

Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías

Computación Tolerante a Fallas



Maestro: Michel Emanuel Lopez Franco

Juan Antonio Perez Juarez

Carrera: INCO

Código: 215660996

Status to Daemons

Introducción

En el ámbito de los sistemas operativos, particularmente en entornos Unix y Linux, los daemons desempeñan un papel fundamental en la operatividad y gestión de servicios en segundo plano. Estos procesos, que funcionan de manera autónoma y sin interacción directa con el usuario, son responsables de tareas críticas como la gestión de redes, la ejecución de servicios de impresión, la administración de bases de datos y la supervisión de sistemas, entre otros. A diferencia de los procesos convencionales, los daemons están diseñados para operar de manera continua, asegurando que los servicios esenciales estén siempre disponibles.

El estudio de los daemons en Linux es especialmente relevante debido a la naturaleza abierta y modular de este sistema operativo, que permite una amplia personalización y control sobre estos procesos. Comprender su funcionamiento, configuración y gestión no solo es crucial para administradores de sistemas y desarrolladores, sino también para cualquier profesional que busque optimizar el rendimiento y la seguridad de un entorno Linux.

Este documento tiene como objetivo explorar en profundidad el concepto de daemons en Linux, analizando su arquitectura, ciclo de vida, y las herramientas disponibles para su gestión. Además, se abordarán aspectos relacionados con la seguridad y las mejores prácticas para garantizar un funcionamiento eficiente y seguro de estos procesos en segundo plano.

Desarrollo

1. Concepto de Daemon

Un daemon es un tipo de proceso en sistemas Unix y Linux que se ejecuta en segundo plano, sin estar asociado a una terminal o interfaz de usuario. Su nombre proviene de la mitología griega, donde los "daimones" eran seres que actuaban como intermediarios entre los dioses y los humanos. En el contexto informático, los daemons actúan como intermediarios entre el sistema operativo y los servicios que gestionan.

2. Características de los Daemons

Ejecución en segundo plano: Los daemons no requieren interacción directa con el usuario y operan de manera independiente.

Desvinculación de la terminal: Para evitar ser afectados por cierres de sesión o terminales, los daemons se desvinculan de la terminal que los inició.

Ciclo de vida prolongado: Están diseñados para ejecutarse continuamente, desde el inicio del sistema hasta su apagado.

Gestión de servicios: Proporcionan funcionalidades como servidores web, de correo, de bases de datos, entre otros.

3. Arquitectura de los Daemons en Linux

En Linux, los daemons siguen un patrón de diseño específico:

Inicialización: El proceso se bifurca (fork) y el proceso padre termina, permitiendo que el proceso hijo se ejecute en segundo plano.

Desvinculación de la terminal: El proceso hijo llama a `setsid()` para crear una nueva sesión y desvincularse de la terminal.

Cambio de directorio: El daemon cambia su directorio de trabajo a la raíz (`/`) para evitar problemas con puntos de montaje.

Cierre de descriptores de archivo: Se cierran los descriptores de archivo heredados para liberar recursos.

4. Gestión de Daemons en Linux

Linux ofrece varias herramientas y métodos para gestionar daemons:

Systemd: El sistema de inicio más común en distribuciones modernas de Linux, que permite gestionar daemons como unidades de servicio.

SysVinit: Un sistema de inicio más antiguo que utiliza scripts en `/etc/init.d/` para gestionar daemons.

Supervisores de procesos: Herramientas como `supervisord` o `monit` permiten monitorear y reiniciar daemons automáticamente.

5. Seguridad y Mejores Prácticas

Los daemons, al estar siempre activos, pueden ser un objetivo para ataques. Algunas prácticas recomendadas incluyen:

Ejecución con privilegios mínimos: Limitar los permisos del daemon para reducir el impacto de posibles vulnerabilidades.

Registro de actividades (logging): Implementar logs para monitorear el comportamiento del daemon y detectar anomalías.

Actualizaciones y parches: Mantener los daemons y sus dependencias actualizados para evitar vulnerabilidades conocidas.

6. Aplicaciones Comunes de Daemons

Algunos ejemplos de daemons ampliamente utilizados en Linux incluyen:

sshd: Servidor SSH para conexiones remotas seguras.

httpd/apache2: Servidor web Apache.

mysqld: Servidor de bases de datos MySQL.

cron: Daemon para la ejecución programada de tareas.

