

**CLASE 1: ¿Qué es un servidor?**

**1. Definición de servidor**

Un **servidor** es un sistema informático diseñado para proporcionar servicios, recursos o datos a otros sistemas conocidos como **clientes**, a través de una red. Los servidores pueden ser tanto **hardware** (una máquina física) como **software** (una aplicación que ofrece un servicio).

**2. Función principal**

Los servidores están diseñados para:

* Responder a peticiones de los clientes.
* Ofrecer servicios como almacenamiento, páginas web, bases de datos, archivos compartidos, correo electrónico, impresión, etc.

**3. Ejemplos comunes de servidores:**

* **Servidor web (Apache, Nginx)**: entrega páginas web.
* **Servidor de archivos (Samba, FTP)**: permite compartir archivos.
* **Servidor de bases de datos (MySQL, PostgreSQL)**: almacena y gestiona datos estructurados.
* **Servidor de correo (Postfix, Exchange)**: gestiona el envío y recepción de correos electrónicos.
* **Servidor DNS**: traduce nombres de dominio a direcciones IP.

**4. Diferencia entre servidor y cliente**

| **Característica** | **Servidor** | **Cliente** |
| --- | --- | --- |
| Provee recursos | ✅ | ❌ |
| Solicita recursos | ❌ | ✅ |
| Generalmente tiene hardware potente | ✅ | ❌ |

**5. ¿Por qué son importantes los servidores?**

* Centralizan recursos y servicios.
* Mejoran la eficiencia en redes.
* Permiten la escalabilidad (más clientes pueden acceder a más servicios).
* Aseguran la disponibilidad y respaldo de información crítica.

**🧠 PREGUNTAS DE RETROALIMENTACIÓN**

1. **¿Cuál es la principal función de un servidor dentro de una red?**
2. **Menciona tres tipos de servicios que puede ofrecer un servidor.**
3. **¿En qué se diferencia un servidor de un cliente?**
4. **¿Un servidor puede ser solo un software sin un hardware dedicado? Explica.**
5. **¿Por qué es importante centralizar servicios en un servidor en lugar de en cada equipo cliente?**

**CLASE 2: Tipos de servidores (Físicos, Virtuales, Cloud)**

**1. 🔩 Servidor físico (bare-metal)**

**¿Qué es?**

Es un **equipo físico real**, con todos sus componentes dedicados a ejecutar servicios. No comparte sus recursos con otros servidores (a menos que se use virtualización).

**Características:**

* Control total del hardware.
* Alto rendimiento.
* Mayor consumo energético.
* Costoso (hardware, mantenimiento, espacio, personal).
* Ideal para tareas críticas o que requieren gran rendimiento.

**Ejemplos:**

* Centros de datos tradicionales.
* Servidores web o de bases de datos de alta carga en una empresa.

**2. 🧰 Servidor virtual**

**¿Qué es?**

Es una **máquina virtual (VM)** que se ejecuta dentro de un servidor físico mediante un **hipervisor** (como VMware ESXi, Hyper-V o KVM). Simula hardware real.

**Características:**

* Compartición de recursos (CPU, RAM, disco).
* Alta eficiencia de uso del hardware.
* Fácil de clonar, mover o respaldar.
* Permite tener múltiples servidores virtuales en un solo equipo físico.

**Ventajas:**

* Escalabilidad rápida.
* Costos reducidos.
* Flexibilidad para pruebas, staging o desarrollo.

**3. ☁️ Servidor en la nube (cloud server)**

**¿Qué es?**

Es un **servidor virtual o servicio** alojado en una infraestructura de nube pública, privada o híbrida (como AWS, Azure, Google Cloud, Oracle Cloud).

**Características:**

* Acceso bajo demanda.
* Facturación por uso (pago por hora, por minuto, por segundo).
* Alta escalabilidad y disponibilidad.
* Se puede gestionar desde cualquier lugar del mundo.

**Ventajas:**

* Reducción de costos operativos.
* Automatización y monitoreo integrados.
* Backup y recuperación más sencillos.

