INSTITUTO POLITECNICO NACIONAL

UNIDAD INTERDICIPLINARIA EN INGENIERIAS Y TECNOLOGIAS AVANZADAS.

ADMINISTRACIÓN DE SISTEMAS OPERATIVOS

PROYECTO FINAL

HÉCTOR MENDOZA CORTEZ

GRUPO: 1TM1

COSIJOEZA VICTORIA JESSICA

BOLETA: 2022640049

- **PROTOCOLO DE SSH.**

SSH™ (o Secure SHell) es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente. A diferencia de otros protocolos de comunicación remota tales como FTP o Telnet, SSH encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas.

SSH está diseñado para reemplazar los métodos más viejos y menos seguros para registrarse remotamente en otro sistema a través de la shell de comando, tales como telnet o rsh. Un programa relacionado, el scp, reemplaza otros programas diseñados para copiar archivos entre hosts como rcp. Ya que estas aplicaciones antiguas no encriptan contraseñas entre el cliente y el servidor, evite usarlas mientras le sea posible. El uso de métodos seguros para registrarse remotamente a otros sistemas reduce los riesgos de seguridad tanto para el sistema cliente como para el sistema remoto.

TIPOS DE PROTECCIÓN.

- Después de la conexión inicial, el cliente puede verificar que se está conectando al mismo servidor al que se conectó anteriormente.
- El cliente transmite su información de autenticación al servidor usando una encriptación robusta de 128 bits.
- Todos los datos enviados y recibidos durante la sesión se transfieren por medio de encriptación de 128 bits, lo cual los hacen extremamente difícil de descifrar y leer.
- El cliente tiene la posibilidad de reenviar aplicaciones X11 [1] desde el servidor. Esta técnica, llamada reenvío por X11, proporciona un medio seguro para usar aplicaciones gráficas sobre una red.

Ya que el protocolo SSH encripta todo lo que envía y recibe, se puede usar para asegurar protocolos inseguros. El servidor SSH puede convertirse en un conducto para convertir en seguros los protocolos inseguros mediante el uso de una técnica llamada reenvío por puerto, como por ejemplo POP, incrementando la seguridad del sistema en general y de los datos.

Red Hat Enterprise Linux contiene el paquete general de OpenSSH (openssh) así como también los paquetes del servidor OpenSSH (openssh-server) y del cliente (openssh-clients). Consulte el capítulo titulado **OpenSSH** en el **Manual de administración del sistema de Red Hat Enterprise Linux** para obtener instrucciones sobre la instalación y el desarrollo de OpenSSH. Observe que los paquetes OpenSSH requieren el paquete OpenSSL (openssl). OpenSSL instala varias bibliotecas criptográficas importantes, permitiendo que OpenSSH pueda proporcionar comunicaciones encriptadas.

USO.
Los usuario nefarios tienen a su disposición una variedad de herramientas que les permiten interceptar y redirigir el tráfico de la red para ganar acceso al sistema. En términos generales, estas amenazas se pueden catalogar del siguiente modo:

- Intercepción de la comunicación entre dos sistemas — En este escenario, existe un tercero en algún lugar de la red entre entidades en comunicación que hace una copia de la información que pasa entre ellas. La parte interceptora puede interceptar y conservar la información, o puede modificar la información y luego enviarla al recipiente al cual estaba destinada.
- Este ataque se puede montar a través del uso de un paquete sniffer — una utilidad de red muy común.
- Personificación de un determinado host — Con esta estrategia, un sistema interceptor finge ser el recipiente a quien está destinado un mensaje. Si funciona la estrategia, el sistema del usuario no se da cuenta del engaño y continúa la comunicación con el host incorrecto.
- Esto se produce con técnicas como el envenenamiento del DNS [2] o spoofing de IP (engaño de direcciones IP) [3].

Ambas técnicas interceptan información potencialmente confidencial y si esta intercepción se realiza con propósitos hostiles, el resultado puede ser catastrófico.
Si se utiliza SSH para inicios de sesión de shell remota y para copiar archivos, se pueden disminuir estas amenazas a la seguridad notablemente. Esto es porque el cliente SSH y el servidor usan firmas digitales para verificar su identidad. Adicionalmente, toda la comunicación entre los sistemas cliente y servidor es encriptada. No servirán de nada los intentos de falsificar la identidad de cualquiera de los dos lados de la comunicación ya que cada paquete está cifrado por medio de una llave conocida sólo por el sistema local y el remoto.

- C**ONFIGURACIÓN EN LINUX DEL SERVIDOR SSH.**

SSH se ha convertido en el estándar de conexión a cualquier servidor UNIX, precisamente, para que los hackers no puedan espiar los datos que enviamos a los servidores a los que nos conectamos, como nuestro usuario y contraseña, por ejemplo.
Antiguamente, nos conectábamos a los servidores mediante telnet o relogin, por ejemplo, pero estos protocolos no estaban cifrados y, por lo tanto, eran vulnerables a un ataque.
En Linux, el demonio SSH se arranca con el boot del sistema y está configurado como servicio.
Ejemplo en Centos 7:
ps -ef |grep ssh |grep -v grep
root 965 1 0 Sep23 ? 00:00:17 /usr/sbin/sshd -D
root 30020 965 0 10:18 ? 00:00:00 sshd: centos [priv]
centos 30022 30020 0 10:18 ? 00:00:00 sshd: centos@pts/0
systemctl list-unit-files |grep -i ssh
sshd-keygen.service static
sshd.service enabled
sshd@.service static
sshd.socket disabled
#
Si el servicio SSH está arrancado y disponemos de un usuario y contraseña, podremos utilizar un cliente SSH para conectarnos al servidor o utilizar el comando SSH.
ssh 192.168.1.2
root@testserver's password:

- **Configuración mínima para garantizar la seguridad de la conexión.**

Llaves SSH

Las llaves SSH son un par de llaves criptográficas que son utilizadas para autenticar usuarios en un servidor SSH y se utilizan como una alternativa al inicio de sesión por medio de contraseña. Una sola llave SSH consta de dos archivos, la llave privada y la llave pública. La llave pública puede ser compartida a través de internet en múltiples servidores. La llave privada debe permanecer secreta y asegurada por el usuario.

