SO-p4.pdf

KIKONASO

Sistemas Operativos

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Práctica 4 - SISTEMAS OPERATIVOS

Actividad 4.1. Consulta de información sobre procesos demonio
A partir de la información proporcionada por la orden ps encuentre los datos asociados a los demonios atd y crond, en concreto: quién es su padre, qué terminal tienen asociado y cuál es su usuario.

Combino las órdenes **ps -ef** (**-e** muestra todos los procesos del sistema. Incluye tanto los procesos del usuario actual como los de otros usuarios y los procesos del sistema y **-f** emplea el formato completo proporcionando más información en cada línea sobre cada proceso, como el PID, PPID, usuario, tiempo de inicio, etc.) con **grep -E "atd|cron"** (podría usarse grep "cron" y grep "atd" en dos comandos, pero con el -E me permite usar | sin usar barras especiales y hacer la búsqueda de ambos parámetros a la vez)

```
ps -ef | grep -E "atd|cron"

root     894      1  0 18:05 ?        00:00:00 /usr/sbin/anacron -d -q -s
root     903      1  0 18:05 ?        00:00:00 /usr/sbin/cron -f -P
tomy   14460  14157  0 18:21 pts/0    00:00:00 grep --color=auto -E atd|cron
```

El **proceso anacron** tiene:

- **PPID**: 1 (su proceso padre es init o systemd).
- **TTY**: ? (no tiene un terminal asociado).
- **Usuario**: root.

El **proceso cron** tiene:

- **PPID**: 1 (su proceso padre es también init o systemd).
- **TTY**: ? (no tiene un terminal asociado).
- **Usuario**: root.

El **comando grep** tiene:

- **PPID**: 14157 (proceso padre que probablemente es la shell o terminal desde la que ejecutaste el comando).
- **TTY**: pts/0 (está asociado a tu terminal interactiva).
- **Usuario**: tomy (el usuario que ejecutó el comando).

Actividad 4.2. Ejecución postergada de órdenes con at (I)

Crea un archivo genera-apunte que escriba la lista de hijos del directorio home en un archivo de nombre listahome-`date +%Y-%j-%T-$$`, es decir, la yuxtaposición del literal "listahome" y el año, día dentro del año, la hora actual y pid (consulte la ayuda de date). Lanza la ejecución del archivo genera-apunte un minuto más tarde de la hora actual. ¿En qué directorio se crea el archivo de salida?

Lo he hecho sin script con el fin de agilizar el uso del terminal, primero ordeno: **at now +1 minute**

Y ya dentro del at hago ls /home > “listahome-`date +%Y-%j-%T-$$`.txt”

O simplemente **ls /home > “listahome-`date +%Y-%j-%T-$$`.txt” | at now +1 minute**

Actividad 4.3. Ejecución postergada de órdenes con at (II)
Lanza varias órdenes at utilizando distintas formas de especificar el tiempo como las siguientes: (será de utilidad la opción -v):

a) a medianoche de hoy

echo “Ejecuto a medianoche” | at -v midnight

b) un minuto después de la medianoche de hoy

echo “Ejecuto a medianoche y un minuto” | at -v midnight +1 minute

c) a las 17 horas y 30 minutos de mañana

echo “Ejecuto mañana” | at -v 17:30 tomorrow

d) a la misma hora en que estemos ahora pero del día 25 de diciembre del presente año

echo "Ejecuto a esta hora el 25 de diciembre" | at -v $(date %H:%M) december 25

e) a las 00:00 del 1 de enero del presente año

echo "Ejecuto a las 00:00 del 1 de enero" | at -v 00:00 1 January

Utiliza las órdenes atq y atrm para familiarizarte con su funcionamiento (consulta la ayuda de estas órdenes).

Actividad 4.4. Cuestiones sobre at
El proceso nuevo que se lanza al cumplirse el tiempo que se especificó en la orden at....

1. ¿qué directorio de trabajo tiene inicialmente? ¿hereda el que tenía el proceso que invocó a at o bien es el home, directorio inicial por omisión?

El proceso que se lanza con at hereda el directorio de trabajo del proceso que invocó a at. Es decir, si ejecutaste el comando at desde un directorio específico, el nuevo proceso comenzará en ese mismo directorio, no en el directorio home por omisión.

2. ¿qué máscara de creación de archivos umask tiene? ¿es la heredada del padre o la que se usa por omisión?

El proceso que se ejecuta a través de at hereda la máscara de creación de archivos (umask) del proceso padre que invocó a at. Esto significa que la umask que tendrá el proceso nuevo es la que estaba configurada en el entorno del usuario en el momento de invocar at, no la configuración por defecto del sistema.

3. ¿hereda las variables locales del proceso padre?

