UNIVERSIDAD VERACRUZANA
DE ESTADÍSTICA E INFORMÁTICA

Carrera: Redes y servicios de cómputo

Tema: Actividades inicio de sistema

Docente: Contreras Vega Gerardo

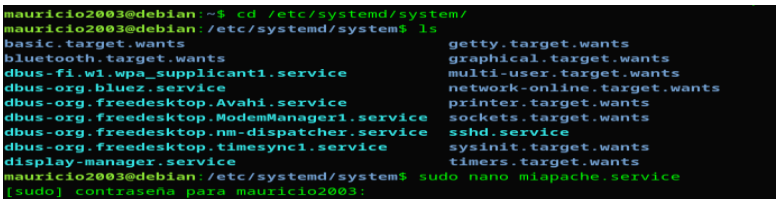
Alumno: Hernández Sánchez Mauricio

Materia: Administración de servidores

Fecha: 29/09/2023

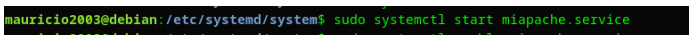
Unidad de servicio SystemD miapache

Primero se coloco el archivo de apache en la carpeta etc.



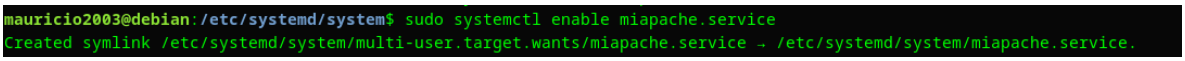
Después se deben de recargar los archivos de configuración de systemd.

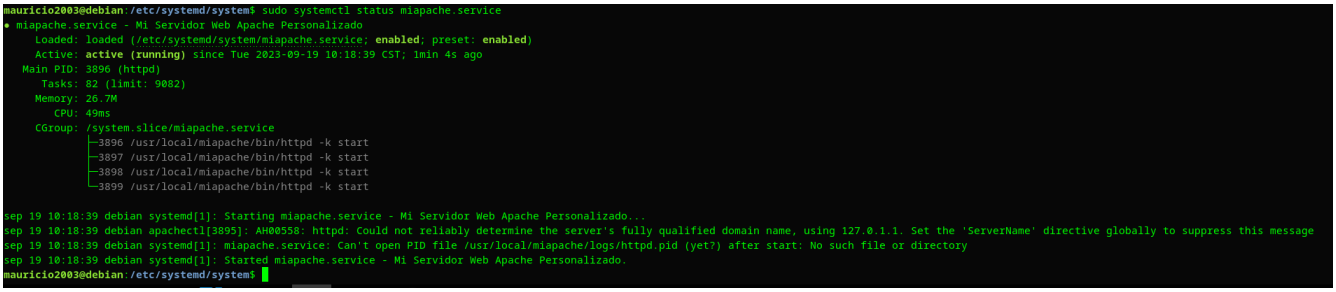
Al realizar el paso anterior, debemos de iniciar el servicio.



Una vez iniciado, se tendrá que habilitar el servicio para que se inicie el arranque.

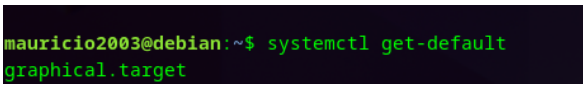
Por último, procedemos a verificar el estado del servicio.



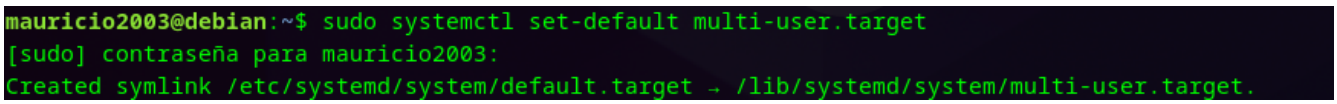


0. Reto cambio de inicio por defecto:
- Descripción: Verifica cual es el modo en que inicia tu Linux por defecto, cambia el modo y comprueba el cambio. Deja por defecto el modo que prefieras.
- Objetivo: Aprender a cambiar el modo por defecto en SystemD.

Primero tenemos que checar como inicia nuestro Linux, que en este caso inicia con un entorno gráfico, al poner el comando systemctl get-default es el que nos indica como inicia nuestro Linux.