Para configurar la autenticación SSH, primero deberás contar con las llaves públicas y privadas. En el servidor donde deseas autenticarte deberás colocar la llave pública mientras que la llave privada la debes mantener en tu equipo local. El cliente SSH utilizará la llave privada y la comparará con la llave pública y de esta forma determinará si la autenticación es correcta.

Luego de esto se deberá configurar el servidor para utilizar SSH como método de autenticación y una vez configurado tenemos que deshabilitar el acceso por contraseña. Hacer esto incrementará la seguridad de tu servidor evitando que usuarios no autorizados se conecten.

En el curso de administración de servidores para PHP y Laravel explicamos cómo configurar el servicio de SSH.

Firewalls

Un firewall es usualmente un software (aunque puede ser también una pieza de hardware) que controla los servicios que están expuestos en la red. Su función principal es bloquear o restringir el acceso a todos los puertos excepto aquellos que tienen que estar disponibles de manera pública.

En un servidor, típicamente se tienen algunos servicios ejecutándose de manera predeterminada. Estos pueden organizarse en los siguientes grupos:

- Servicios públicos, que pueden ser accedidos por cualquiera en Internet, usualmente de manera anónima. Un buen ejemplo de esto es un servidor Web que permite el acceso a los sitios que aloja.
- Servicios privados, que solamente pueden ser accedidos por un selecto grupo de cuentas autorizadas o desde ciertas ubicaciones. Un ejemplo de esto puede ser el panel de control de una base de datos.
- Servicios internos, que son accesibles solamente desde el mismo servidor y no están expuestos al mundo exterior. Por ejemplo, una base de datos que solamente acepte conexiones locales.

Los firewalls pueden asegurar que el acceso a su software esté restringido de acuerdo a las categorias mencionadas anteriormente. Los servicios públicos pueden quedarse abiertos y disponibles para todo el mundo, mientras que los servicios privados pueden ser restringidos basándose en diferentes criterios según la organización que los esté utilizando. Los servicios internos siempre son completamente inaccesibles desde el mundo exterior.

En el curso de administración de servidores para PHP y Laravel explicamos cómo configurar reglas para el Firewall con iptables.

VPNs y Redes Privadas

Las redes privadas son redes que solamente están disponibles para ciertos usuarios o servidores. Por ejemplo, una organización puede tener servidores en distintas regiones de todo el mundo y por medio de una red privada estos servidores pueden comunicarse entre sí sin importar su ubicación real.

Un VPN o Red Privada Virtual (Virtual Private Network) es una manera de crear conexiones seguras entre computadoras remotas y presentan la conexión como si se estuviera trabajando en una red local privada. Esto brinda una manera de configurar sus servicios como si estuvieran en una red privada y conectar servidores remotos a través de una conexión segura.

Infraestructura de Llave Pública y Cifrado SSL/TLS

La infraestructura de llave pública o PKI por sus siglas (Public Key Infrastructure) se refiere a un sistema que está diseñado para crear, administrar y validar certificados para identificar individuos y encriptar comunicaciones. Los certificados SSL o TLS pueden ser utilizados para autenticar diferentes entidades entre sí. Después de autenticarse, también pueden ser utilizados para encriptar comunicaciones.

Auditoría de servicios

Hasta ahora, hemos mencionado algunas de las tecnologías que podemos utilizar para implementar y mejorar la seguridad. Sin embargo, una de las principales funciones de la seguridad es analizar los sistemas, identificando los puntos vulnerables a ataques y asegurando los componentes lo mejor que se pueda.

Auditar los servicios es una manera de descubrir qué servicios están en ejecución, qué puertos se están utilizando para comunicación y qué protocolos son aceptados. Esta información puede ayudarte a establecer una correcta configuración de tu firewall.

Auditoría de Archivos y Sistemas de Detección de Intrusos

La auditoría de archivos es el proceso de comparar el sistema actual contra un registro de archivos característicos de tu sistema cuando está en un estado saludable. Esto es utilizado para detectar cambios en el sistema que puedan no haber sido autorizados.

Un Sistema de Detección de Intrusos es un software que monitorea un sistema o una red para detectar actividad no autorizada. Varios sistemas de detección de intrusos utilizan una implementación de la auditoría de archivos para comprobar si un sistema ha sido modificado.

Entornos de Ejecución Aislados

Los Entornos de Ejecución Aislados se refieren a cualquier método en el que componentes individuales se ejecutan en su propio espacio dedicado.

Esto consta de separar los componentes de tu aplicación en sus propios servidores, o en dado caso, configurar los servicios para operar en entornos chroot o en contenedores. El nivel de aislamiento depende en gran parte en los requerimientos de tu aplicación y las capacidades de tu infraestructura.

Concluimos que debemos tomar en cuenta estas 7 medidas de seguridad para poder estar confiados de que nuestras aplicaciones van a estar protegidas contra atacantes, pero siempre es recomendable estar al tanto de las amenazas que surgen así como de los parches de seguridad para poder realizar todas las actualizaciones de seguridad en nuestros servidores.

- **Investigar cual es la diferencia del comando adduser a useradd**

Un sistema operativo se utiliza para dar instrucciones al hardware. Linux es un sistema operativo. Es un clon de UNIX. La principal ventaja de Linux es que los programadores pueden construir sus propios sistemas operativos utilizando el Kernel. Algunas distribuciones de Linux ampliamente utilizadas son Ubuntu, Fedora y Debian. Las tareas más frecuentes de la computadora son buscar, crear, mover y eliminar archivos. Hay dos métodos para manejar archivos de manera eficiente. Esto es mediante el uso de la interfaz de línea de comandos (CLI) o mediante la interfaz gráfica de usuario (GUI). Usar CLI es mejor en Linux porque es flexible y rápido. Los comandos se dan utilizando la CLI y Linux contiene un terminal para dar comandos. Hay una gran cantidad de comandos. Los comandos, adduser y useradd son para la administración de usuarios. los diferencia clave entre adduser y useradd es que adduser se usa para agregar usuarios con la configuración de la carpeta de inicio de la cuenta y otras configuraciones, mientras que useradd es un comando de utilidad de bajo nivel para agregar usuarios.