Para la implementación de este problema, hice 2 scripts e hice uso de linux, por la facilidad de levantar un daemon, por que se hace desde bash.

El primer script es un daemon que hace una carita en la consola.

Código fuente:

```
Python
import os
import time
import curses

def run_animation(stdscr):
    """Ejecuta la animación en pantalla."""
    curses.curs_set(0) # Oculta el cursor
    stdscr.nodelay(True) # Hace que getch() no bloquee
    stdscr.timeout(300) # Controla la velocidad de la animación
```

```

frames = [
    "  (o_o)  ",
    "  (-_-)  ",
    "  (o_o)  ",
    "  (^_^)  ",
]

idx = 0
while True:
    stdscr.clear()
    stdscr.addstr(5, 10, frames[idx]) # Muestra el frame en la posición (5,10)
    stdscr.refresh()
    idx = (idx + 1) % len(frames) # Cambia de frame

    key = stdscr.getch()
    if key == ord('q'): # Permite salir con 'q'
        break

if __name__ == "__main__":
    curses.wrapper(run_animation)

```

Ahora, el segundo script, lo que hace es ser el monitor del primer daemon, que muestra en pantalla si el daemon anterior está activo o no.

```

Python
import os
import time
import psutil
import curses

DAEMON_NAME = "ascii_daemon.py"

def is_process_running(process_name):
    """Verifica si el proceso está en ejecución."""
    for proc in psutil.process_iter(['pid', 'name', 'cmdline']):
