to each user.
- **OFDM (Orthogonal FDM):** Uses overlapping orthogonal subcarriers to improve spectral efficiency and reduce interference.

Methodology:

TDM and FDM Configuration Algorithms **FDM Bandwidth Requirement Calculation**

1. Identify the number of channels N and bandwidth required per channel B_c.
2. Identify the guard band bandwidth B_g required to prevent interference.
3. Calculate Total Bandwidth: $B_{Total} = N \times B_c + (N-1) \times B_g$.

TDM Bit Rate Calculation

1. Identify the number of incoming lines N and data rate of each line R.
2. Determine frame size (in bits) = $N \times$ bits per slot.
3. Determine frame rate = $\frac{R}{\text{bits per slot}}$.
4. Multiplexed Link Data Rate = Frame Rate $\times$ Frame Size.

Worked Example:

Bandwidth Allocation in Frequency Division Multiplexing Five channels each with a 100 kHz bandwidth are to be multiplexed together. What is the minimum bandwidth of the link if there is a need for a guard band of 10 kHz between the channels to prevent interference?
Solution:

1. Number of channels $N = 5$.
2. Bandwidth per channel $B_c = 100$ kHz.
3. Guard band bandwidth $B_g = 10$ kHz.
4. Total number of guard bands needed = $N - 1 = 5 - 1 = 4$.
5. Total bandwidth $B_{Total} = (5 \times 100 \text{ kHz}) + (4 \times 10 \text{ kHz}) = 500 \text{ kHz} + 40 \text{ kHz} = 540 \text{ kHz}$.

The minimum bandwidth required is 540 kHz.

4.3 Wired Media

Network media provide the physical path for signal transmission.

- **Twisted-pair Cable:** Consists of insulated copper wires twisted together to reduce electromagnetic interference (EMI) and crosstalk. Exists as Unshielded (UTP) and Shielded (STP).
- **Coaxial Cable:** Features a central copper core surrounded by an insulating layer a braided metallic shield and an outer plastic cover. Provides higher bandwidth than twisted pair.
- **Fiber Optics:** Uses glass or plastic threads (core) surrounded by cladding to transmit data as pulses of light via total internal reflection. It is immune to EMI and offers immense bandwidth.
- **RJ-45 Connectors:** Standard 8-pin connectors used primarily for terminating twisted-pair cables in Ethernet networks.

Methodology:

Signal Attenuation Analysis Attenuation is the loss of signal strength over distance. To calculate attenuation or received signal power use the decibel (dB) formula.

1. Identify transmitted power P_{tx} and received power P_{rx} in Watts or milliwatts.
2. Calculate Attenuation in dB: $A = 10 \log_{10} \left(\frac{P_{tx}}{P_{rx}} \right)$.
3. If cable attenuation rating α (in dB/km) and length L (in km) are known total attenuation is $A = \alpha \times L$.
4. Calculate received power if attenuation is known: $P_{rx} = P_{tx} \times 10^{-\frac{A}{10}}$.

Worked Example:

Calculating Received Power in a Fiber Optic Link A fiber optic cable has an attenuation of 0.2 dB/km. A signal is transmitted with an initial power of 2 mW over a 30 km distance. Calculate the received power.
Solution:

1. Determine total attenuation $A = \alpha \times L = 0.2 \text{ dB/km} \times 30 \text{ km} = 6 \text{ dB}$.
2. Setup the power equation: $6 = 10 \log_{10} \left(\frac{2 \text{ mW}}{P_{rx}} \right)$.
3. Divide by 10: $0.6 = \log_{10} \left(\frac{2}{P_{rx}} \right)$.
4. Convert from logarithmic to exponential form: $10^{0.6} = \frac{2}{P_{rx}}$.
5. Calculate $10^{0.6} \approx 3.98$.
6. Solve for P_{rx}: $P_{rx} = \frac{2}{3.98} \approx 0.502$ mW.

The received signal power is approximately 0.502 mW.

4.4 Ethernet Cable Configuration

Ethernet configurations standardizes how wires are mapped to the 8 pins of an RJ-45 connector using specific color codes.