Adduser.
Los datos pueden ser cambiados o robados. Por lo tanto, es vital mantener los datos seguros. La seguridad es la principal preocupación en Linux. Es un sistema operativo multiusuario. Así que hay niveles de autorización en Linux. Cada archivo en Linux o Unix tiene un usuario. Hay tres tipos de usuarios en Linux. Son un usuario, grupo y otros. 'Usuario' es el propietario del archivo. Por defecto, el usuario que crea el archivo se convierte en el usuario. 'Grupo' puede contener múltiples usuarios. Todos los usuarios del grupo tienen los mismos permisos de archivo. Es posible agregar muchos usuarios al grupo y asignar permisos de grupo. 'Otro' no crea el archivo, pero tienen acceso al archivo.
Useradd.
El comando useradd también se usa para agregar usuarios. Viene con unas banderas. Algunos de ellos son los siguientes.

-D Valores predeterminados
-m crea un directorio de inicio
-s define el shell para el usuario
-e Fecha en que se deshabilitará la cuenta de usuario
-b Directorio base para el directorio de inicio del usuario.
-u UID
-g número de grupo inicial
-G Grupos adicionales por nombre
-c comentario
Diferencia entre Adduser y Useradd.
Adduser es el comando para agregar usuarios al sistema de acuerdo con las opciones de la línea de comandos y la información de configuración en /etc/adduser.conf.
Useradd es una utilidad de bajo nivel para agregar usuarios.

Caracteristicas

El comando adduser crea el usuario y configura las carpetas de inicio de la cuenta y otras configuraciones.
El comando useradd simplemente crea el usuario.

Creación de directorios

El comando adduser crea un directorio de usuario en la página de inicio (/ home / user) automáticamente.
El comando useradd no crea un directorio de usuario en el hogar, si no se especifica con -m.

Complejidad de la sintaxis

La sintaxis del comando para adduser no es complicada como en useradd.
El comando useradd tiene cierta complejidad..

- **Variantes de sintaxis del comando adduser.**

El comando adduser es uno de los más usado en los sistemas operativos UNIX. Se utiliza para agregar los usuarios.Adduser añade usuarios al sistema de acuerdo a las opciones de la línea de comando y a la configuración en /etc/adduser.conf. Ofrecen una interfaz más sencilla para programas de bajo nivel como useradd, groupadd y usermod, seleccionando valores para el identificador de usuario (UID) e identificador de grupo de usuarios (GID) conformes con las normas de Debian. También crean un directorio personal (/home/USUARIO) con la configuración predeterminada, ejecutan un script personalizado y otras funcionalidades. adduser puede ejecutarse de varias maneras distintas:
Sintasis
adduser [opciones] [--home DIRECTORIO] [--shell CONSOLA] [--no-create-home] [--uid ID] [--firstuid ID] [--lastuid ID] [--ingroup GRUPO | --gid ID] [--disabled-password] [--disabled-login] [--gecos GECOS] [--add_extra_groups] USUARIO
adduser --system [opciones] [--home DIRECTORIO] [--shell CONSOLA] [--no-create-home] [--uid ID] [--group | --ingroup GRUPO | --gid ID] [--disabled-password] [--disabled-login] [--gecos GECOS] USUARIO
adduser [opciones] usuario grupo
Opciones más comunes
[--quiet] [--debug] [--force-badname] [--help|-h] [--version] [--conf FICHERO]
Añadir un usuario normal
Si se invoca con un argumento que no es ninguna opción y sin la opción--system o --group, adduser añadirá un usuario normal. Adduser elegirá el primer UID disponible dentro del rango especificado para usuarios normales en el fichero de configuración. Puede elegir uno manualmente usando la opción

--uid.
Puede modificar el rango especificado en el fichero de configuración usando las opciones

--firstuid y --lastuid.
Por omisión, cada usuario en Debian GNU/Linux tiene su grupo correspondiente con el mismo nombre. Los grupos de usuarios permiten mantener directorios con permisos de escritura para un grupo de usuarios de forma sencilla añadiendo los usuarios apropiados al nuevo grupo, habilitando después el bit set-group-ID en el directorio, y comprobando que todos los usuarios tengan un umask de 002. Si esta opción se deshabilita definiendo USERGROUPS como no, todos los GID de usuario corresponder a USERS_GID. Los grupos primarios de usuario también se pueden deshabilitar usando las opciones de la línea de órdenes --gid o --ingroup para establecer el grupo por id o por nombre, respectivamente. Así mismo, se pueden añadir usuarios a uno o más grupos definidos en adduser.conf, bien definiendo ADD_EXTRA_GROUPS como en adduser.confo introduciendo --add_extra_groups en la línea de órdenes. Adduser creará los directorios personales de acuerdo con DHOME, GROUPHOMES, y LETTERHOMES. El directorio personal se puede especificar mediante la opción de línea de ódenes --home, y la consola mediante la opción --shell. El bit set-group-ID del directorio personal está habilitado si USERGROUPS es yes, de forma que cualquier fichero creado en el directorio personal del usuario tendrá el grupo correcto. Adduser copiará los ficheros desde SKEL en el directorio personal y preguntará por la información del campo gecos y por la clave.

El campo gecos también se puede definir con la opción --gecos. Con la opción--disabled-login, la cuenta se creará pero estará deshabilitada hasta que se proporcione una clave. La opción --disabled-password no establecerá la clave, pero todavía será posible trabajar con la cuenta, por ejemplo mediante claves SSH RSA. Si existe el fichero /usr/local/sbin/adduser.local, se ejecutará después de que la cuenta de usuario esté lista, posibilitando realizar ajustes locales. Los argumentos que se pasan a adduser.local son:

nombre-usuario UID GID directorio-personal
La variable de entorno VERBOSE se define de acuerdo a la siguiente regla:

0 si se define --quiet
1 si no se definen ni --quiet ni --debug
2 si se define --debug
Añadir un usuario del sistema
Si se invoca con un argumento que no es ninguna opción y la opción --system, adduser añadirá un usuario del sistema. Si ya existe un usuario con el mismo nombre en el rango del sistema de UID (o si se especifica el UID y ya existe un usuario con ese UID), adduser abandonará con un aviso. Puede suprimir este aviso añadiendo --quiet. adduser elegirá el primer UID disponible en el rango especificado en el fichero de configuración para usuarios del sistema (FIRST_SYSTEM_UID y LAST_SYSTEM_UID). Si desea un UID específico, lo puede especificar con la opción --uid. Por omisión, los usuarios del sistema se añaden al grupo nogroup. Para añadir el nuevo usuario del sistema a un grupo existente, use las opciones --gid o --ingroup. Para añadir el nuevo usuario del sistema a un grupo con su mismo ID, use la opción --group. El directorio personal se crea con las mismas normas que para los usuarios normales. Los nuevos usuarios del sistema tendrán como consola /bin/false (a menos que se modifique con la opción --shell), y tienen la clave deshabilitada. Los ficheros de configuración esqueleto no se copian.

