COMPUTER AND NETWORK SECURITY

FUOTECH Solutions, an IT firm, is facing cyber threats like phishing and unauthorized access. To improve security, they hired a network administrator to protect data and maintain secure network operations.

Based on this scenario, answer the following questions:

1a. What are the three main goals of network security? (3marks)

1. **Confidentiality:** Assures that private or confidential information is not made available or disclosed to unauthorized individuals.
2. **Integrity:** Assures that information and programs are changed only in a specified and authorized manner.
3. **Availability:** Assures that systems work promptly and service is not denied to authorized users.

1b. Define a firewall and explain its role in network security. (2 marks)

Firewall: a firewall is a network security system that acts as a barrier between a trusted internal network and untrusted external networks such as internet.

It monitors and filters incoming and outgoing network traffic based on predefined security rules, effectively controlling what traffic is allowed to pass through.

1c. List four common challenges in computer network security. (2 marks)

Challenges:

1. Evolving Cyber Threats
2. Human Error
3. Lack of Security Awareness
4. Third-Party Risks

1d. Who is a network administrator and mention three of their functions? (4marks)

A network administrator is a person responsible for managing, maintaining, and securing a computer network within an organization.

Functions/Role

- Setting up networks (e.g., LAN, WAN, intranet)
- Installing and configuring network hardware like routers, switches, firewalls
- Monitoring network performance and fixing issues when something goes wrong
- Ensuring network security by managing firewalls, VPNs, and access controls
- Managing user access and permissions
- Backing up data and preparing disaster recovery plans

1e. What do you understand by computer and network security? State three reasons why it is important to keep a network secure. (4 marks)

Computer and network security refers to the practices, technologies, and policies used to protect computer systems and networks from unauthorized access, misuse, data breaches, and other cyber threats.

Three reasons why it is important to keep a network secure:

1. **Protect Sensitive Data:** Prevent unauthorized access to personal, financial, or confidential business information.
2. **Prevent Financial Loss:** Avoid costs associated with data breaches, ransomware attacks, and downtime.
3. **Ensure Business Continuity:** Maintain smooth and uninterrupted network operations by preventing disruptions from cyber threats.

CLIENT- SERVER ANSWERS

(a) Definition and Operation of Client-Server Computing (5 Marks)

Client-server computing is a distributed model where multiple **clients interact with one or more dedicated servers** to access resources, services, or data. In this architecture, the **server acts as a centralized system** providing specific services, while **clients are end-user devices or applications** that request and consume these services.

Typically, the communication occurs over a **computer network** (like the internet or a LAN) using a **request-response pattern**. A client sends a request to the server, which processes the request—possibly querying a database or performing a computation—and sends back a response.

Example: When a user types a URL in a web browser, the browser (client) sends an **HTTP request** to a web server. The server processes the request (e.g., by retrieving data from a database) and returns a web page, which the browser displays to the user.

(b) Roles and Responsibilities of Client and Server (6 Marks)

Server:
A **server** is a dedicated computer system or software application that provides services, data, or resources to clients.
Its key responsibilities include:

- **Processing Requests**: Receives and processes service queries from clients.
- **Resource Management**: Manages access to files, databases, and applications.
- **Security & Control**: Enforces security measures such as authentication and authorization.

Examples of servers include:

- Web servers (e.g., Apache, Nginx)
- Database servers (e.g., MySQL, PostgreSQL)
- File servers

Client:
A **client** is any device or application that initiates communication with a server to use its services.
Its key responsibilities include:

- **Request Initiation**: Sends service requests to the server (e.g., retrieving a webpage).
- **User Interface Presentation**: Displays server responses in a user-friendly format.
- **Local Processing**: May handle basic computations or input validation locally.

Examples of clients include:

- Web browsers (e.g., Chrome)
- Email clients (e.g., Outlook)
- Mobile apps
- Desktop applications

(c) Comparison of Two-Tier, Three-Tier, and N-Tier Architectures (9 Marks)

Two-Tier Architecture:
This model has **two layers**: the **client tier** and the **server tier**. The client handles both the **UI** and part of the **business logic**, while the server manages **data storage** and sometimes the complete business logic.
Example: A desktop application directly communicating with a remote database.
Use Case: Suitable for small business applications with straightforward logic.
Advantages: Simple design and fast development.
Limitations: Poor scalability, security, and maintainability when scaled up.

Three-Tier Architecture:
This architecture introduces an intermediate **application logic layer** between the **presentation (client)** and the **data (server)** layers.