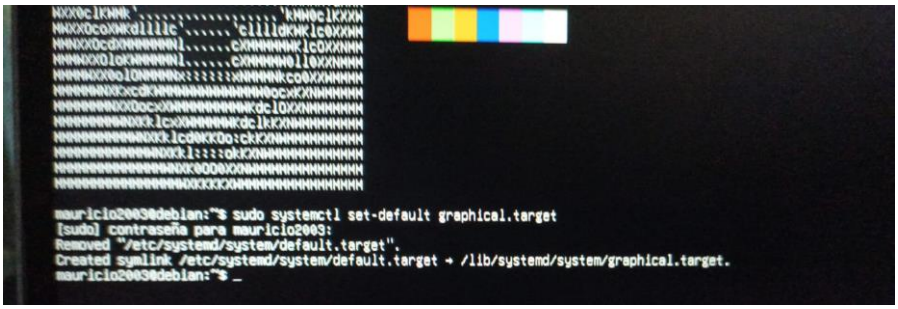
Después tendremos que poner el comando sudo systemctl set-defautl multi-user.target para cambiar a que Linux inicie sin entorno gráfico.



Una vez realizado esto, reiniciamos la maquina para ver si funciono.

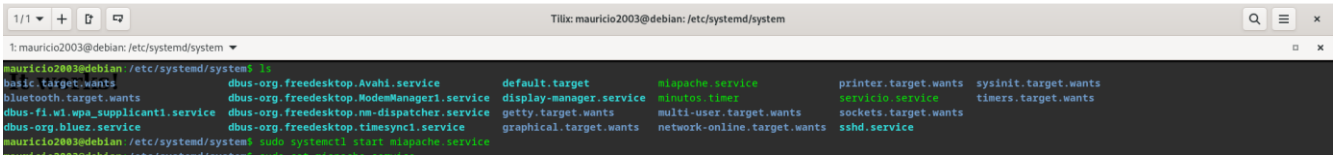
Al reiniciar nos damos cuenta que la maquina prendo sin entorno gráfico.

Para volver a establecer que la maquina inicia con entorno grafico ponemos los siguentes comandos y reiniciamos.



1. Reto del Servidor Web Personalizado:
- Descripción: Crea un servicio para el servidor web instalado desde código fuente.
- Objetivo: Aprender a configurar servicios personalizados en ubicaciones no estándar.

Para hacer esto, primero debo de iniciar algún servidor, en este caso utilice apache. Se debe de poner el código que se muestra continuación.

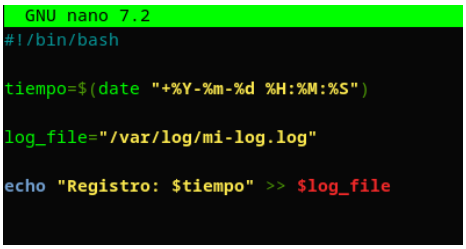


Después debemos de entrar a nuestro buscador y ponemos la local hots que en este caso es 127.0.0.1 y tendremos que dar Enter y debe de mostrar lo siguiente.

Tome como ejemplo el script que vimos en clase de mi apache personalizado.

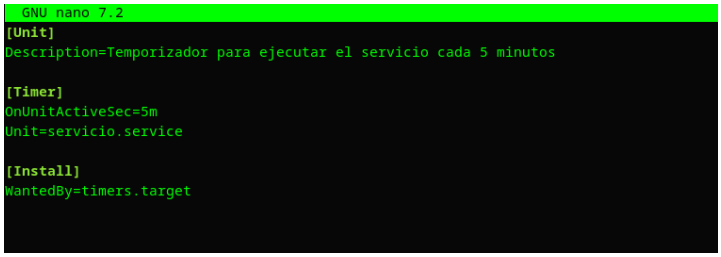
2. Reto del Temporizador:
- Descripción: Escribe un script que genere un archivo de registro con la fecha y hora actual en /var/log/mi-log.log. Luego, deben crear un servicio systemd para ejecutar el script y un temporizador (timer) que ejecute el servicio cada 5 minutos.
- Objetivo: Entender cómo funcionan los temporizadores y cómo pueden ser usados para programar la ejecución de servicios.

Lo primero que se debe de realizar es crear el script en el cual funcionara para que genere un archivo de registro.