El proceso que se ejecuta a través de at no hereda las variables locales del proceso padre. Cuando se invoca un comando a través de at, se ejecuta en un entorno separado, por lo que las variables de entorno y las variables locales definidas en la sesión del usuario no se pasan al nuevo proceso. Solo se heredan las variables de entorno que están definidas de manera global.

Actividad 4.5. Relación padre-hijo con órdenes ejecutadas mediante at
El proceso nuevo que se lanza al cumplirse el tiempo que se especificó en la orden at.... ¿de quién es hijo? Investiga lanzando la ejecución retardada de un script que muestre la información completa sobre los procesos existentes y el pid del proceso actual; el script podría contener lo que sigue:

```
nombrearchivo=`date +%Y-%j-%T`
ps -ef > $nombrearchivo
echo Mi pid = $$ >> $nombrearchivo
```

El proceso ps -ef tiene un PPID de 14157, lo que significa que fue lanzado por el proceso que estaba en ejecución en ese momento, y que su PID es 14157. Este es un ejemplo clásico de cómo los procesos pueden ser hijos de otros procesos en un sistema operativo basado en Unix.

Actividad 4.6. Script para orden at
Construye un script que utilice la orden find para generar en la salida estándar los archivos modificados en las últimas 24 horas (partiendo del directorio home y recorriéndolo en profundidad), la salida deberá escribirse el archivo de nombre "modificados_<año><día><hora>" (dentro del directorio home). Con la orden at provoque que se ejecute dentro de un día a partir de este momento.

Funcionalidades de repaso de find:

Mostrar Archivos Modificados en las Últimas 24 Horas

Para buscar y mostrar archivos que han sido modificados en las últimas 24 horas, puedes utilizar el siguiente comando:

```
find /ruta/del/directorio -type f -mtime -1
```

Desglose del Comando:

/ruta/del/directorio: Especifica el directorio donde deseas realizar la búsqueda. Puedes usar . para referirte al directorio actual.

-type f: Indica que estás buscando solo archivos (no directorios). Si deseas incluir directorios, puedes omitir esta opción.

-mtime -1: Este parámetro busca archivos modificados en las últimas 24 horas. El -1 significa "modificado en menos de 1 día" (es decir, en las últimas 24 horas).

Ejemplo Completo:

find . -type f -mtime -1

Este comando buscará todos los archivos en el directorio actual y sus subdirectorios que han sido modificados en las últimas 24 horas.
Funcionalidades de find

Buscar Archivos por Nombre:

find /ruta/del/directorio -name "nombre_archivo.txt"

Busca archivos que coincidan exactamente con el nombre especificado. Puedes usar comodines, como *.txt, para buscar todos los archivos de texto.

Buscar por Extensión:

find /ruta/del/directorio -name "*.jpg"

Busca todos los archivos con la extensión .jpg.

Buscar Archivos por Tamaño:

find /ruta/del/directorio -size +100M

Busca archivos que sean mayores a 100 megabytes. Puedes usar -size -100M para archivos menores de 100 MB.

Ejecutar Comandos sobre Resultados:

find /ruta/del/directorio -type f -name "*.log" -exec rm {} \;

Busca archivos con la extensión .log y los elimina. {} se sustituye por cada archivo encontrado, y \; indica el final del comando a ejecutar.

Buscar Archivos por Propietario:

find /ruta/del/directorio -user nombre_usuario

Busca todos los archivos pertenecientes a un usuario específico.

Buscar Archivos por Permisos:

find /ruta/del/directorio -perm 644

Busca archivos que tienen permisos específicos (en este caso, permisos 644).

Buscar Archivos y Directorios Vacíos:

find /ruta/del/directorio -empty

Encuentra archivos y directorios vacíos.

RESPUESTA FINAL: find ~/ -type f -mtime -1 > "modificados_`date +%Y-%j-%T`" | at tomorrow

Para construir un script que se ejecute con una periodicidad de un minuto usando `crontab`, sigue estos pasos:

MANUAL DE CRONTAB

Paso 1: Crear el Script

Primero, necesitas crear el script que deseas ejecutar. Para este ejemplo, vamos a hacer un script simple que registre la fecha y hora actuales en un archivo de texto.

1. **Crea el script**. Puedes usar un editor de texto como `nano` o `vim`. Aquí te muestro cómo hacerlo con `nano`.

```
nano /ruta/a/mi_script.sh
```

2. **Agrega el siguiente contenido** al script:

```
#!/bin/bash
echo "El script se ejecutó a las: $(date)" >> /ruta/a/log.txt
```

Asegúrate de reemplazar `/ruta/a/` con la ruta donde quieres guardar tu script y el archivo de log.

3. **Guarda y cierra el editor**. Si estás usando `nano`, presiona `CTRL + X`, luego `Y` para confirmar los cambios y `Enter`.