        try:
            if process_name in " ".join(proc.info['cmdline']):
                return True
        except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):
            pass
    return False

def monitor_daemon(stdscr):
    """Monitorea el demonio y muestra su estado con colores en curses."""
    curses.curs_set(0) # Ocultar cursor
    stdscr.nodelay(True)
    stdscr.timeout(1000) # Refrescar cada segundo

    # Inicializar colores
    curses.start_color()

```

```

curses.init_pair(1, curses.COLOR_GREEN, curses.COLOR_BLACK) # Verde
curses.init_pair(2, curses.COLOR_RED, curses.COLOR_BLACK)  # Rojo

while True:
    stdscr.clear()
    h, w = stdscr.getmaxyx() # Obtener tamaño de pantalla
    msg = "Presiona 'q' para salir"

    if is_process_running(DAEMON_NAME):
        status = "● Demonio en ejecución"
        color = curses.color_pair(1)
    else:
        status = "● Demonio detenido"
        color = curses.color_pair(2)

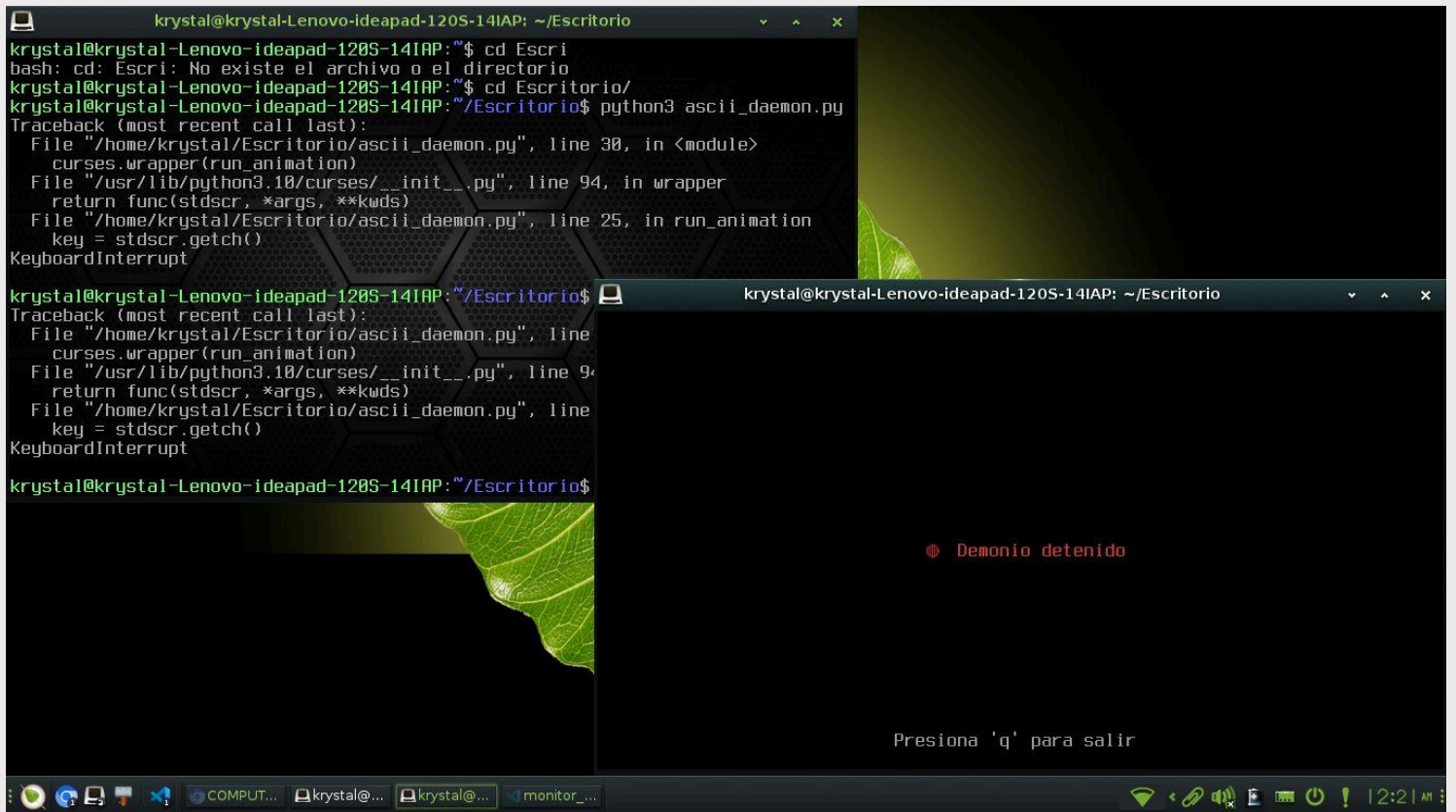
    # Centrar texto en pantalla
    stdscr.addstr(h // 2, (w - len(status)) // 2, status, color)
    stdscr.addstr(h - 2, (w - len(msg)) // 2, msg, curses.A_DIM)
    stdscr.refresh()

    key = stdscr.getch()
    if key == ord('q'): # Salir con 'q'
        break

if __name__ == "__main__":
    curses.wrapper(monitor_daemon)