Opciones
--conf FICHERO
Usa FICHERO en vez de /etc/adduser.conf.

--disabled-login
No ejecuta passwd para establecer la clave. El usuario no podrá usar la cuenta hasta que se establezca una clave.

--disabled-password
Como --disabled-login, pero todavía es posible usar la cuenta, por ejemplo mediante claves SSH RSA, pero no usando autentificación de claves.

--force-badname
Por omisión, el nombre de usuario se compara con la expresión regular configurable NAME_REGEX definida en el fichero de configuración. Esta opción fuerza a adduser a ser más indulgente en sus comprobaciones de la validez de un nombre.

--gecos GECOS
Especifica el nuevo campo gecos para la entrada generada. adduser no solicitará esta información si se proporciona esta opción.

--help
Muestra unas instrucciones breves.
Crea un usuario del sistema o grupo.

--home DIRECTORIO
Usa DIRECTORIO para el directorio personal, en vez del predeterminado especificado en el fichero de configuración. Si el directorio no existe, se crea y se copian los ficheros de esqueleto.

--shell CONSOLA
Usar CONSOLA como la consola de entrada del usuario, en vez del predeterminado especificado en el fichero de configuración.

--no-create-home
No crea el directorio personal, incluso si no existe.

--quiet
Elimina los mensajes informativos, sólo muestra avisos y errores.

--debug
Muestra más información, útil si desea encontrar el origen de un problema con adduser.

--system
Crea un usuario del sistema o grupo.

--uid ID
Fuerza el nuevo identificador de usuario al número dado. adduser fallará si el UID ya está en uso.

--firstuid ID
Modifica el primer UID del rango del cual se eligen los UID (anula el valor de FIRST_UID definido en el fichero de configuración).

--lastuid ID
Modifica el último UID del rango del cual se eligen los UID (LAST_UID).

--add_extra_groups
Añade un nuevo usuario a los grupos adicionales definidos en el fichero de configuración.

--version
Muestra la versión e información acerca del copyright.

Valores de salida
0 El usuario definido ya existe. Puede tener dos causas: El usuario se ha creado mediante adduser, o el usuario ya existía en el sistema antes de invocar adduser. Si adduser devuelve 0, invocar adduser por segunda vez con los mismos parámetros también devuelve 0.
1 Ha fallado la creación de un usuario porque ya existía con un UID/GID diferente del especificado. El nombre de usuario ha sido rechazado por no coincidir con la expresión regular configurada. Una señal ha cancelado la ejecución de adduser.
O por otras razones no documentadas que se muestran en el interprete de ordenes. Puede entonces considerar eliminar --quiet para que adduser sea más informativo.

Fichero de configuración
/etc/adduser.conf
Es el fichero de configuración predeterminado de adduser .

- **Y si todas las distribuciones de Linux cuentas con el comando adduser.**

Ejemplos prácticos del comando useradd linux
Los siguientes ejemplos de uso de userdd te servirán en cualquier distribución linux.
Crear cuenta de usuario
Esta es la forma mas sencilla de crear una cuenta.
En caso de no estar desde la cuenta root, debemos ejecutar el comando con sudo.
Automáticamente se asignara un uid y gid (user id y group id) a la nueva cuenta ademas de una carpeta personal en /home.
sudo useradd pepe
man useradd, la mas completa
Como siempre, tendrás la información mas completa y actualizada en Linux haciendo uso de las paginas man.
man useradd
Pidiendo ayuda en consola para useradd linux
La otra forma algo mas rápida y sencilla es utilizar la ayuda del comando.
useradd --help
Ejemplos prácticos del comando useradd linux
Los siguientes ejemplos de uso de userdd te servirán en cualquier distribución linux.
Crear cuenta de usuario
Esta es la forma mas sencilla de crear una cuenta.
En caso de no estar desde la cuenta root, debemos ejecutar el comando con sudo.
Automáticamente se asignara un uid y gid (user id y group id) a la nueva cuenta ademas de una carpeta personal en /home.
sudo useradd pepe.

- Investigación sobre el protocolo web, servidor apache2, iis (este último para Windows).

Apache HTTP Server es un software de servidor web gratuito y de código abierto para plataformas Unix con el cual se ejecutan el 46% de los sitios web de todo el mundo. Es mantenido y desarrollado por la Apache Software Foundation.
Le permite a los propietarios de sitios web servir contenido en la web y es uno de los servidores más antiguos y confiables, con la primera versión lanzada hace más de 20 años, en 1995.
Cuando alguien quiere visitar un sitio web, ingresa un nombre de dominio en la barra de direcciones de su navegador. Luego, el servidor envía los archivos solicitados actuando como un repartidor virtual.
Aquí en Hostinger, nuestra infraestructura de hosting web utiliza LiteSpeed, que es otro software popular de servidor web.
¿Qué es un servidor web?
Un servidor web es un programa de tipo informático que se encarga de procesar una aplicación del lado del servidor, cada una de las cuales puede acceder a archivos almacenados en un servidor físico y usarlos para diferentes propósitos, mediante conexiones bidireccionales o unidireccionales con la máquina del cliente, tras lo cual se genera una respuesta del lado del cliente.
El trabajo de un servidor es servir sitios web en Internet. Para lograr ese objetivo, actúa como un intermediario entre el servidor y las máquinas de los clientes. Extrae el contenido del servidor en cada solicitud de usuario y lo envía a la web.
El mayor desafío de un servidor es servir a muchos usuarios diferentes de la web al mismo tiempo, cada uno de los cuales solicita diferentes páginas. Los servidores web procesan archivos escritos en diferentes lenguajes de programación como PHP, Python, Java y otros.
Los convierten en archivos HTML estáticos y le entregan estos archivos al navegador de los usuarios de la web. Cuando escuches la palabra servidor web, piensa que es la herramienta responsable de la comunicación adecuada entre el servidor y el cliente.

¿Cómo funciona Apache?

Aunque llamamos a Apache un servidor web, no es un servidor físico, sino un software que se ejecuta en un servidor. Su trabajo es establecer una conexión entre un servidor y los navegadores de los visitantes del sitio web (Firefox, Google Chrome, Safari, etc.) mientras envían archivos entre ellos (estructura cliente-servidor). Apache es un software multiplataforma, por lo cual funciona tanto en servidores Unix como en Windows.