- The **presentation layer** handles UI.
- The **application logic layer** processes data and business rules.
- The **data layer** manages data storage using databases.

Example: Modern web applications where a browser interacts with a web server, which in turn communicates with a database.
Advantages:

- **Scalability**: Each layer can be scaled independently.
- **Modularity**: Changes in one layer have minimal impact on others.
  Use Case: Applications requiring maintenance, scalability, and complex logic.

N-Tier Architecture:
An extension of the three-tier model, it adds more specialized layers such as **authentication, caching, load balancing**, or **API gateways**.
Example: ERP systems used by enterprises, involving layers for security, logging, and resource management.
Advantages:

- **Flexibility**: Manages complex processes and large-scale transactions.
- **Resilience**: Fault tolerance improves system reliability.
  Use Case: Large systems like healthcare platforms, banks, and e-commerce requiring high availability and performance.

DISTRIBUTED COMPUTING

Questions

a. Define distributed computing, explaining how it operates and why it is a useful and widely deployed tool in today's world.
b. Describe three key features that distinguish distributed systems from single-computer systems, providing a brief explanation for each.
c. Explain the fundamental differences between the Client-Server and Peer-to-Peer architectural models in distributed computing, focusing on how resources are shared and managed in each.
d. Discuss three significant benefits that distributed computing offers, providing examples of how these benefits are realized in practical applications.

Answer

a. Distributed computing is the division of a task among multiple computers, or nodes, which collaborate over a network. Each node performs some part of the total computation, and the nodes exchange messages. Practically, a bigger problem is broken down into subtasks, and they are tackled by several machines. The machines are integrated in such a way that the user has just one, unified system.

b. Scalability: This features implies that new nodes, or servers, can be added to the system in an attempt to manage larger loads. Distributed systems accomplish this by scaling horizontally, or adding more machines, which enables the system to increase nearly linearly with the workload increase.

   Fault Tolerance: A distributed system is built to handle single node failure. When a machine in the system fails, all the other nodes keep on running, thus preventing the system from crashing completely.

   Transparency: This feature conceals the fact that the system is distributed from the users. It looks to an end user or a program that is communicating with the system as though it is a unified resource and not a number of discrete machines.

c. **Client-Server** and **Peer-to-Peer (P2P**) are two basic architectural patterns in distributed computing. In **Client-Server Architecture**, the tasks are divided among clients, which are the service requesters, and servers, which are the service providers. Clients make the initial contact by requesting a service, information or computation, from one or multiple servers. Servers execute programs that make resources, for example, files, web pages, or computing resources, available to be accessed by clients.

In contrast, **Peer-to-Peer (P2P) Architecture** gives equal status to all nodes, or peers. In this architecture, every peer is both a resource consumer and provider, meaning there is no central server. Nodes share data directly with one another and also share their own resources like CPU, storage, and bandwidth.

d. Scalability and Elasticity: Distributed systems can add machines to scale up capacity so that they can grow and do more work. Cloud platform, for instance, take advantage of this by provisioning servers on demand to match changing workloads.

   Fault Tolerance and Reliability: Distributed systems remove single points of failure by replicating services and data. When a node fails, other nodes may offload its workload so that the system work goes on uninterrupted. This redundancy increases the overall system reliability by orders of magnitude.

   High Performance: Distributed system tasks execute in parallel across many nodes, generally leading to reduced computation times. For instance, large data processing involves breaking up heavy data analysis tasks into a huge number of machines, which significantly lowers completion time. Load balancing also assists with high performance by not allowing any server to become a bottleneck.

WEB APPLICATION

QUESTIONS

1. Differentiate between a framework and a library and give an example of each.
ANSWER

s/n	Framework	Library
1.	A framework is a complete structure or platform for developing software applications. It dictates the architecture and flow of control of the application.	A library is a collection of pre-written code that developers can call upon to perform specific tasks. It gives more control to the developer.
2.	In a framework, the framework **calls your code** (Inversion of Control).	In a library, **you call the library** functions as needed.

3.	Frameworks often enforce coding standards and project structure.	Libraries are more flexible and can be used anywhere in the code.

Examples of a framework: Angular, Django, Laravel
Examples of library: jQuery, NumPy, Lodash

2. Gmail accessed from the web is a type of what web application? State your reason for your answer.

ANSWER

Gmail accessed from the web is a type of *Single-Page Application (SPA)*

Reason:

Gmail loads the basic structure of the application once, and then dynamically updates the content as the user interacts with it (such as reading emails, composing, switching tabs) without reloading the entire page. This behaviour is a characteristic of SPAs, which provide a smoother, app-like experience in the browser.