4.4.1 Color Codes and Cable Types

There are two primary wiring standards: **T568A** and **T568B**.

Pin	T568A Color	T568B Color
1	White/Green	White/Orange
2	Green	Orange
3	White/Orange	White/Green
4	Blue	Blue
5	White/Blue	White/Blue
6	Orange	Green
7	White/Brown	White/Brown
8	Brown	Brown

- **Straight-Through Cable:** Has the same standard (either T568A or T568B) on both ends. Used to connect unlike devices (e.g. PC to Switch).
- **Crossover Cable:** Has T568A on one end and T568B on the other. Used to connect similar devices directly (e.g. PC to PC or Switch to Switch).

Methodology:

Ethernet Cable Wiring Algorithms **Determining Cable Configuration**

1. Identify the two devices to be connected.
2. If the devices belong to different OSI layers (e.g. PC to Switch) select the Straight-Through Cable configuration.
3. If the devices belong to the same OSI layer (e.g. Router to Router Switch to Switch PC to PC) select the Crossover Cable configuration.
4. Strip the outer jacket untwist the pairs and align the wires according to the appropriate color code (T568A or T568B) based on the chosen cable type.
5. Trim wires flush and insert them into the RJ-45 connector before crimping.

Worked Example:

Designing a Direct Link Connection A network engineer needs to directly connect two laptops to transfer a large dataset without using a switch. Determine the type of cable required and the exact pin configurations for pins 1 2 3 and 6 on both ends.
Solution:

1. **Device Identification:** Laptop to Laptop (PC to PC). These are identical devices.
2. **Cable Selection:** A Crossover Cable is required.
3. **Configuration:** One end must be wired using T568A and the other using T568B.
4. **Pin Mapping for End 1 (T568A):**
   - Pin 1: White/Green
   - Pin 2: Green
   - Pin 3: White/Orange
   - Pin 6: Orange
5. **Pin Mapping for End 2 (T568B):**
   - Pin 1: White/Orange
   - Pin 2: Orange
   - Pin 3: White/Green
   - Pin 6: Green

This ensures the transmit pins (1 and 2) on one laptop connect to the receive pins (3 and 6) on the other laptop.

CHAPTER 5
Networking

5.1 The OSI Model

The Open Systems Interconnection (OSI) model standardizes the communication functions of a telecommunication or computing system into seven logical layers.

Methodology:

Analyzing OSI Layers To determine the functions, hardware, and protocols for any OSI layer, use the following structured mapping from bottom to top:

1. **Physical Layer (Layer 1):** Responsible for the transmission and reception of raw bitstreams over a physical medium.
   - *Hardware:* Hub, Repeater, Cables, NIC.
   - *Protocols:* IEEE 802.3, ISDN, DSL.
2. **Data Link Layer (Layer 2):** Responsible for node-to-node data transfer, error detection, and MAC addressing. Data is packaged into *frames*.
   - *Hardware:* Switch, Bridge, NIC.
   - *Protocols:* Ethernet, Point-to-Point Protocol (PPP), ARP.
3. **Network Layer (Layer 3):** Responsible for logical IP addressing and routing packets across multiple networks. Data is packaged into *packets*.
   - *Hardware:* Router, Layer 3 Switch.
   - *Protocols:* IPv4, IPv6, ICMP, IPSec.
4. **Transport Layer (Layer 4):** Provides end-to-end communication, reliability, and flow control. Data is packaged into *segments*.
   - *Hardware:* Gateway.
   - *Protocols:* TCP (Reliable), UDP (Fast).
5. **Session Layer (Layer 5):** Establishes, maintains, and terminates communication sessions between applications.
   - *Protocols:* NetBIOS, RPC, PPTP.
6. **Presentation Layer (Layer 6):** Handles data translation, encryption, decryption, and compression.
   - *Protocols:* SSL, TLS, JPEG, ASCII.
7. **Application Layer (Layer 7):** Provides network services directly to user applications.
   - *Hardware:* End hosts, Firewalls.
   - *Protocols:* HTTP, FTP, DNS, SMTP.