Cuando un visitante quiere cargar una página de tu sitio web, por ejemplo la página de inicio o tu página «Acerca de nosotros», su navegador le envía una solicitud a tu servidor y Apache le devuelve una respuesta con todos los archivos solicitados (texto, imágenes, etc.) El servidor y el cliente se comunican a través del protocolo HTTP y Apache es responsable de garantizar una comunicación fluida y segura entre las dos máquinas.

Apache es altamente personalizable, ya que tiene una estructura basada en módulos. Los módulos le permiten a los administradores del servidor activar y desactivar funcionalidades adicionales.

Apache tiene **módulos** de seguridad, almacenamiento en caché, reescritura de URL, autenticación de contraseña y más. También puedes ajustar tus propias configuraciones del servidor a través de un archivo llamado **.htaccess**, que es un archivo de configuración de Apache compatible con todos los planes de **Hostinger**.

Apache vs otros servidores web

Además de Apache, hay muchos otros servidores web. Cada aplicación de servidor ha sido creada para un propósito diferente. Si bien Apache es el más utilizado, tiene bastantes alternativas y rivales.

Apache vs NGINX

Nginx, pronunciado como Engine-X, es una aplicación de servidor web más reciente lanzada por primera vez en el 2004. A la fecha, ha ganado una gran popularidad entre los propietarios de sitios web. Nginx fue creado para resolver el denominado problema c10k, lo que significa que un servidor que utiliza subprocesos para manejar las solicitudes de los usuarios no puede administrar más de 10,000 conexiones al mismo tiempo.

1. Dado que Apache utiliza la estructura basada en subprocesos, los propietarios de sitios web con mucho tráfico pueden tener problemas de rendimiento. Nginx es uno de los servidores web que abordan el problema c10k y probablemente el más exitoso.
2. Nginx tiene una arquitectura guiada por los eventos que no crea un nuevo proceso para cada solicitud. En cambio, maneja todas las solicitudes entrantes en un solo proceso. Este proceso maestro gestiona varios procesos de trabajo que realizan el procesamiento real de las solicitudes. El modelo basado en eventos de Nginx distribuye las solicitudes de los usuarios entre los procesos de trabajo de una manera eficiente, por lo tanto, conduce a una escalabilidad mucho mejor.
3. Si necesitas administrar un sitio web de alto tráfico, Nginx es una excelente opción, ya que puede hacerlo mediante el uso de recursos mínimos. No puede ser una coincidencia que sea utilizado por muchos sitios web de alta visibilidad como Netflix, Hulu, Pinterest y Airbnb.
4. Sin embargo, para los sitios web pequeños y medianos, Apache tiene varias ventajas sobre Nginx, como su fácil configuración, muchos módulos y un entorno amigable para principiantes.

¿Cómo funciona Apache?

Aunque llamamos a Apache un servidor web, no es un servidor físico, sino un software que se ejecuta en un servidor. Su trabajo es establecer una conexión entre un servidor y los navegadores de los visitantes del sitio web (Firefox, Google Chrome, Safari, etc.) mientras envían archivos entre ellos (estructura cliente-servidor). Apache es un software multiplataforma, por lo cual funciona tanto en servidores Unix como en Windows.

Cuando un visitante quiere cargar una página de tu sitio web, por ejemplo la página de inicio o tu página «Acerca de nosotros», su navegador le envía una solicitud a tu servidor y Apache le devuelve una respuesta con todos los archivos solicitados (texto, imágenes, etc.) El servidor y el cliente se comunican a través del protocolo HTTP y Apache es responsable de garantizar una comunicación fluida y segura entre las dos máquinas.

Apache es altamente personalizable, ya que tiene una estructura basada en módulos. Los módulos le permiten a los administradores del servidor activar y desactivar funcionalidades adicionales.

Apache tiene **módulos** de seguridad, almacenamiento en caché, reescritura de URL, autenticación de contraseña y más. También puedes ajustar tus propias configuraciones del servidor a través de un archivo llamado **.htaccess**, que es un archivo de configuración de Apache compatible con todos los planes de **Hostinger**.

Apache vs otros servidores web

Además de Apache, hay muchos otros servidores web. Cada aplicación de servidor ha sido creada para un propósito diferente. Si bien Apache es el más utilizado, tiene bastantes alternativas y rivales.

Apache vs NGINX

Nginx, pronunciado como Engine-X, es una aplicación de servidor web más reciente lanzada por primera vez en el 2004. A la fecha, ha ganado una gran popularidad entre los propietarios de sitios web. Nginx fue creado para resolver el denominado problema c10k, lo que significa que un servidor que utiliza subprocesos para manejar las solicitudes de los usuarios no puede administrar más de 10,000 conexiones al mismo tiempo.

1. Dado que Apache utiliza la estructura basada en subprocesos, los propietarios de sitios web con mucho tráfico pueden tener problemas de rendimiento. Nginx es uno de los servidores web que abordan el problema c10k y probablemente el más exitoso.
2. Nginx tiene una arquitectura guiada por los eventos que no crea un nuevo proceso para cada solicitud. En cambio, maneja todas las solicitudes entrantes en un solo proceso. Este proceso maestro gestiona varios procesos de trabajo que realizan el procesamiento real de las solicitudes. El modelo basado en eventos de Nginx distribuye las solicitudes de los usuarios entre los procesos de trabajo de una manera eficiente, por lo tanto, conduce a una escalabilidad mucho mejor.
3. Si necesitas administrar un sitio web de alto tráfico, Nginx es una excelente opción, ya que puede hacerlo mediante el uso de recursos mínimos. No puede ser una coincidencia que sea utilizado por muchos sitios web de alta visibilidad como Netflix, Hulu, Pinterest y Airbnb.
4. Sin embargo, para los sitios web pequeños y medianos, Apache tiene varias ventajas sobre Nginx, como su fácil configuración, muchos módulos y un entorno amigable para principiantes.