4. **Dale permisos de ejecución al script**:

```
chmod +x /ruta/a/mi_script.sh
```

1/6

Este número es indicativo del riesgo del producto, siendo 1/6 indicativo de menor riesgo y 6/6 de mayor riesgo.

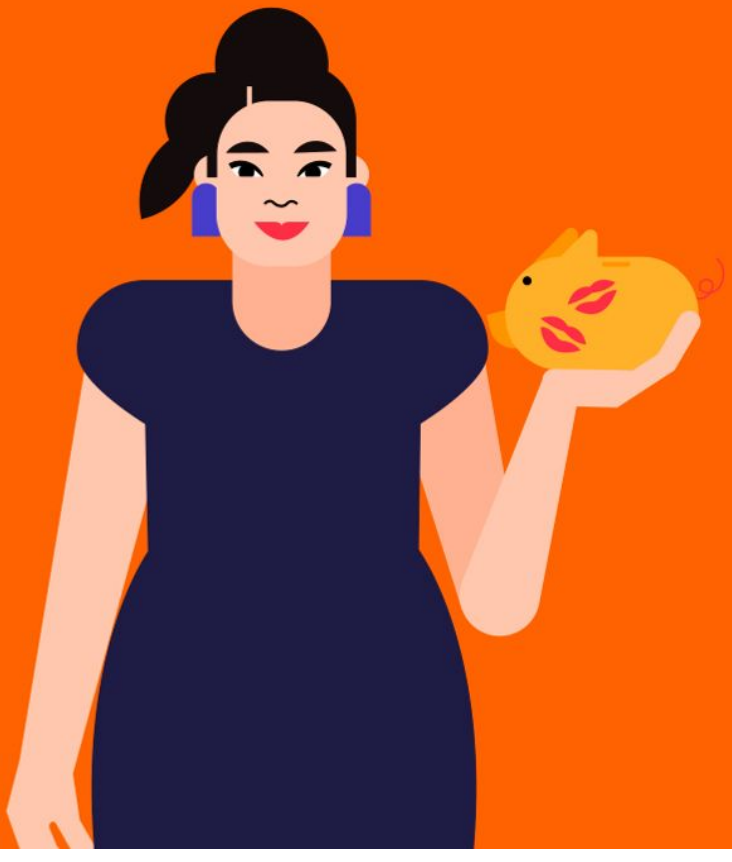
ING BANK NV se encuentra adherido al Sistema de Garantía de Depósitos Holandés con una garantía de hasta 100.000 euros por depositante. Consulta más información en ing.es

Que te den **10 € para gastar** es una fantasía. ING lo hace realidad.

Abre la **Cuenta NoCuenta** con el código **WUOLAH10**, haz tu primer pago y llévate 10 €.

Quiero el cash

Consulta condiciones **aquí**



do your thing

Paso 2: Programar el Script con Crontab

Ahora que tienes el script listo, necesitas programarlo para que se ejecute cada minuto.

1. **Abre el archivo crontab para editar**:

```
crontab -e
```

2. **Agrega la siguiente línea al final del archivo** para programar la ejecución del script cada minuto:

```
* * * * * /ruta/a/mi_script.sh
```

Esto le dice a `cron` que ejecute `mi_script.sh` cada minuto, sin importar el día, mes o día de la semana.

3. **Guarda y cierra el editor**. Al igual que antes, si usas `nano`, presiona `CTRL + X`, luego `Y` y `Enter`.

Paso 3: Verificar la Configuración

Para asegurarte de que tu tarea está programada correctamente, puedes listar las tareas programadas con:

```
crontab -l
```

Deberías ver la línea que acabas de agregar.

Paso 4: Comprobar la Ejecución

1. **Espera un minuto** para que se ejecute el script.

2. **Verifica el archivo de log** para ver si se registró la ejecución:

```
cat /ruta/a/log.txt
```

Deberías ver líneas como esta:

```
El script se ejecutó a las: Thu Oct 21 14:01:00 UTC 2024
El script se ejecutó a las: Thu Oct 21 14:02:00 UTC 2024
...
```

```
```

Notas Finales

- **Salida de Errores**: Si deseas redirigir los errores al mismo archivo de log, puedes modificar la línea en `crontab` de la siguiente manera:

```bash
* * * * * /ruta/a/mi_script.sh >> /ruta/a/log.txt 2>&1
```

- **Ver Logs de Cron**: Los logs de cron generalmente se pueden encontrar en `/var/log/syslog` (en distribuciones basadas en Debian como Ubuntu). Puedes verificar los logs con:

```bash
grep CRON /var/log/syslog
```

Si tienes alguna pregunta o necesitas más ayuda, ¡házmelo saber!