Worked Example:

Identify OSI Layers for Specific Scenarios Identify the OSI layer responsible for the following scenarios and list a protocol associated with it: 1. Converting an electrical signal into bits. 2. Translating a domain name into an IP address. 3. Routing a packet from one continent to another. 4. Ensuring a downloaded file is not corrupted during transit.

Solution:

1. **Converting an electrical signal into bits:** This deals with raw data on a physical medium.
   - Layer: Physical Layer (Layer 1).
   - Protocol: IEEE 802.3 (Ethernet Physical).
2. **Translating a domain name into an IP address:** This is an application-level network service.
   - Layer: Application Layer (Layer 7).
   - Protocol: DNS (Domain Name System).
3. **Routing a packet from one continent to another:** This involves logical addressing and optimal path determination.
   - Layer: Network Layer (Layer 3).
   - Protocol: IPv4 or IPv6.
4. **Ensuring a downloaded file is not corrupted during transit:** This involves end-to-end error recovery and flow control.
   - Layer: Transport Layer (Layer 4).
   - Protocol: TCP (Transmission Control Protocol).

5.2 Network Topologies and Types

A network topology is the physical or logical arrangement of nodes in a network. Networks are also classified by their geographical span.

Methodology:

Evaluating Network Topologies To evaluate or design a network topology, apply the following definitions and computational formulas:

1. **Bus Topology:** All nodes share a single central cable (backbone). A break in the cable brings down the entire network. Number of cables = 1 backbone + n drop lines.
2. **Star Topology:** All nodes connect to a central device (hub or switch). Number of cables = n. If the central node fails, the network fails, but a single link failure only affects one node.
3. **Ring Topology:** Each node connects to exactly two other nodes, forming a closed loop. Number of cables = n.
4. **Mesh Topology:** Every node connects to every other node. It is highly reliable but expensive.
   - Total number of cables in a fully connected mesh = $\frac{n(n-1)}{2}$
   - Number of ports required per node = $n - 1$
5. **Hybrid Topology:** A combination of two or more topologies (e.g. Star-Bus or Star-Ring).

Worked Example:

Calculating Mesh Topology Requirements A company wants to set up a highly reliable fully connected Mesh network consisting of 8 computers. Calculate the total number of cables required and the number of I/O ports needed on each computer.

Solution:

1. Identify the number of nodes: $n = 8$.
2. Use the formula for the total number of cables:

$$\text{Total Cables} = \frac{n(n-1)}{2}$$

$$\text{Total Cables} = \frac{8(8-1)}{2} = \frac{8 \times 7}{2} = \frac{56}{2} = 28$$

3. Use the formula for the number of ports per node:

$$\text{Ports per node} = n - 1 = 8 - 1 = 7$$

A total of 28 cables are needed, and each computer requires 7 dedicated network ports.

Methodology:

Classifying Types of Computer Networks Determine the network type based on geographical span:

1. **LAN (Local Area Network):** Spans a small area like a room, building, or campus (up to a few kilometers). High speed (up to Gbps), low setup cost, owned by a single organization.
2. **MAN (Metropolitan Area Network):** Spans a city or a large campus (up to 50 km). Moderate speed, used to connect multiple LANs. E.g. Cable TV networks.
3. **WAN (Wide Area Network):** Spans countries or continents (thousands of kilometers). Uses public transmission systems (satellites, telephone lines). Lower speed, complex, and expensive. The Internet is the largest WAN.

5.3 Network Addressing and Protocols

An IP address uniquely identifies a device on an IP network. IPv4 uses 32-bit addresses, usually represented in dotted-decimal notation (four 8-bit octets).