3. Web applications are now widely adopted in the modern-day world. Do you agree? Sate your reasons for your answer

ANSWER

Yes, I agree that web applications are now widely adopted.

Reasons:

- **Cross-platform accessibility**: Users can access web applications on any device with a browser—no need for installation.
- **Ease of updates**: Developers can update a web app centrally without requiring users to install updates manually.
- **Cost-effective development**: Building one web app often serves multiple platforms (mobile, desktop).
- **Cloud integration**: Many web apps are cloud-based, making them scalable and accessible from anywhere.
- **Examples in daily life**: Popular tools like Google Docs, Facebook, online banking, e-commerce platforms, and Zoom (web versions) show how integrated web applications have become in our routines.

4. Why is Requirement analysis an important stage to take when building web applications

ANSWER

Requirement analysis is an important stage to when building web applications because It helps to:

- **Understand User Needs:** Requirement analysis helps gather information about what users want the web application to do. This ensures the product is user-centered and solves real problems.

- **Define Clear Project Goals and Scope:** It sets clear expectations by outlining what the application will do and what it won't. This avoids confusion during development and keeps the project focused.

- **Avoid Misunderstandings and Scope Creep:** By documenting requirements early, teams can prevent miscommunication and unexpected feature additions (scope creep) that delay the project or increase costs.

- **Ensure Proper Planning and Resource Allocation:** Knowing what's required helps project managers allocate time, budget, and human resources effectively, which leads to smoother execution.

- **Identify Potential Risks Early:** Requirement analysis can reveal technical, security, or business risks in advance, allowing teams to plan mitigation strategies before problems arise.

- **Lay the Foundation for Design and Development:** All design and coding decisions are based on the requirements. Clear requirements ensure that the system architecture, user interface, and functionality are well-aligned.

- **Ensure the Final Product Meets Client Expectations:** When requirements are gathered and approved upfront, it increases the likelihood that the finished application will match what the client or end-user wanted.

GROUP 2: MOBILE AND WIRELESS COMPUTING

APPLIED QUESTION:

You are developing a mobile application for a logistics company that allows delivery drivers to receive real-time updates, navigate to delivery locations, and report delivery statuses from the field. Considering the advantages and challenges of mobile and wireless computing, explain how mobile computing and wireless technologies would support this application. Also, identify at least two potential issues the company may face and suggest how they can be addressed.

(Note: Pick either solution 1 or 2 for your answer, in case!)

SOLUTION 1

Mobile computing and wireless technologies play a crucial role in supporting the logistics mobile application. Mobile computing allows delivery drivers to use smartphones or tablets to access the app while on the move, enabling them to receive delivery instructions, navigate to customer locations, and update delivery statuses in real time. Wireless technologies such as GPS, 4G/5G mobile networks, and Wi-Fi allow continuous communication with the central system, real-time tracking of vehicles, and instant data exchange without needing any physical connection.

Two potential issues and solutions:

1. Issue: Poor Network Connectivity in Remote Areas
   Delivery drivers may experience weak or no internet signal, which can affect real-time updates and navigation.

Solution:
Implement offline capabilities in the app that allow drivers to access stored maps and delivery details without internet. The app can automatically sync data when the network is available again.

2. Issue: Security Risks and Data Privacy
   Sensitive delivery and customer data could be exposed if the mobile app is not secure.

Solution:
Use data encryption (SSL/TLS) for communication, implement secure login, and apply Mobile Device Management (MDM) tools to protect company data and manage lost or stolen devices.

SOLUTION 2:

Mobile computing allows delivery drivers to use portable devices like smartphones or tablets to work from anywhere. In this logistics app, drivers can receive real-time delivery updates, view customer addresses, and mark deliveries as completed while on the road. **Wireless technologies** like mobile networks (4G/5G) and GPS make it possible for the app to send and receive information instantly and provide accurate navigation directions.

Two potential issues and solutions:

1. **Issue: Network Congestion or Signal Loss**
   If drivers are in areas with poor signal, the app might not update or show maps correctly.

Solution: Add offline features to the app so drivers can still see routes and enter delivery data, which will sync once the connection is restored.

2. **Issue: Security and Data Breaches**
   Drivers may be using personal devices, which could expose customer data to risk.

Solution: Use strong security features like app encryption, secure logins, and regular updates. The company can also provide secured work devices or use Mobile Device Management (MDM) tools.