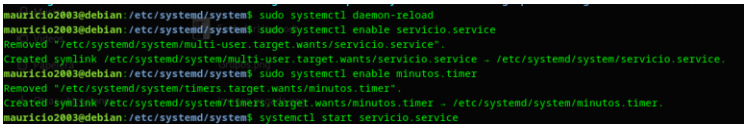


En segundo se crea el servicio

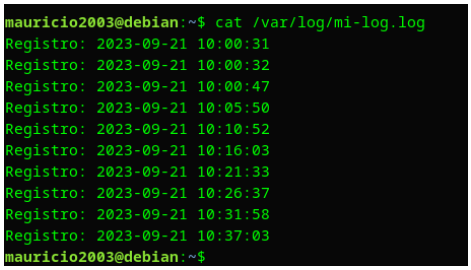
Se tiene que crear el temporizador



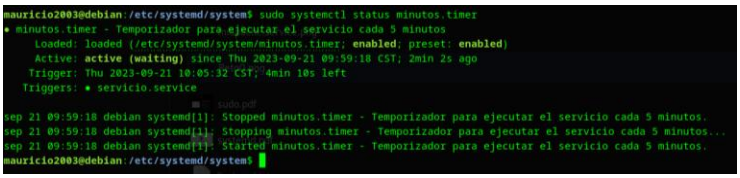
Al tener todo creado, se tiene que activar los servicios, para hacer esto se ejecutan los siguientes comandos que se muestran a continuación.



Después procedemos a ejecutar el archivo log para iniciar el registro

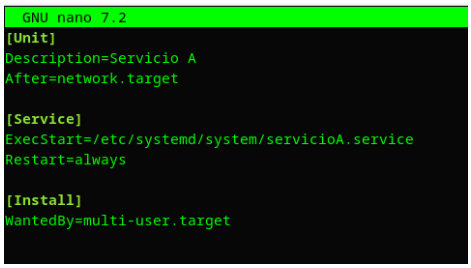


Una forma de ver cuanto tiempo falta para que se realice cada servicio es poniendo este comando, con este comando mostrara los minutos que faltan para que se haga el registro.

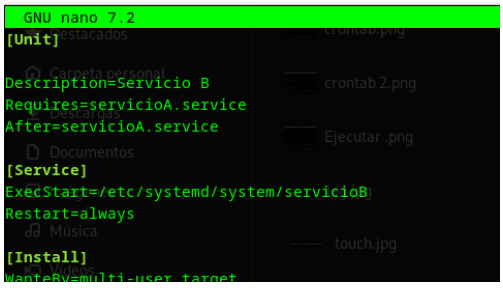


3. Reto del Servicio Dependiente:
- Descripción: Configura dos servicios: servicioA y servicioB. servicioB no debe poder iniciar a menos que servicioA esté en funcionamiento.
- Objetivo: Aprender sobre las dependencias entre servicios y cómo configurarlas usando directivas como Requires, After, entre otras.

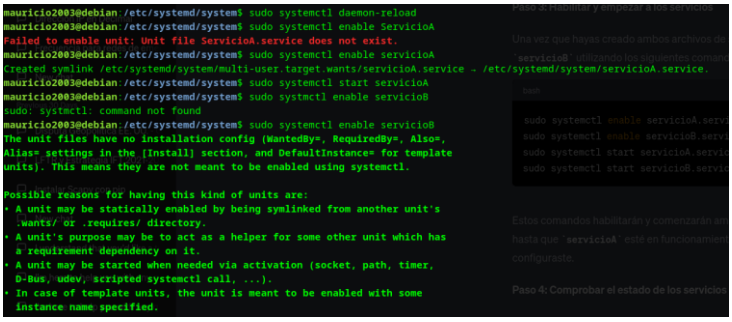
Primero se crea el archivo del servicioA, en este caso lo cree en la ruta de systemd y quedo así.



Despues se crea el archivo del servicioB, así fue como quedo.

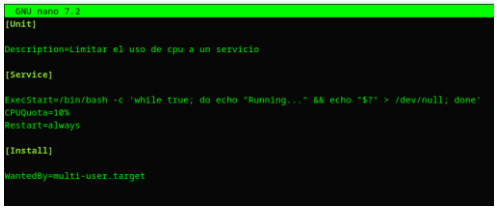


Al tener los dos archivos procedemos a iniciarlos y habilitarlos para que empiecen los servicios, los siguientes comandos que se muestran a continucion son para habilitar los servicios.