Methodology:

Classful IPv4 Addressing Scheme To find the Class, Network ID (NetID), and Host ID (HostID) of an IPv4 address:

1. Examine the first octet of the IP address (Format: `W.X.Y.Z`, where `W` is the first octet).
2. **Class A (1 to 126):** First bit is 0.
   - NetID = First octet (`W`). HostID = Last three octets (`X.Y.Z`).
   - Default Subnet Mask: `255.0.0.0`
3. **Class B (128 to 191):** First bits are 10.
   - NetID = First two octets (`W.X`). HostID = Last two octets (`Y.Z`).
   - Default Subnet Mask: `255.255.0.0`
4. **Class C (192 to 223):** First bits are 110.

- NetID = First three octets (`W.X.Y`). HostID = Last octet (`Z`).
- Default Subnet Mask: `255.255.255.0`

5. **Class D (224 to 239):** Used for Multicasting. No NetID/HostID distinction.
6. **Class E (240 to 255):** Reserved for experimental purposes.
7. **Note:** 127 is reserved for loopback testing (e.g. `127.0.0.1`).

Worked Example:

Determining Class NetID and HostID Given the following IPv4 addresses, determine their Class, default subnet mask, Network ID, and Host ID. 1. `192.168.10.55` 2. `10.45.2.1` 3. `172.16.50.2`
Solution:

1. **IP Address:** `192.168.10.55`
   - First octet is 192. This falls in the range 192-223, so it is **Class C**.
   - Subnet Mask: `255.255.255.0`
   - NetID: The first three octets → `192.168.10`
   - HostID: The last octet → `55`
2. **IP Address:** `10.45.2.1`
   - First octet is 10. This falls in the range 1-126, so it is **Class A**.
   - Subnet Mask: `255.0.0.0`
   - NetID: The first octet → `10`
   - HostID: The last three octets → `45.2.1`
3. **IP Address:** `172.16.50.2`
   - First octet is 172. This falls in the range 128-191, so it is **Class B**.
   - Subnet Mask: `255.255.0.0`
   - NetID: The first two octets → `172.16`
   - HostID: The last two octets → `50.2`

Methodology:

Justifying the Need for IPv6 To evaluate why IPv6 is replacing IPv4, assess the following factors:

1. **Address Space Exhaustion:** IPv4 uses 32 bits, offering $2^{32} \approx 4.3$ billion addresses. IPv6 uses 128 bits, offering $2^{128} \approx 3.4 \times 10^{38}$ addresses, completely solving the shortage.
2. **Header Efficiency:** IPv6 eliminates checksums at the network layer and fixes the base header size, speeding up routing processes.
3. **Security:** IPv6 has native support for IPsec, providing mandatory end-to-end encryption and authentication.
4. **Auto-configuration:** IPv6 supports stateless auto-configuration (SLAAC), allowing devices to generate their own IP address without needing a DHCP server.

5.3.1 IEEE 802 Standards

Methodology:

Identifying IEEE 802 Standards The IEEE 802 committee defines standards for LANs and MANs. Match the standard number to its corresponding function:

1. **IEEE 802.1:** Higher Layer LAN Protocols (Bridging, VLANs).
2. **IEEE 802.3:** Ethernet (Wired LAN using CSMA/CD).
3. **IEEE 802.11:** Wireless LAN (Wi-Fi using CSMA/CA).
4. **IEEE 802.15:** Wireless PAN (Bluetooth, Zigbee).
5. **IEEE 802.16:** Broadband Wireless Access (WiMAX).

5.3.2 DNS Internet and Intranet

Methodology:

Analyzing DNS and Domain Names To systematically analyze the Domain Name System (DNS):

1. **Need for DNS:** Humans remember text names (e.g., `google.com`), but routers and switches route using IP addresses (e.g., `142.250.190.46`). DNS acts as a phonebook, translating domain names into IP addresses.
2. **Domain Name Types:**
   - *Generic TLDs (gTLD):* `.com` (commercial), `.edu` (education), `.org` (organization), `.net` (network).
   - *Country Code TLDs (ccTLD):* `.in` (India), `.uk` (United Kingdom), `.us` (United States).

