Institute of Technology of Cambodia
Department of Information Technology and communication

Distributed Systems

Group: C4

Name of students	ID of students
1. Pheng Menghour	e20210737

Lecturer: Mr. Khem Thay

Academic Year 2024-2025

TP1

1. What happens if the request needs to be forwarded from a server to another server?
2. What is peer-to-peer computing, give an example?
3. Explain client-server program between web browser and web server, give an EX
4. Explain the 2 alternative ways for byte ordering of integers.
5. Find 1 EX of each transparency
6. Find functional and nonfunctional properties of a system with EX (QoS)

Answer

1. .If the request need to be forwarded from a server to another server that mean the first server doesn't have data or function in the first server which is received requests from client and it will forward to another severe to find the requirement the request sometime cause it used for improve the performance , Better scalability, Improve security , Software update or moving to new server can be part of relocating data center.
2. Peer to Peer is the network that can share data without a central server. Peer to Peer Arc there are decentralization Arc and Distributed Arc. Advantage of peer-to-peer reducing the load and cost of servers, increasing the availability and reliability of resources, and enhancing the privacy and anonymity of nodes and disadvantage is Network security has to be applied to each computer separately. In company why they use P2P because
   - Speed: P2P networks are faster than traditional networks because they eliminate the need for a central server and allow direct communication between peers.
   - Resilience: P2P networks are more resilient than traditional networks because they don't rely on a single point of failure. If one node goes down, the rest of the network can compensate.
   - Scalability: P2P networks are inherently scalable because new nodes add capacity to the network as they join.
3. The Client-server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters called clients.

- Example:
  - Client Request: User types a URL, the DNS (Domain Name System) is consulted to resolve the domain name into an IP address so the browser knows where to send the request. Once the IP address is resolved, the client (browser) sends an HTTP request to the web server at that IP address. The request includes the URL, HTTP method (like GET or POST), and any headers.
  - Server Response: The server receives the request at the resolved IP address and process it. The server may retrieve files from storage, query a database, or execute scripts to generate a response. The response is send back to the client, including an HTTP status code (200 OK, 404 Not Found), headers (content type), and the requested content (HTML page, JSON data, etc).
  - Client Display: The client (browser) receives the server's response. If it's an HTML file, the browser parses and renders it as a web page. Finally, the user see the content displayed on their screen.

4. Two alternative ways for byte ordering of integers:

- Big-Endian byte ordering: The most significant byte is stored at the lowest memory address. Meaning that the higher-order bytes come first, followed by the lower-order bytes. (Written in human-readable form, from left to right).
- Little-Endian byte ordering: The least significant byte is store at the lowest memory address. Meaning that the lower-order bytes come first, followed by the higher-order bytes. (Written in backward).

5. Example of each transparency:

- Access transparency: enables local and remote resources to be accessed using identical operations.
    - Example: Accessing files from a Network File System. When a user opens a file, they don't need to know whether the file is stored locally or on a remote server. The system hides the underlying access mechanism.
- Location transparency: enables resources to be accessed without knowledge of their physical or network location.
    - Example: A user interacts with a DNS (Domain Name System) to access a website. The website can be hosted on multiple servers in different physical locations, but the user just types the domain name in the browser and is connected to the nearest available server without knowing its location.
- Concurrency transparency: enables several processes to operate concurrently using shared resources without interference between them.
    - Example: In Google Docs, multiple users can edit the same document concurrently. The system synchronizes changes, so users don't have to worry about conflicts or managing concurrent access.
- Replication transparency: enables multiple instances of resources to be used to increase reliability and performance without knowledge of the replicas by users or application programmers.
    - Example: Google Drive store multiple copies of a user's files across different data centers. Users interact with their files as though there is only one version, unaware of the underlying replication.
- Failure transparency: enables the concealment of faults, allowing users and application programs to complete their tasks despite the failure of hardware or software components.
    - Example: In Amazon Web Services (AWS), when a server hosting your application fails, the service automatically transfer the workload to another server or instance. The failure is hidden from users, and the application continues running.
- Mobility transparency: allows the movement of resources and clients within a system without affecting the operation of users or programs.
    - Example: You can switch from Wi-Fi to mobile data during a video call without the call dropping or stopping, even when the network changes.
- Performance transparency: allows the system to be reconfigured to improve performance as loads vary.

