**Introducción a Tumbleweed:**

OpenSUSE tumbleweed es una distribución de linux del tipo rolling release, significa que ofrece actualizaciones constantes y continuas de software,característica la cual vuelve a este sistema muy atractivo ya que permite estar al día con las últimas tecnologías y actualizaciones, a pesar de ser una distribución de tipo rolling release tumbleweed no sacrifica la estabilidad, la comunidad de openSUSE se encarga de garantizar que las actualizaciones de software que se introducen sean probadas y validadas adecuadamentes antes de ser liberadas.

Relación con openSUSE: openSUSE tumbleweed es un proyecto oficial dentro de openSUSE, por lo tanto está mantenido y respaldado por la comunidad de desarrolladores y colaboradores de openSUSE, haciendo que tumbleweed se beneficie de la experiencia y el conocimiento colectivo de la comunidad que tiene openSUSE en el desarrollo y mantenimiento.

**Ventajas de utilizar una distribución rolling release como Tumbleweed:**

* Tumbleweed ofrece un acceso inmediato a las últimas versiones de software, bibliotecas y características, lo que es beneficioso para usuarios que requieran estar al día con las últimas innovaciones y parches de seguridad.
* No es necesario realizar una actualización completa del sistema operativo,mientras que en distribuciones estáticas como Ubuntu LTS o CentOS, se necesita realizar una actualización a una versión completamente nueva cada pocos años, lo que puede ser un proceso tedioso y que requiere tiempo, mientras que en rolling release, las actualizaciones son continuas y automáticas, lo que simplifica este proceso.
* Se le ofrece al usuario mayor control sobre las actualizaciones, ya que el usuario puede elegir cuándo instalar actualizaciones y qué componentes desea actualizar, esto es extremadamente útil donde la estabilidad es de suma importancia, dado a que se pueden realizar pruebas antes de aplicar actualizaciones críticas.

**Configuración del servidor:**

* Hardware:

Servidor con suficiente capacidad de procesamiento (CPU) y memoria RAM, dependiendo de la carga de trabajo prevista.

Almacenamiento adecuado para los datos de la aplicación y el sistema operativo.

Conexión de red confiable y ancho de banda adecuado para la aplicación web.

* Software:

openSUSE Tumbleweed: Descarga la última imágen ISO de Tumbleweed desde el sitio web oficial de openSUSE (https://software.opensuse.org/). La instalación de Tumbleweed proporcionará una base sólida para alojar la aplicación web.

Servidor web: Se puede instalar servidores web populares como Apache, Nginx o incluso soluciones más específicas, dependiendo de las necesidades de la aplicación.

Software de aplicación: Instalar y configurar el software de aplicación necesario, como un servidor de bases de datos (por ejemplo, MySQL, PostgreSQL), lenguajes de programación (como PHP, Python o Node.js) y cualquier otra dependencia específica de tu aplicación web.

* Red:

Configuración de red adecuada con IP.

Configuración de DNS para asociar nombres de dominio con la dirección IP del servidor.

Reglas de firewall para permitir el tráfico web entrante y saliente, si es necesario.

Instalación y configuración de Tumbleweed en el servidor:

La instalación de openSUSE Tumbleweed sigue un proceso estándar similar a otras distribuciones de Linux. Se puede realizar una instalación desde una unidad USB o un DVD de arranque siguiendo las instrucciones del asistente de instalación. Durante la instalación, hay que asegurarse de configurar la red y establecer una contraseña segura para el usuario root.

Después de la instalación, realizamos las siguientes configuraciones iniciales:

* Actualiza el sistema:

sudo zypper refresh

sudo zypper update

Configura la red:

Editamos el archivo de configuración de la red en /etc/sysconfig/network/ifcfg-eth0 (o el nombre de tu interfaz de red) para configurar la dirección IP, la máscara de red, la puerta de enlace y los servidores DNS.

Configura el cortafuegos:

Utiliza el comando ‘firewalld’ para configurar las reglas de firewall y permitir el tráfico web (por ejemplo, el puerto 80 o 443) según sea necesario.

Configura los repositorios de software:

Nos aseguramos de que los repositorios de software estén configurados correctamente para Tumbleweed.

* Uso de Zypper para administrar software en Tumbleweed:

Zypper es el gestor de paquetes utilizado en openSUSE Tumbleweed. Se puede utilizar Zypper para buscar, instalar, actualizar y gestionar software en el sistema. Aquí hay algunos comandos comunes:

Buscar un paquete:

sudo zypper search nombre\_paquete

Instalar un paquete:

sudo zypper install nombre\_paquete

Actualizar el sistema:

sudo zypper update

Actualizar un paquete específico:

sudo zypper update nombre\_paquete

Eliminar un paquete:

sudo zypper remove nombre\_paquete

Listar todos los paquetes instalados:

sudo zypper se -i

**Mantenimiento y actualizaciones:**

* **Copia de seguridad**: se le realiza una copia a los datos críticos y configuraciones del servidor, para garantizar que en la aparición de algún problema, poder restaurar el sistema a un estado anterior.
* **Actualizar paquetes de software**:
  + Abrir el el terminal o una conexión SSH al servidor tumbleweed.
  + Actualizar la lista de paquetes disponibles y sus versiones con el siguiente comando: sudo zypper refresh
  + Actualizar todos los paquetes instalados a las últimas versiones disponibles: sudo zypper update
  + Revisar las actualizaciones propuestas y confirmar para instalarlas.
* **Actualización del kernel:**
  + se realiza de manera regular junto con otros paquetes. Sin embargo, para asegurarnos de que estamos utilizando la última versión del kernel, ejecutamos el siguiente comando: sudo zypper in --type pattern kernel
  + Reiniciar el servidor después de instalar el nuevo kernel. Para que los cambios tengan efecto.
* **Comprobación de la estabilidad**: una vez realizadas las actualizaciones, se verifica que el servidor siga siendo estable y que todas las aplicaciones críticas funcionen correctamente.
* **Monitoreo del sistema**: Se utilizan herramientas de monitorización del sistema(por ejemplo:Nagios, Prometheus, Grafana, etc) para supervisar la carga del CPU, el uso de memoria, la utilización de recursos de red y otros indicadores de rendimiento. Esto nos ayuda a identificar cualquier problema que pueda surgir relacionado a las actualizaciones.
* **Programación de actualizaciones regulares**: para mantener el servidor seguro y estable, se programan actualizaciones regulares.Esto lo podemos lograr mediante herramientas como cron para automatizar el proceso de actualización en momentos que sean convenientes(lo recomendado para este proyecto es que se haga semanalmente,)
* **Mantener copias de seguridad actualizadas**: es necesario que se mantengan copias de seguridad actualizadas después de cada actualización importante. De esta manera, siempre tendremos una copia reciente a la que recurrir en caso de problemas o errores.

**Escalabilidad:**

* Escalabilidad vertical (scale-up):
  + Aumento de recursos de hardware: La forma más simple de escalar verticalmente es agregar más recursos de hardware al servidor. Se puede agregar más RAM, CPU, almacenamiento más rápido o incluso migrar a un servidor más potente. Esta es una solución inmediata, pero tiene límites, ya que en algún momento se puede alcanzar la capacidad máxima del hardware.
  + Contenedores y máquinas virtuales: Utilizar contenedores (como Docker) o máquinas virtuales (VM) permite escalar verticalmente al clonar instancias de la aplicación y equilibrar la carga entre ellas. Esto proporciona más flexibilidad y puede ayudar a gestionar cargas de trabajo variables.
* Escalabilidad horizontal (scale-out):
  + Balanceo de carga: Un método común es utilizar balanceadores de carga para distribuir el tráfico entre múltiples servidores web. A medida que aumenta la demanda, puedes agregar más servidores al grupo. Esto es escalabilidad horizontal y es altamente efectivo para manejar cargas de tráfico variables.
  + Servicios en la nube: Las plataformas en la nube, como Amazon Web Services (AWS) o Microsoft Azure, ofrecen la capacidad de escalar horizontalmente de manera dinámica. Puedes configurar autoescalado para agregar o quitar servidores automáticamente según las necesidades.
* Optimización del rendimiento:
  + Caching: Utiliza sistemas de caché para almacenar en memoria datos que se acceden con frecuencia. Esto reduce la carga en la base de datos y acelera las respuestas.
  + Compresión y optimización de imágenes: Optimizando el contenido multimedia, como imágenes y videos, para que se carguen más rápido.
  + Reducción de latencia: Minimiza la latencia de la red utilizando CDNs (Content Delivery Networks) para distribuir contenido estático y ubicando servidores más cerca de los usuarios.
  + Bases de datos escalables: Utiliza bases de datos escalables, como bases de datos NoSQL o sistemas de bases de datos distribuidas, para manejar cargas de datos cada vez mayores.
* Desarrollo y arquitectura eficiente:
  + Microservicios: Dividir la aplicación en componentes más pequeños y manejables, conocidos como microservicios, para facilitar la escalabilidad, ya que podemos escalar los componentes individuales que experimentan una mayor demanda.
  + Optimización de código: Es importante escribir un código eficientemente para minimizar el uso de recursos.
* Monitorización y ajuste constante:
  + Implementar herramientas de monitorización para supervisar el rendimiento del servidor y de la aplicación web en tiempo real. Ajustar la configuración y la capacidad en función de las métricas y la demanda observada.

**Monitoreo y registro:**

* Prometheus:
  + Prometheus es un sistema de monitoreo y alerta de código abierto que se utiliza ampliamente para recopilar métricas y supervisar sistemas y aplicaciones.
  + Configuración: Configura Prometheus para recopilar métricas de tu aplicación mediante la exportación de métricas en un formato compatible con Prometheus (por ejemplo, mediante bibliotecas específicas en tu código). Luego, configura reglas de alerta para recibir notificaciones cuando se superan umbrales específicos.
* Grafana:
  + Grafana es una plataforma de análisis y visualización que se utiliza comúnmente junto con Prometheus para crear paneles de control y visualizar datos de rendimiento.
  + Configuración: Conecta Grafana a Prometheus u otras fuentes de datos de monitoreo. Crea paneles de control personalizados con gráficos y alertas para rastrear métricas clave de tu aplicación.
* Nagios:
  + Nagios es una herramienta de monitoreo de red de código abierto que se utiliza para supervisar hosts y servicios, incluidos servidores web.
  + Configuración: Configurar Nagios para realizar comprobaciones periódicas de la disponibilidad y el rendimiento de los servicios y recursos de tu servidor. Define umbrales y alertas para recibir notificaciones cuando se detecten problemas.