**4. Comparación rápida**

| **Tipo de servidor** | **Hardware dedicado** | **Escalabilidad** | **Costo inicial** | **Flexibilidad** | **Ejemplo** |
| --- | --- | --- | --- | --- | --- |
| Físico | ✅ | Baja | Alto | Baja | Data center local |
| Virtual | ❌ (compartido) | Media | Medio | Alta | VM en Proxmox |
| En la nube | ❌ (compartido) | Alta | Bajo | Muy alta | AWS EC2 |

**🧠 PREGUNTAS DE RETROALIMENTACIÓN**

1. ¿Qué es un servidor físico y en qué casos es preferido?
2. ¿Cuál es la diferencia principal entre un servidor físico y un servidor virtual?
3. ¿Qué es un hipervisor y qué función cumple en un entorno de virtualización?
4. Menciona dos ventajas de usar servidores en la nube frente a servidores físicos.
5. ¿Cuál sería una desventaja de depender únicamente de servidores virtuales o en la nube?

**CLASE 3: Componentes de hardware del servidor**

Un servidor físico es más robusto que una PC tradicional. Sus componentes están diseñados para **rendimiento constante, alta disponibilidad y tolerancia a fallos**.

**🔩 1. Procesador (CPU)**

* **Función**: Ejecuta instrucciones. En servidores, se usan CPUs de gama alta como Intel Xeon o AMD EPYC.
* **Características clave**:
  + Múltiples núcleos e hilos.
  + Soporte para instrucciones de virtualización.
  + Gran caché y rendimiento sostenido.

**💾 2. Memoria RAM (Memoria de acceso aleatorio)**

* **Función**: Almacena datos temporales que usa el sistema operativo y las aplicaciones.
* **Importante**:
  + Los servidores usan RAM tipo **ECC (Error-Correcting Code)** para evitar errores de memoria.
  + Alta capacidad (32 GB a varios TB).

**📀 3. Almacenamiento (Discos duros o SSD)**

* **Tipos**:
  + **HDD**: Más económicos y con gran capacidad, pero lentos.
  + **SSD**: Más rápidos, ideales para bases de datos o sistemas operativos.
  + **NVMe**: Más rápidos aún que los SSD tradicionales.
* **Sistemas RAID**: Se usan para mejorar rendimiento y tolerancia a fallos (RAID 1, 5, 10, etc.).

**⚡ 4. Fuente de poder (PSU)**

* Diseñadas para trabajar 24/7.
* **Redundantes**: Muchas fuentes de servidores tienen 2 PSUs, por si una falla.

**🌐 5. Interfaces de red (NIC)**

* Los servidores pueden tener múltiples **tarjetas de red (NICs)**.
* Se usan para segmentación de red, balanceo de carga o redundancia (NIC Bonding).

**🌡️ 6. Sistema de enfriamiento**

* Importante para mantener la temperatura en rangos seguros.
* Incluye ventiladores, disipadores, y controladores térmicos.
* En entornos profesionales se usan **racks con refrigeración pasiva o activa**.

**🧱 7. Placa base (Motherboard)**

* Diseñada para soportar CPUs de servidor, gran cantidad de RAM, múltiples discos, varias NICs, etc.
* Tiene características específicas como soporte para IPMI (gestión remota).

**🧊 8. Chasis del servidor**

* Puede ser tipo **rack (1U, 2U, 4U)** o **torre**.
* El tipo rack se monta en gabinetes especiales y es más común en centros de datos.

**📊 Resumen visual**

| **Componente** | **Función clave** |
| --- | --- |
| CPU | Procesamiento de instrucciones |
| RAM ECC | Memoria confiable y estable |
| Almacenamiento | Guarda sistema y datos (HDD, SSD, NVMe) |
| PSU redundante | Suministro eléctrico seguro |
| NICs | Conectividad de red |
| Enfriamiento | Evita sobrecalentamiento |
| Placa base | Interconecta todos los componentes |
| Chasis | Protege y organiza los componentes |

**🧠 PREGUNTAS DE RETROALIMENTACIÓN**

1. ¿Qué tipo de memoria RAM se recomienda para servidores y por qué?
2. ¿Por qué es importante tener fuentes de poder redundantes en un servidor?
3. ¿Qué ventajas ofrecen los discos SSD o NVMe frente a los HDD?
4. ¿Qué función cumple una tarjeta de red (NIC) en un servidor?
5. ¿Para qué se utiliza el sistema RAID en el almacenamiento?

## **CLASE 4: TIPOS DE SISTEMAS OPERATIVOS DE SERVIDOR**

**📌 ¿Qué es un sistema operativo de servidor?**

Es el software base que **gestiona los recursos físicos y virtuales del servidor**, permitiendo que se ejecuten servicios (web, bases de datos, DHCP, etc.) de forma eficiente, segura y continua.