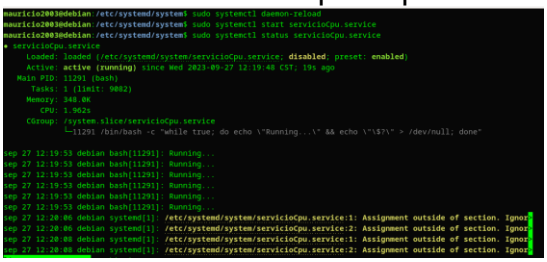


4. Reto de la Restricción de Recursos:
- Descripción: Configurar un servicio que inicie un proceso que consuma una gran cantidad de CPU (por ejemplo, un bucle infinito en Python o Bash). Luego, debes limitar el uso de CPU de ese servicio al 10% utilizando systemd.
- Objetivo: Aprender a gestionar y restringir los recursos (como CPU, memoria) que un servicio puede usar.

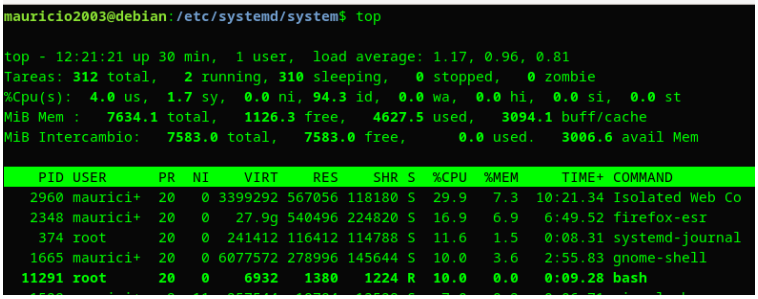
Lo primero que realice fue crear un servicio, el cual quedo así:



Después se tienen que activar todos los servicios para que inicie el programa.



Y para comprobar que el servicio este utilizando el 10% del systemd tenemos que poner un top.



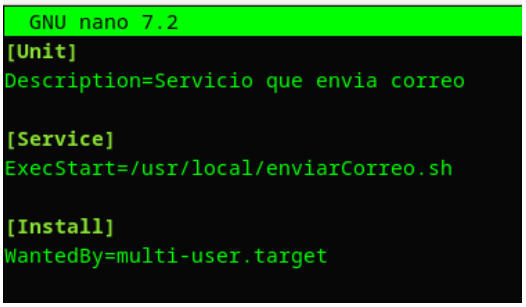
5. Reto del Análisis de Fallos:
- Descripción: Se tiene el siguiente archivo de servicio:

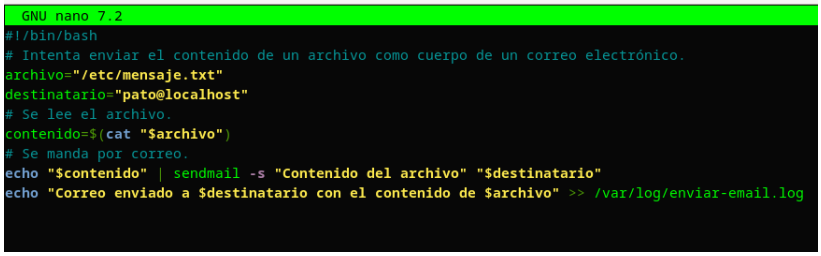
Archivo: enviaCorreo.service
[Unit]
Description=Servicio Fallido
[Service]
ExecStart=/usr/local/enviaCorreo.sh
[Install]

WantedBy=multi-user.target
Archivo enviarCorreo.sh

#!/bin/bash
# Intenta enviar el contenido de un archivo como cuerpo de un correo electrónico.
archivo="/etc/mensaje.txt"
destinatario="pato@localhost"
# Se lee el archivo.
contenido=$(cat $archivo)
# Se manda por correo.
echo "$contenido" | sendmail -s "Contenido del archivo" $destinatario
Se genera una bitácora de la actividad
echo "Correo enviado a $destinatario con el contenido de $archivo" /var/log/enviar-email.log
Identifica y corrije los errores para que el servicio funcione correctamente.
- Objetivo: Familiarizarse con las herramientas de diagnóstico y registro de systemd, como systemctl
status, journalctl, entre otras, y aprender a solucionar problemas comunes.

El error que tenia el script era la redirección del var log, el script y el servicio quedaron de esta manera.





Al corregir el script, procedemos a dar inicio a los servicios.