- **Configuración en Linux del servidor apache.**

Installation Manager instala y configura HTTP Server de Apache como servidor web para Build Forge. El uso del HTTP Server de Apache proporcionado es el modo más rápido de configurar un servidor web para Build Forge.
Como alternativa a la configuración estándar, puede configurar un HTTP Server de Apache existente en lugar de uno instalado y configurado por Build Forge. Las instrucciones proporcionadas suponen que tiene experiencia instalando y configurando HTTP Server de Apache en el sistema operativo.
Para usar HTTP Server de Apache, modifique la instalación de la siguiente forma:

1. Modifique el archivo de configuración de Apache HTTP Server (httpd-vhosts.conf) para que señale a la aplicación Build Forge.
2. Instale PHP y configure los módulos PHP necesarios para HTTP Server de Apache, la base de datos de Build Forge y el cifrado de contraseña, si desea utilizar esta función de seguridad.
3. Configurar Apache para la base de datos.Instalar Build Forge mediante Installation Manager
4. En Installation Manager, en la página Configuración de aplicación y servidor web, seleccione Sí en el indicador Proporcionar su propio servidor web.
5. Software de requisito previo
6. Apache HTTP Server 2.2.4
7. PHP 5.2.4
8. Edite el archivo de configuración del servidor de Apache
9. Localice el archivo http-vhosts.conf de Apache en el directorio extras de la instalación del servidor
10. Edite el archivo http-vhosts.conf de Apache. Para añadir información sobre Build Forge a httpd-vhosts.conf, añada las siguientes líneas:
11. Modifique el valor de DocumentRoot para que señale la aplicación web de Build Forge. En este ejemplo, el directorio de instalación de Build Forge es /opt/buildforge.
12. Deje el puerto como 80 o cámbielo al puerto en el que se ejecute Apache HTTP Server localmente.
13. <VirtualHost *:80>

- Configuración mínima para garantizar la seguridad de la conexión web.

Llaves SSH
Las llaves SSH son un par de llaves criptográficas que pueden ser usadas para autenticarse en un servidor SSH; es un método alternativo al uso de contraseñas. La creación del par compuesto por llave pública y privada es llevada a cabo como un paso anterior a la autenticación. La llave privada la conserva el usuario de manera secreta y segura, mientras que la llave pública puede ser compartida con otros usuarios sin restricción.
Para configurar la autenticación mediante llaves SSH, debes colocar la llave pública del usuario en un directorio específico dentro del servidor. Cuando el usuario se conecta al servidor, éste requerirá una prueba de que el cliente tiene la llave privada asociada. El cliente SSH hará uso de la llave privada, respondiendo de tal forma que comprobará que se es propietario de la llave privada. A continuación, el servidor permitirá al cliente la conexión sin el uso de contraseña. Si deseas aprender más acerca de cómo funcionan las llaves SSH, puedes referirte al siguiente artículo.
¿Cómo éstas mejoran la seguridad?
Al usar SSH, cualquier tipo de autentificación, incluyendo la autenticación mediante contraseña, estará totalmente encriptada. Ahora bien, al permitir autenticaciones basadas en contraseñas, usuarios maliciosos podrían realizar intentos repetitivos de acceso al servidor. Gracias al poder computacional actual, es posible acceder a un servidor mediante intentos automáticos de ingreso de contraseñas, una palabra clave tras otra, hasta hallar la que es válida para ese servidor.

¿Qué tan difícil es implementarlas?
Configurar llaves SSH es muy sencillo, y su uso es la práctica recomendada al acceder remotamente a un ambiente de servidores Linux o Unix. Un par de llaves SSH pueden ser generadas en tu propia máquina y puedes transferir la llave pública a tus servidores en pocos minutos.
Para aprender cómo configurar las llaves, puedes seguir esta guía. En el caso que aún sientas que debes usar autenticación mediante contraseña, puedes considerar implementar una solución como: fail2ban en tus servidores, de tal manera que limites la posibilidad de adivinar las contraseñas.
Cortafuegos
Un cortafuegos es una pieza de software (o hardware) que controla cuáles servicios se encuentran expuestos a la red. Es decir, que bloquean o restringen el acceso a todo puerto exceptuando únicamente aquellos que deben estar habilitados para el público.
Típicamente, en un servidor se encuentran diferentes servicios ejecutándose por defecto. Éstos pueden ser categorizados dentro de los siguientes grupos:

- Servicios públicos que pueden ser accedidos sin restricción en internet, normalmente de manera anónima. Un buen ejemplo de ésto es el servidor web que probablemente da acceso a su sitio.
- Servicios privados que solo deberían ser accedidos por un grupo selecto de cuentas autorizadas o desde lugares específicos. Un ejemplo de éstos, puede ser el panel de control de una base de datos.
- Servicios internos que solo deberían ser accedidos desde el mismo servidor, sin exponer el servicio al mundo exterior. Por ejemplo, éstos podrían ser una base de datos que solo acepta conexiones locales.

El cortafuegos puede asegurar que tu software tiene las restricciones acorde con las anteriores categorías. Los servicios públicos pueden ser abiertos sin restricción y disponibles para todos, por su lado, los servicios privados se pueden restringir basándose en diferentes criterios. Los servicios internos se pueden configurar de tal manera que sean completamente inaccesibles al mundo exterior. Para los puertos que no se encuentren en uso, la configuración más común es un bloqueo completo al acceso.

- Investigar porque utilizamos el puerto 8009 y para qué sirve el puerto 80 y 8080

PUERTO 80
En el ámbito de la informática, se conoce como Puerto 80 al que puerto por default, por el medio del cual un servidor HTTP “escucha” la petición hecha por un cliente, es decir por una PC en específico. De acuerdo a los expertos, todas aquellas aplicaciones que funcionan en base a la IP (bien si son TCP o UDP) establecen comunicación con un servidor específico (puede ser SMTP, FTP, TELNET o HTTP, etc.) a través de un puerto, en el caso del HTTP, ese puerto es el 80. Así que mientras la PC de cada uno ocupa un puerto aleatorio, al momento de originar una petición al servidos, en el caso del HTTP siempre será, indistintamente el puerto 80, el que escuche o reciba la solicitud de servicio hecha por la PC cliente.
HTTP (Puerto 80)
El protocolo de transferencia de hipertexto (HTTP, HyperText Transfer Protocol) es el protocolo usado en cada transacción de la Web (WWW). HTTP fue desarrollado por el consorcio W3C y la IETF. HTTP define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Al cliente que efectúa la petición (un navegador o un spider) se lo conoce como «user agent» (agente del usuario). A la información transmitida se la llama recurso y se la identifica mediante un URL. Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento, etc.
HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. El desarrollo de aplicaciones web necesita frecuentemente mantener estado. Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente. Esto le permite a las aplicaciones web instituir la noción de «sesión», y también permite rastrear usuarios ya que las cookies pueden guardarse en el cliente por tiempo indeterminado.