Methodology:

Decoding URLs and Network Scope To process a Uniform Resource Locator (URL) and network scope:

1. **URL Components:** Break down a URL into: `Protocol :// Subdomain . Domain . TLD : Port / Path`
2. **Internet:** A public, global network of networks accessible to anyone.
3. **Intranet:** A private, internal network restricted to employees or members of a specific organization.
4. **Comparison:**

Feature	Internet	Intranet
Accessibility	Public	Private
Security	Lower	Higher
Size	Global	Local to an organization
Users	Anyone	Authorized members only

Worked Example:

Parsing a URL Structure Given the URL: `https://www.gtu.ac.in:443/syllabus/diploma.pdf` Identify its components based on the URL parsing method.
Solution:

1. **Protocol:** `https` (Hypertext Transfer Protocol Secure)

2. **Subdomain:** `www`
3. **Domain Name:** `gtu`
4. **Domain Type (TLD):** `.ac.in` (Academic institution in India, hybrid of generic and country code)
5. **Port Number:** `443` (Standard port for HTTPS)
6. **Path to File:** `/syllabus/diploma.pdf`

5.4 Networking Devices

Methodology:

Selecting Networking Devices Use the following criteria to select the appropriate networking hardware for a given problem:

1. **Repeater (Layer 1):** Select when a signal needs to be amplified or regenerated to travel a longer distance without degradation.
2. **Hub (Layer 1):** Select for a simple, cheap star-topology setup. It broadcasts all received data to all ports (creates high traffic and collisions).
3. **Switch (Layer 2):** Select for an efficient LAN. It learns MAC addresses and unicasts data only to the specific destination port, significantly reducing collisions.
4. **Bridge (Layer 2):** Select when connecting two LAN segments that use the *same* protocol to divide collision domains.
5. **Router (Layer 3):** Select when connecting *different* networks (e.g., LAN to WAN) and routing packets optimally based on IP addresses.
6. **Gateway (Layer 4-7):** Select when connecting two completely *different architectures* (e.g., connecting a TCP/IP network to an IBM mainframe network).
7. **Modem (Layer 1):** Select to convert digital signals from a computer into analog signals for telephone lines (Modulation) and vice versa (Demodulation).
8. **Wireless Access Point (WAP) (Layer 2):** Select to allow wireless Wi-Fi devices to connect to a wired network.
9. **NIC (Layer 1/2):** Select to physically connect a computer to a network (provides a unique burned-in MAC address).

Worked Example:

Hardware Selection for Network Scenarios Determine the most appropriate networking device for each of the following scenarios: 1. A college wants to connect its computer science department LAN to the mechanical department LAN. Both use Ethernet. 2. An organization needs to connect its local private network to the public Internet. 3. A signal traveling over a 100-meter Ethernet cable is getting weak and needs a boost to travel another 100 meters. 4. An office needs to connect 10 computers together efficiently without wasting bandwidth on broadcasting traffic to every machine.

Solution:

1. **Bridge:** Because both LANs use the same protocol (Ethernet) and a bridge is used to connect similar network segments. (A Switch is also a valid modern answer).
2. **Router:** A router connects different networks together and routes traffic based on IP addresses, making it essential for connecting a LAN to a WAN (Internet).

3. **Repeater:** A repeater regenerates and amplifies weak signals over long physical distances.

4. **Switch:** A switch operates at Layer 2 and sends data only to the specific port where the destination MAC address is located, unlike a hub which blindly broadcasts to everyone.

CHAPTER 6

Information Security

6.1 Need for Information Security

Information security ensures that computational systems and data are protected from unauthorized access modification or destruction.

6.1.1 Basic Terminology

Methodology:

Information Security Concepts Understanding information security requires familiarity with the following core concepts:

1. **Cryptography**: The mathematical technique of securing information and communications through the use of codes so that only those for whom the information is intended can read and process it.
2. **Vulnerability**: A weakness or flaw in a system application or network that can be exploited by a malicious entity.
3. **Threat**: A potential cause of an unwanted incident that may result in harm to a system or organization.
4. **Attack**: An intentional attempt to bypass security controls and exploit a vulnerability to cause harm.
5. **Encryption**: The algorithmic process of converting original data (plaintext) into an unreadable format (ciphertext) using a cryptographic key.
6. **Decryption**: The reverse algorithmic process of converting ciphertext back into its original plaintext format using the correct cryptographic key.