**📚 Principales tipos de sistemas operativos de servidor**

| **Sistema Operativo** | **Características Principales** | **Usos Comunes** |
| --- | --- | --- |
| **Windows Server** | Interfaz gráfica amigable, integración con Active Directory, servicios como IIS, Hyper-V | Empresas con infraestructura basada en Microsoft |
| **Linux (Ubuntu Server, CentOS, Debian, RHEL)** | Ligero, estable, seguro, gratuito (en su mayoría), soporte a servicios web, DNS, DHCP, etc. | Hosting web, bases de datos, firewalls, virtualización |
| **Unix (FreeBSD, Solaris)** | Muy robusto y seguro, orientado a sistemas críticos | Grandes sistemas bancarios, científicos o de telecomunicaciones |
| **VMware ESXi** | Sistema especializado en virtualización (hipervisor bare-metal) | Centros de datos, cloud privada |
| **Proxmox VE** | Basado en Linux, interfaz web para gestionar VMs y contenedores | Laboratorios, pequeñas empresas |
| **MacOS Server (en desuso)** | Antiguamente usado en entornos Apple, limitado hoy | No recomendado para infraestructura moderna |

**📌 Elección del sistema operativo**

Dependerá de:

* **Requerimientos de hardware y software**
* **Nivel de seguridad y estabilidad requerido**
* **Costos (licencias, soporte)**
* **Compatibilidad con las aplicaciones que se van a instalar**

**🔧 Buenas prácticas al elegir un SO de servidor**

* Usar versiones **LTS (Long Term Support)** si es Linux.
* Mantenerlo **actualizado** con parches de seguridad.
* Minimizar servicios instalados para **reducir superficie de ataque**.
* Elegir distribuciones estables y documentadas.

**🛠 INICIO DE PROYECTO PRÁCTICO**

**🎯 Proyecto: Montar un servidor básico local con Linux Ubuntu Server**

**Objetivo:** Instalar Ubuntu Server en una máquina virtual y preparar el entorno para instalar servicios en clases futuras.

**🧰 Requisitos**

| **Herramienta** | **Descripción** |
| --- | --- |
| **VirtualBox** o **VMware Workstation** | Para crear una máquina virtual |
| **ISO de Ubuntu Server (22.04 LTS)** | Imagen del sistema operativo |
| **Al menos 2 GB de RAM y 20 GB de disco** | Para la VM |
| Conexión a Internet | Para descargar actualizaciones |

**🔨 Pasos iniciales**

1. **Descarga** Ubuntu Server: [ubuntu.com/download/server](https://ubuntu.com/download/server)
2. Crea una nueva máquina virtual con estas opciones:
   * Tipo: Linux
   * Versión: Ubuntu (64-bit)
   * RAM: 2048 MB
   * Disco: 20 GB (dinámico)
3. Monta el ISO y arranca la VM
4. Sigue el instalador (modo CLI):
   * Idioma
   * Teclado
   * Conexión a Internet
   * Disco (usar todo)
   * Crear usuario y contraseña
   * Esperar instalación

✅ Cuando termines, tendrás una consola de Ubuntu lista para comenzar a instalar roles de servidor.

**🧠 PREGUNTAS DE RETROALIMENTACIÓN**

1. ¿Cuál es una ventaja de usar Linux como sistema operativo de servidor?
2. ¿Qué sistema operativo es ideal para una empresa con servicios de Active Directory?
3. ¿Qué es un hipervisor bare-metal y cuál es un ejemplo?
4. ¿Qué beneficios ofrece usar una distribución con soporte LTS?
5. ¿Por qué es importante reducir los servicios instalados en un servidor?

## **CLASE 5: Arquitectura Cliente-Servidor**

**🎯 Objetivo de la clase:**

Comprender qué es la arquitectura cliente-servidor, cómo funciona, sus ventajas, desventajas y su aplicación en entornos reales.

**🔎 ¿Qué es la arquitectura cliente-servidor?**

Es un **modelo de comunicación en red** donde dos roles principales se definen:

* **Cliente:** el que solicita servicios (como un navegador web).
* **Servidor:** el que responde y proporciona los servicios solicitados (como un servidor web).