Cómo comprobar el puerto 80

Así mismo, algunos expertos en informática aconsejan que para tener señas del puerto 80, se puede realizar un procedimiento bastante simple. Bastará con encender la PC personal, no abrir ningún otro explorador que no sea Internet Explorer, al tiempo en que se mantienen cerradas otras ventanas de funciones. Posteriormente el usuario deberá abrir también la ventana del MSDOS, en donde debe introducir el comando netstar –n, con lo cual se obtendrá una fila de cuatro columnas, por ejemplo:

TCP 134. 293.1.2:4569 55.798.7.567:80 ESTABLISHED

Con respecto a esta información, la primera columna estaría indicando el puerto de origen, puede ser TCP, como UDP. En segundo lugar, se señala la IP de la PC de donde está saliendo la información. Por su parte, la tercera columna indica el IP correspondiente al servidor de Google, y el puerto 80, que es el que escucha la petición que ha salido de la PC. Finalmente, la última columna señala si se ha establecido efectivamente la comunicación entre la PC y el servidor. Así se puede tener señas del Puerto 80, ubicado en el Servidor, que recibe las peticiones de los equipos remotos.

PUERTO 8080

Abrir o cerrar puertos de internet 8080 – puertosabiertos.com. Una forma de navegar de forma más privada por Internet, ya que el servidor oculta tu IP al navegar por Internet.

El puerto por defecto para los servicios HTTP es el 80, y ahi puedes correr el IIS, el Apache, el Tomcat, el XAMPP o lo que quieras. La cuestión del 8080 es que Tomcat supone que ya tienes algo en el puerto 80, el Apache, el IIS o el XAMPP, y para evitar problemas viene pre-configurado con el puerto 8080. Pero después se lo cambias sin más problemas y listo.

- **Variantes del servicio apache2**

El servidor es de una arquitectura modular que está formado por un Core que soporta las funciones básicas comunes y luego una serie de módulos propios y de terceros que extienden su funcionalidades que se pueden activar o desactivar en una instalación según se requiera.

Entre los más conocidos podemos encontrar auth_basic y mod_rewrite que son los que un programador utiliza más sin embargo la lista es mucho más extensa, la lista completa de módulos se puede ver en índice de módulos disponible en la documentación de Apache.

Entre los módulos más importantes podemos encontrar los módulos MPM que definen la arquitectura interna y cómo se distribuye el trabajo puede variar según el módulo de multiprocesamiento que se utilice entre los que podemos elegir:

- Multiprocesamiento Prefork (mpm-prefork).
- Multiprocesamiento Worker (mpm-worker).
- Multiprocesamiento Event (mpm-event).
- Multiprocesamiento ITK (mpm-itk).

Cuando se habla de multiprocesamiento podemos referirnos simplemente como MPM, estos módulos podemos tener varios aunque solo puede estar uno funcionando al mismo tiempo, estos módulos son los que se encargan de procesar las requests HTTP, administrar los procesos y los diferentes hilos de ejecución del servicio.

La elección del módulo es una decisión crucial ya que determinara que tan bien el buen funcionamiento o no del servidor dependiendo del uso que se le quiera dar al servidor.

mpm-prefork

El módulo usado por defecto para procesamiento es mpm-prefork que abre diferentes procesos para organizar el trabajo, se considera el más seguro ya que existen ciertas configuraciones y módulos que no son seguros de usar con procesamiento por hilos por lo que es más seguro usarlo con mpm-prefork que en lugar de abrir hilos abre procesos independientes.

Si bien se gana seguridad también es el que más recursos consume ya que los procesos independientes consumen mucho más CPU y memoria RAM que los hilos.

mpm-workera en el uso de recursos, tanto de memoria como de CPU

mpm-event

Estas peticiones no son más que mensajes sin contenido de tamaño insignificante cuyo única función mantener con vida la comunicación para no tener que volver al proceso de negociación tanto a nivel de protocolos de red como los procesos internos que el propio servidor ejecutar para responder a una petición.
Por ejemplo como puede ser abrir un nuevo hilo de ejecución para atender dicha petición, mientras la conexión siga abierta continuará respondiendo el mismo hilo, si la conexión se cerrará la próxima solicitud abrirá un hilo nuevo que generaría tiempo de proceso adicional.
Es en ese sentido es qué Event supera a Worker ya por el resto comparten las ventajas y desventajas ya que Event está basado en Worker únicamente con la mejora de estas requests.
mpm-itk
Este es el modulo mas reciente y al igual que Prefork trabaja con procesos hijos en lugar de hilos, la principal innovación de este módulo es que permite asignar a cada VirtualHost (cada dominio alojado) un usuario para generar aislamiento y seguridad.
Esta forma de separar los sitios con diferentes usuarios permite que cada sitio tenga sus propios permisos de seguridad y que los procesos de los usuarios no puedan interactuar entre si obteniendo privacidad y seguridad de los datos.
Lo mismo puede obtenerse PHP-FPM donde cada sitio puede correr con su respectivo usuario sin embargo la configuración de cada uno es más engorrosa e implica un mayor uso de memoria al existir un proceso PHP-FPM por cada sitio.