Worked Example:

Classifying Security Concepts in a Scenario An organization stores unencrypted passwords in a database. A hacker discovers this and uses an automated script to extract the passwords. Identify the Vulnerability Threat and Attack in this scenario and propose a cryptographic solution.
Solution:

1. **Identify the Vulnerability**: The weakness is storing passwords in plaintext without protection.
2. **Identify the Threat**: The potential danger is an unauthorized user accessing the database.
3. **Identify the Attack**: The hacker executing an automated script to extract the data is the attack.
4. **Propose a Cryptographic Solution**: The organization should apply an **Encryption** algorithm (or hashing) to the passwords before storing them. When a user logs in the system would use **Decryption** or hash comparison to verify the identity.

6.2 The Principles of Security and the CIA Triad

Methodology:

Applying the CIA Triad Analysis The CIA triad is a framework for evaluating the security posture of an information system. To analyze a system apply these three principles:

1. **Confidentiality**: Ensure that information is not disclosed to unauthorized individuals. Implemented via encryption and access controls.
2. **Integrity**: Ensure that data is accurate and unchanged from its original state. Implemented via hashing and digital signatures.
3. **Availability**: Ensure that information and systems are accessible to authorized users when needed. Implemented via redundant systems and backups.

Worked Example:

Evaluating System Security using the CIA Triad A hospital uses an electronic health record (EHR) system. Analyze the system requirements based on the CIA triad.
Solution:

1. **Confidentiality Step**: Patient medical records must only be viewable by the assigned doctors and nurses. Solution: Implement role-based access control and encrypt database files.
2. **Integrity Step**: A patient's prescribed medication must not be altered by a malicious user. Solution: Implement cryptographic hash functions to verify that records have not been tampered with.
3. **Availability Step**: Doctors in the emergency room must be able to access records 24/7. Solution: Deploy backup servers and uninterruptible power supplies to prevent system downtime.

6.3 Security Services

Methodology:

Implementing Security Services Security services are mechanisms designed to enhance the security of data processing systems and information transfers. Key services include:

1. **Authentication**: The process of verifying the identity of a user or system.
2. **Access Control**: Restricting access to resources based on authenticated identities and permissions.
3. **Data Confidentiality**: Protecting data from passive attacks and unauthorized disclosure.
4. **Data Integrity**: Protecting data against active attacks like modification or insertion.
5. **Non-repudiation**: Preventing either the sender or the receiver from denying a transmitted message.

Worked Example:

Mapping Requirements to Security Services An online banking application needs to securely transfer funds. Map the following requirements to the appropriate security services: 1. The bank must ensure the user is who they claim to be. 2. The user cannot deny sending a money transfer request. 3. The transaction details must not be altered during transit.
Solution:

1. **Requirement 1**: Verifying user identity maps to the **Authentication** service (e.g. using a password and OTP).

2. **Requirement 2**: Preventing the denial of a transaction maps to the **Non-repudiation** service (e.g. using a digital signature).

3. **Requirement 3**: Preventing alteration during transit maps to the **Data Integrity** service.

6.4 Cyberattacks

Methodology:

Identifying Common Cyberattacks A cyberattack targets computer information systems infrastructures or networks. The most common attack vectors include:

1. **Malware**: Malicious software (viruses worms ransomware) designed to disrupt damage or gain unauthorized access.

2. **Man-in-the-middle (MitM) attack**: An attacker secretly relays and possibly alters the communication between two parties who believe they are directly communicating with each other.

3. **Denial-of-service (DoS) attack**: An attack meant to shut down a machine or network making it inaccessible to its intended users by flooding it with traffic.

4. **SQL injection**: The placement of malicious code in SQL statements via web page input allowing the attacker to interfere with the database.

5. **Zero-day exploit**: An attack that occurs on the same day a weakness is discovered exploiting the vulnerability before a fix becomes available.