**📊 Características principales**

| **Cliente** | **Servidor** |
| --- | --- |
| Inicia la conexión | Espera conexiones |
| Hace peticiones (HTTP, DNS, FTP, etc.) | Responde con datos o acciones |
| Puede ser una aplicación de escritorio, móvil o navegador | Puede ser físico, virtual o en la nube |

**📦 Ejemplos comunes**

* **Navegador web** (cliente) 🖥️ → **Servidor web (Apache/NGINX)** 🌐
* **Cliente de correo** (Thunderbird) ✉️ → **Servidor de correo (Postfix, Exchange)** 📬
* **Cliente FTP** (FileZilla) → **Servidor FTP**

**🧰 Flujo básico de comunicación**

1. El cliente envía una **petición** (por ejemplo, GET /index.html).
2. El servidor **recibe** la solicitud.
3. El servidor **procesa** la solicitud (consulta a la base de datos, busca archivos...).
4. El servidor **envía la respuesta** al cliente.
5. El cliente **muestra** el resultado.

**✅ Ventajas**

* **Centralización:** la información y los servicios están en un lugar controlado.
* **Escalabilidad:** puedes ampliar el servidor si hay más demanda.
* **Seguridad:** los accesos están controlados.

**⚠️ Desventajas**

* **Dependencia del servidor:** si cae, todos los clientes pierden acceso.
* **Sobrecarga posible:** si hay demasiadas solicitudes.

**🛠️ PROYECTO PRÁCTICO (Parte 2)**

**Objetivo:** Instalar y probar un servidor HTTP para aplicar la arquitectura cliente-servidor.

**1. Instalar Apache en tu Debian Server**

bash

CopiarEditar

sudo apt update

sudo apt install apache2 -y

**2. Verifica que esté corriendo**

bash

CopiarEditar

sudo systemctl status apache2

Debe decir “active (running)”.

**3. Desde otro equipo en la red (cliente), abre un navegador y escribe la IP de tu servidor:**

cpp

CopiarEditar

http://<IP\_DEL\_SERVIDOR>

Deberías ver la página por defecto de Apache.

**🔁 Retroalimentación – 5 Preguntas**

1. ¿Qué papel juega el cliente en la arquitectura cliente-servidor?
2. ¿Por qué es mejor tener los servicios centralizados en un servidor?
3. ¿Qué ocurriría si el servidor deja de funcionar en esta arquitectura?
4. ¿Cuál es la diferencia entre un servidor web y un servidor de base de datos?
5. ¿Qué servicio instalaste en tu proyecto práctico y cómo lo comprobaste?

## **CLASE 6: Protocolos de Red Esenciales**

**🎯 Objetivo:**

Conocer y entender los protocolos de red más utilizados en entornos de servidores y cómo aplicarlos en un entorno práctico.

**🔗 ¿Qué es un protocolo de red?**

Un **protocolo de red** es un **conjunto de reglas** que determinan cómo se comunican los dispositivos en una red. Permiten que clientes y servidores se entiendan, transmitan datos, establezcan conexiones y garanticen la seguridad.

**📡 Protocolos esenciales en servidores**

| **Protocolo** | **Función Principal** | **Puerto Típico** |
| --- | --- | --- |
| **HTTP / HTTPS** | Transmisión de páginas web | 80 / 443 |
| **FTP / SFTP** | Transferencia de archivos | 21 / 22 |
| **SSH** | Acceso remoto seguro | 22 |
| **DNS** | Traduce nombres de dominio a IP | 53 |
| **DHCP** | Asignación automática de IPs | 67 / 68 |
| **SMTP, IMAP, POP3** | Envío y recepción de correos electrónicos | 25, 143, 110 |
| **SNMP** | Monitorización de dispositivos de red | 161 |
| **NTP** | Sincronización de reloj de servidores | 123 |
| **LDAP** | Servicios de autenticación | 389 |
| **ICMP** | Diagnóstico de red (ej. ping) | No tiene puerto, usa IP directamente |

**🌐 ¿Por qué son importantes?**

* Garantizan que los servicios funcionen de forma **interoperable** (Linux, Windows, móviles, etc.).
* Permiten **automatizar tareas** (como asignar IPs con DHCP).
* Ayudan a **diagnosticar problemas** de red y rendimiento.
* Son la base de **seguridad, administración y conectividad** en cualquier entorno de servidor.

**🛠️ PROYECTO PRÁCTICO – Parte 3: Probar protocolos clave**

**1. Probar conectividad con ICMP**

Desde otro equipo, prueba conectividad al servidor:

**bash**

ping <IP\_DEL\_SERVIDOR>

**2. Habilitar y probar SSH**

Ya instalado en Debian, habilítalo y verifica su estado:

**bash**

sudo systemctl enable ssh

sudo systemctl start ssh

sudo systemctl status ssh

Desde otro equipo, accede:

**bash**

ssh usuario@IP\_DEL\_SERVIDOR

**3. Instalar y probar FTP**

**bash**

sudo apt install vsftpd

sudo systemctl enable vsftpd

sudo systemctl start vsftpd

Conéctate desde FileZilla o desde terminal:

**bash**

ftp IP\_DEL\_SERVIDOR

(Recuerda que para seguridad real, se recomienda usar SFTP o FTPS).