- **Ejemplo sudo systemctl status apache2**

systemdes un sistema init y un administrador del sistema que se ha convertido en el nuevo estándar para las distribuciones Linux. Debido a su gran adopción, merece la pena familiarizarse con systemd, ya que hará que administrar servidores sea mucho más fácil. Conocer y utilizar las herramientas y daemons que componen systemdle ayudarán a apreciar mejor la potencia, la flexibilidad y las capacidades que proporciona, o al menos a simplificar su trabajo.
En esta guía, hablaremos del comando systemctl, que es la herramienta de administración central para controlar el sistema init. Explicaremos cómo administrar servicios, comprobar estados, cambiar estados del sistema y trabajar con los archivos de configuración.
Tenga en cuenta que aunque systemdes el sistema init predeterminado para muchas distribuciones Linux, no se implementa universalmente en todas las distribuciones. A medida que avanza en este tutorial, si su terminal arroja el error bash: systemctl is not installed, es probable que su equipo tenga un sistema diferente instalado.
La finalidad principal de un sistema init es inicializar los componentes que deben iniciarse tras arrancar el kernel Linux (típicamente conocidos como componentes “userland”). El sistema init también se utiliza para administrar servicios y daemons para el servidor en cualquier momento mientras se ejecuta el sistema. Eso teniendo en cuenta, comenzaremos con algunas operaciones básicas de administración de servicio.
En systemd, el destino de la mayoría de las acciones son “unidades”, que son recursos que systemdsabe cómo administrar. Las unidades se categorizan por el tipo de recurso al que representan y se definen con archivos conocidos como archivos de unidad. El tipo de cada unidad puede deducirse del sufijo al final del archivo.
Para las tareas de administración de servicio, la unidad de destino será unidades de servicio, que tienen archivos de unidad con un sufijo .service. Sin embargo, para la mayoría de los comandos de administración de servicio, puede dejar fuera el sufijo .service, ya que systemdes lo suficientemente inteligente para saber que probablemente quiere operar sobre un servicio cuando utiliza comandos de administración de servicio.

Para iniciar un servicio systemd, ejecutar en el archivo de la unidad del servicio, utilice el comando start. Si está eliminado como usuario non-root, tendrá que usar sudo, ya que esto afectará al estado del sistema operativo.

1. sudo systemctl start application.service

Como hemos mencionado antes, systemdsabe buscar los archivos *.servicepara los comandos de administración de servicio, de forma que el comando podría escribirse fácilmente así:

1. sudo systemctl start application

Aunque puede usar el formato anterior para la administración general, para mayor claridad, usaremos el sufijo .servicepara el resto de los comandos, con el objetivo de ser explícitos sobre el destino en el que estamos operando.
Para detener un servicio que se esté haciendo actualmente, puede usar el comando stop:

1. sudo systemctl stop application.service

Reiniciar y volver a cargar
Para reiniciar un servicio en ejecución, puede usar el comando restart:

1. sudo systemctl restart application.service

Si la aplicación en cuestión puede volver a cargar sus archivos de configuración (sin reiniciar), puede emitir el comando reloadpara iniciar ese proceso:

1. sudo systemctl reload application.service

Si no está seguro de si el servicio tiene la funcionalidad de volver a cargar su configuración, puede emitir el comando reload-or-restart. Esto volverá a cargar la configuración en vigor, si está disponible. De lo contrario, reiniciará el servicio de forma que se recoja la nueva configuración:

1. sudo systemctl reload-or-restart application.service

- **El comando systemctl para que es empleado, mencione tres ejemplos**

Iniciar y detener servicios
Para iniciar servicios utilizando el comando systemctl, solo habrá que ejecutar algo como el siguiente comando:
sudo systemctl start application.service
También podemos hacer referencia al nombre de la aplicación sin el .service final. Para detener el servicio, el comando a utilizar será algo como:

sudo systemctl stop application.service
Reiniciar y recargar servicios
Si buscas reiniciar el servicio, debes escribir en la terminal algo como:

sudo systemctl restart application.service
Para recargar el servicio, el comando a utilizar será:

sudo systemctl reload application.service
Al recargar un servicio solo se vuelven a cargar los cambios de configuración en un servicio en ejecución y no se reiniciará por completo el servicio. Para reiniciar completamente un servicio en ejecución, lo ideal es utilizar la opción restart.

Habilitar y deshabilitar servicios

Si queremos deshabilitar o habilitar un servicio, no habrá más que utilizar los siguientes comandos.
Habilitar un servicio nos permitirá que se inicie automáticamente cada vez que se inicie el servidor.
Para habilitar un servicio el comando que debemos utilizar debe ser algo como:
1
sudo systemctl enable application.service
Si deshabilitamos un servicio, el servicio no se ejecutará a menos que lo volvamos a habilitar. Para deshabilitar un servicio el comando debe ser:
1
sudo systemctl disable application.service

- **Porque se creó la carpeta public_html**

El directorio public_html es la raíz web para el nombre del dominio principal.
Esto significa que public_html es la carpeta donde se colocan todos los archivos del sitio web que se desea aparezcan cuando alguien escribe el dominio principal.
Dicho de otra manera, cuando alguien escribe el nombre de tu dominio en el navegador, lo que está en la carpeta public_html se le será mostrado.

- **Qué pasa si se modifica el permiso de la carpeta public_html a 750**

Permisos:
La carpeta public_html debe siempre tener 0750 permisos.
Todas las carpetas que se encuentren dentro de la carpeta public_html deben tener 0755 permisos.
Todos los archivos dentro de la carpeta public_html deben tener 0755 o 0644 permisos.

Importante: Si eliminas este directorio por error, puedes volver a crearlo tu mismo a través de FTP o del cPanel > Administrador de Archivos

- Conclusiones

Realizar las diversas prácticas de esta materia me ha dado a conocer varios comandos que puedo usar en lo largo de mi carrera, temas que nunca en mi vida he visto y aprenderlo fue una gran experiencia, todos están en ingles, eso también me ha ayudado a mejorar mis conocimientos en este idioma.
El profesor fue de gran ayuda en este curso, ya que resolvía todas mis dudas en clase y fuera de este.

REFERENCIAS.
https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/ch-ssh.html#:~:text=SSH%E2%84%A2%20(o%20Secure%20SHell,conectarse%20a%20un%20host%20remotamente.

https://styde.net/7-medidas-de-seguridad-para-proteger-tu-servidor/

https://es.sawakinome.com/articles/operating-system/difference-between-adduser-and-useradd.html

https://www.drivemeca.com/comando-useradd-linux/

https://www.hostinger.mx/tutoriales/que-es-apache/

https://www.ibm.com/docs/es/rational-build-forge/7.1.2?topic=components-apache-http-server-installation-configuration

https://interpolados.wordpress.com/2017/06/28/puerto-80-y-8080/#:~:text=En%20el%20%C3%A1mbito%20de%20la,por%20una%20PC%20en%20espec%C3%ADfico.

https://blog.infranetworking.com/que-es-apache-servidor/

https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units-es

https://ubunlog.com/systemctl-trabaja-servicios-terminal/#Ejemplos_de_systemctl

https://c.neolo.com/knowledgebase/36/Sobre-el-directorio-publichtml.html#:~:text=El%20directorio%20public_html%20es%20la,alguien%20escribe%20el%20dominio%20principal.