6. **Phishing**: The fraudulent practice of sending emails purporting to be from reputable companies to induce individuals to reveal personal information.

7. **Password cracking**: The process of attempting to guess passwords given the hash or through brute-force dictionary attacks.

Worked Example:

Classifying Cyberattack Scenarios Identify the type of cyberattack in the following scenarios: 1. An attacker sends an overwhelming number of requests to a web server causing it to crash. 2. A user receives an email that looks exactly like their bank's portal asking them to verify their account details. 3. An attacker intercepts public Wi-Fi traffic to steal session cookies between a user and a website. 4. A user inputs `' OR '1'='1` into a login form to bypass authentication.
Solution:

1. **Scenario 1**: Flooding a server to make it inaccessible is a **Denial-of-service (DoS) attack**.

2. **Scenario 2**: Fraudulent emails attempting to steal credentials is a **Phishing** attack.

3. **Scenario 3**: Intercepting communication between a user and a server is a **Man-in-the-middle (MitM) attack**.

4. **Scenario 4**: Injecting malicious queries to manipulate a database is an **SQL injection** attack.

6.5 Cyber Law: IT Amendment Act 2008

Methodology:

Applying the IT Amendment Act 2008 The Information Technology (Amendment) Act 2008 outlines legal implications for cyber offenses. Key sections include:

1. **Section 66 (Computer Related Offenses)**: Penalizes offenses such as unauthorized access data theft and damaging computer systems. If a person dishonestly or fraudulently does any act referred to in Section 43 they shall be punishable with imprisonment or a fine.
2. **Section 67 (Publishing Obscene Information)**: Penalizes the publishing or transmitting of material containing sexually explicit acts or conduct in electronic form.

Worked Example:

Determining Cyber Law Violations Determine which section of the IT Amendment Act 2008 applies to the following cases: 1. An employee maliciously deletes critical financial records from the company's server before resigning. 2. An individual uploads and shares explicit adult content on a public social media platform.

Solution:

1. **Case 1**: Deleting data without authorization falls under damaging computer systems which is penalized under **Section 66**.
2. **Case 2**: Publishing sexually explicit electronic content is penalized under **Section 67**.

Formulae

Appendix: Key Formulae

This appendix provides a quick reference to the key formulae and mathematical relationships discussed in this book.

Number Systems (Unit 2)

Base-b to Decimal Conversion (Polynomial Expansion)

To convert a number from base b to decimal (base 10), multiply each digit by the corresponding power of the base and sum the results. For an n-digit integer $d_{n-1}d_{n-2}\dots d_1d_0$ in base b:

$$N_{10} = \sum_{i=0}^{n-1} d_i \times b^i = d_{n-1}b^{n-1} + d_{n-2}b^{n-2} + \cdots + d_1b^1 + d_0b^0$$

Variables:

- N_{10}: The resulting decimal value.
- d_i: The digit at position i (0-indexed from right to left).
- b: The base of the original number (e.g., 2 for binary, 8 for octal, 16 for hexadecimal).
- n: The total number of digits in the integer.

Decimal to Base-b Conversion (Repeated Division)

To convert a decimal integer N_{10} to another base b, use the repeated division method. The decimal number is divided by the base b repeatedly, and the remainders are recorded until the quotient becomes zero.

$$\begin{aligned} N_{10} \div b &= Q_0 \quad (\text{remainder } R_0) \\ Q_0 \div b &= Q_1 \quad (\text{remainder } R_1) \\ &\vdots \\ Q_{k-1} \div b &= 0 \quad (\text{remainder } R_k) \end{aligned}$$

The base-b representation is formed by reading the remainders from last to first: $(R_kR_{k-1}\dots R_1R_0)_b$.

Variables:

- N_{10}: The initial decimal number to be converted.
- b: The target base for the conversion.
- Q_i: The quotient obtained at step i.
- R_i: The remainder obtained at step i, representing the digits of the converted base-b number. R_0 is the Least Significant Digit (LSD), and R_k is the Most Significant Digit (MSD).