**🔁 RETROALIMENTACIÓN – 5 Preguntas**

1. ¿Qué función cumple el protocolo DNS en una red?
2. ¿Cuál es la diferencia entre HTTP y HTTPS?
3. ¿Por qué SSH es más seguro que Telnet?
4. ¿Qué protocolo permite asignar IPs automáticamente a los clientes?
5. ¿Qué utilidad tiene ICMP en administración de servidores?

**CLASE 7: MODELOS OSI Y TCP/IP**

**🎯 Objetivo:**

Comprender las capas del modelo OSI y TCP/IP, y cómo se relacionan con los protocolos y servicios que ya has visto.

**📦 Modelo OSI (Open Systems Interconnection)**

Es un modelo conceptual de **7 capas** que define cómo se comunican los sistemas en red. Cada capa tiene una función específica y se comunica con la superior e inferior.

| **Capa** | **Nombre** | **Función** |
| --- | --- | --- |
| 7 | Aplicación | Interfaz con el usuario o aplicación (HTTP, FTP, DNS) |
| 6 | Presentación | Formato y cifrado de datos (SSL, TLS, JPEG, MPEG) |
| 5 | Sesión | Establecimiento y cierre de sesiones (NetBIOS, RPC) |
| 4 | Transporte | Control de flujo y confiabilidad (TCP, UDP) |
| 3 | Red | Direccionamiento lógico y enrutamiento (IP, ICMP) |
| 2 | Enlace de datos | Direcciones físicas, control de errores (Ethernet, PPP) |
| 1 | Física | Transmisión de bits en medios físicos (cables, señales) |

📌 **Ejemplo real:** Cuando abres una web, estás usando:  
HTTP (capa 7) → TCP (4) → IP (3) → Ethernet (2) → señal eléctrica (1).

**🌐 Modelo TCP/IP (más usado en la práctica)**

Es el modelo implementado en Internet. Tiene solo 4 capas, más prácticas:

| **Capa** | **Equiv. OSI** | **Ejemplos** |
| --- | --- | --- |
| 4. Aplicación | 5–7 | HTTP, FTP, SMTP |
| 3. Transporte | 4 | TCP, UDP |
| 2. Internet | 3 | IP, ICMP |
| 1. Acceso a red | 1–2 | Ethernet, Wi-Fi |

**🧠 Comparación OSI vs TCP/IP**

* OSI es **teórico y descriptivo** (ideal para aprender).
* TCP/IP es **práctico y funcional**, implementado en todos los sistemas.

**🧰 Aplicación práctica en servidores**

* Cuando configuras un **servidor web (Apache)**: trabajas en la **capa de aplicación**.
* Si aseguras SSH con TLS: estás en la **capa de presentación y transporte**.
* Cuando configuras IP, DNS o NICs: estás en las **capas de red y enlace**.

**🛠️ PROYECTO PRÁCTICO – Parte 4**

**🎯 Objetivo: Observar cómo fluye el tráfico por capas usando tcpdump o wireshark (sniffer de red)**

1. En tu Debian Server, instala tcpdump:

bash

CopiarEditar

sudo apt install tcpdump -y

1. Captura tráfico HTTP en tiempo real (en otra terminal accede desde un cliente):

bash

CopiarEditar

sudo tcpdump -i eth0 port 80

1. Verás tráfico estructurado por IP, puertos (capa 3 y 4), y si activas -A puedes ver datos HTTP (capa 7).

📌 Esto **te muestra directamente las capas del modelo TCP/IP** en acción. Verás el origen, destino, protocolo y contenido de la comunicación.

**🔁 RETROALIMENTACIÓN – 5 Preguntas**

1. ¿Cuántas capas tiene el modelo OSI y cuál es su propósito?
2. ¿En qué capa del modelo OSI se encuentra el protocolo IP?
3. ¿Qué diferencia principal hay entre OSI y TCP/IP?
4. ¿En qué capa se encuentra el protocolo HTTP?
5. ¿Por qué es importante entender estas capas cuando trabajamos con servidores?