  - Example: Cloud auto-scaling on platforms like AWS or Google Cloud. If a wed application experiences a surge in traffic, the cloud platform automatically allocates more resource to maintain performance. Users don't notice any performance degradation.
- Scaling transparency: allows the system and applications to expand in scale without making changes to the system structure or the application algorithms.
  - Example: NoSQL databases like MongoDB or Cassandra scale horizontally by adding more nodes as data or user load increases. Users or applications don't need to be aware of the addition of nodes to the system.

6. Functional and non-functional properties of a system (Facebook):

- Functional properties (What the system does):
  - User Registration and Authentication: New users can sign up, and existing users can log in using email, phone number, or social accounts (e.g., Google).
  - Profile Management: Users can create and edit their personal profiles, including bio, photos, and personal information.
  - Posting and Sharing Content: Users can post status updates, photos, videos, and share links. Includes functionality to comment, like, or react to posts.
  - News feed: A dynamically updated feed of posts from friends, groups, and pages a user follows.
  - Messaging: Direct messaging through Messenger, including text, video calls, and file sharing.
- Non-functional properties (How the system perform, with QoS considerations):
  - Performance:
    - **Latency**: Facebook strives to minimize the time it takes for users' actions (e.g., posting, liking, commenting) to be reflected on the platform.
    - **Throughput**: Facebook handles millions of posts, messages, and interactions per minute. It needs to scale efficiently to manage this load.
  - Scalability:
    - Facebook can scale to accommodate over 3 billion active users, supporting growth in terms of users, data, and traffic.
    - **Horizontal scaling**: Facebook uses distributed systems to ensure high availability globally.
  - **Availability**:
    - Facebook aims for near **100% uptime** and reliability. Users expect to access their accounts and services at all times, from any location.
    - **QoS** for availability often guarantees high uptime percentages (e.g., 99.99%).
  - **Reliability**:
    - Facebook needs to maintain a consistent user experience across devices and regions.
    - **Error Handling**: The system should manage failures (e.g., message sending failure) with appropriate feedback to users.

- **Security**:
  - **Data Privacy**: Protecting users' personal information, following privacy laws like GDPR.
  - **Authentication & Authorization**: Facebook supports two-factor authentication (2FA), ensuring only authorized users can access accounts.
  - **Encryption**: Communication through Facebook Messenger is encrypted, ensuring data privacy.
- **Usability**:
  - **User Interface**: Facebook has an intuitive, responsive design across mobile, desktop, and tablets.
  - **Accessibility**: Facebook aims to be accessible for people with disabilities, offering screen readers, keyboard shortcuts, and alternative text for images.
- **Maintainability**:
  - The platform is built using a modular structure, enabling updates to specific features without disrupting the whole system.
  - **Continuous Deployment**: Facebook employs a system of frequent updates and bug fixes without noticeable downtime.
- **Interoperability**:
  - Facebook integrates with third-party services (like Instagram and WhatsApp, both owned by Meta) for a seamless experience.
  - **API Support**: It offers APIs for developers to create apps or services that interact with Facebook data.
- **Compliance**:
  - Facebook must comply with regulations like GDPR and CCPA, ensuring data handling respects legal requirements.
  - **Content Moderation**: Automatic systems and human reviewers work to ensure compliance with content policies, removing harmful or inappropriate posts.
- **QoS-Specific Properties**:
  - **Service Level Agreements (SLAs)**: Facebook may have internal SLAs for performance, uptime, and fault tolerance, especially with advertisers and partners.
  - **QoS for Content Delivery**: Facebook uses content delivery networks (CDNs) to ensure fast delivery of multimedia content (images, videos) worldwide.
  - **Fault Tolerance**: In case of regional outages, Facebook’s system automatically reroutes traffic or uses backups to maintain service.