```

Pruebas del Funcionamiento:

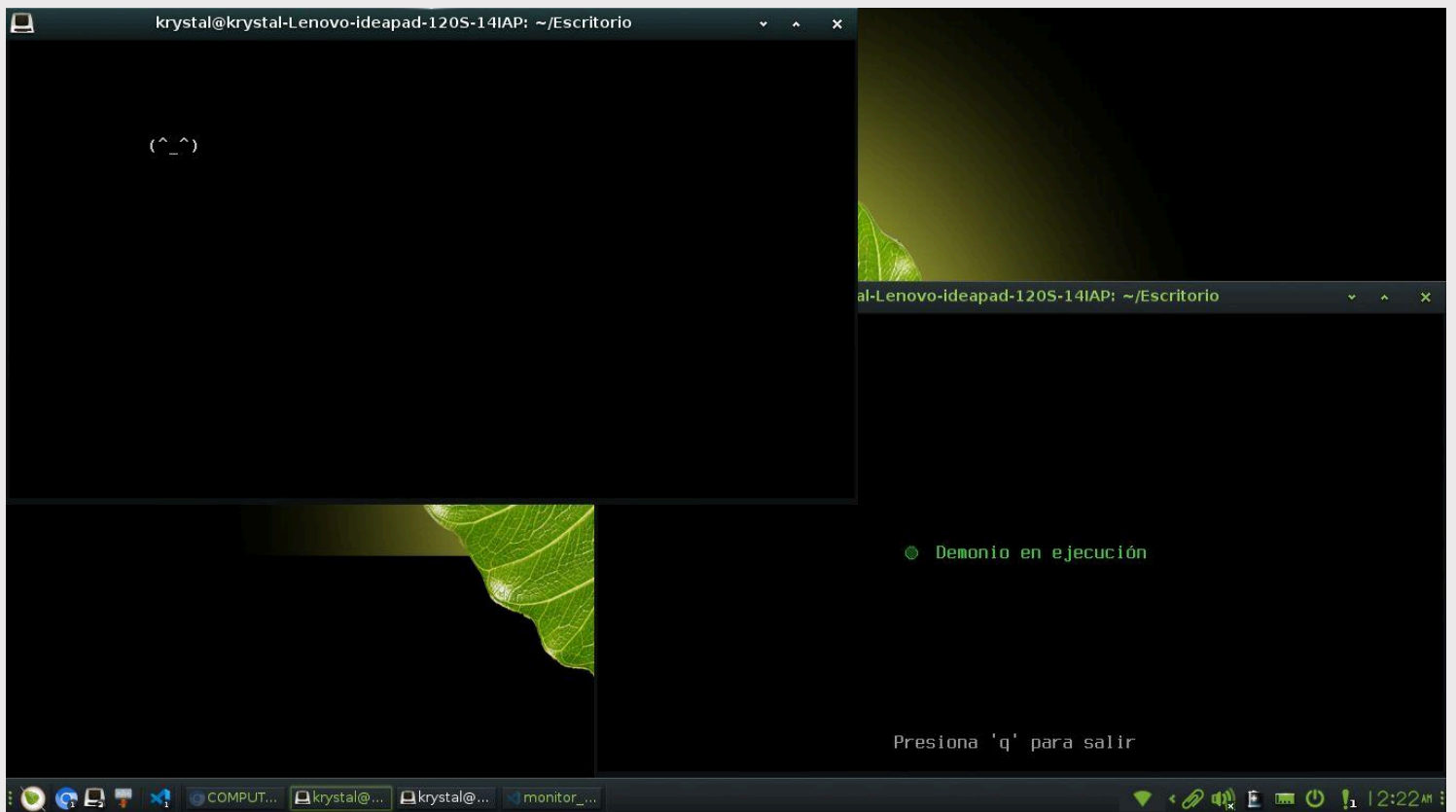


```
krystal@krystal-Lenovo-ideapad-120S-14IAP: ~/Escritorio
krystal@krystal-Lenovo-ideapad-120S-14IAP:~$ cd Escri
bash: cd: Escri: No existe el archivo o el directorio
krystal@krystal-Lenovo-ideapad-120S-14IAP:~$ cd Escritorio/
krystal@krystal-Lenovo-ideapad-120S-14IAP:~/Escritorio$ python3 ascii_daemon.py
Traceback (most recent call last):
  File "/home/krystal/Escritorio/ascii_daemon.py", line 30, in <module>
    curses.wrapper(run_animation)
  File "/usr/lib/python3.10/curses/_init_.py", line 94, in wrapper
    return func(stdscr, *args, **kws)
  File "/home/krystal/Escritorio/ascii_daemon.py", line 25, in run_animation
    key = stdscr.getch()
KeyboardInterrupt

krystal@krystal-Lenovo-ideapad-120S-14IAP:~/Escritorio$
Traceback (most recent call last):
  File "/home/krystal/Escritorio/ascii_daemon.py", line
curses.wrapper(run_animation)
  File "/usr/lib/python3.10/curses/_init_.py", line 94
return func(stdscr, *args, **kws)
  File "/home/krystal/Escritorio/ascii_daemon.py", line
key = stdscr.getch()
KeyboardInterrupt

krystal@krystal-Lenovo-ideapad-120S-14IAP:~/Escritorio$
```

Presiona 'q' para salir



```
krystal@krystal-Lenovo-ideapad-120S-14IAP: ~/Escritorio
(^_^)
```

Presiona 'q' para salir

Conclusión

En resumen, los daemons son procesos esenciales en Linux que trabajan en segundo plano para mantener servicios importantes en funcionamiento, como servidores web, bases de datos o tareas programadas. Aunque no interactúan directamente con el usuario, su papel es fundamental para que el sistema opere de manera eficiente y sin interrupciones.

A través de este tema, aprendí que los daemons tienen un ciclo de vida bien definido, desde su creación hasta su gestión con herramientas como systemd o SysVinit. También es importante considerar aspectos de seguridad, como ejecutar estos procesos con privilegios mínimos y mantenerlos actualizados para evitar vulnerabilidades.

Referencias

Linux Foundation. (2023). Systemd: System and service manager. <https://systemd.io/>

Robbins, A. (2022). Linux Command Line and Shell Scripting Bible. Wiley.

Shotts, W. (2019). The Linux Command Line: A Complete Introduction. No Starch Press.

Tanenbaum, A. S., & Bos, H. (2015). Modern Operating Systems (4th ed.). Pearson.

Torvalds, L., & Linux Community. (2023). Linux Kernel Documentation.
<https://www.kernel.org/doc/html/latest/>