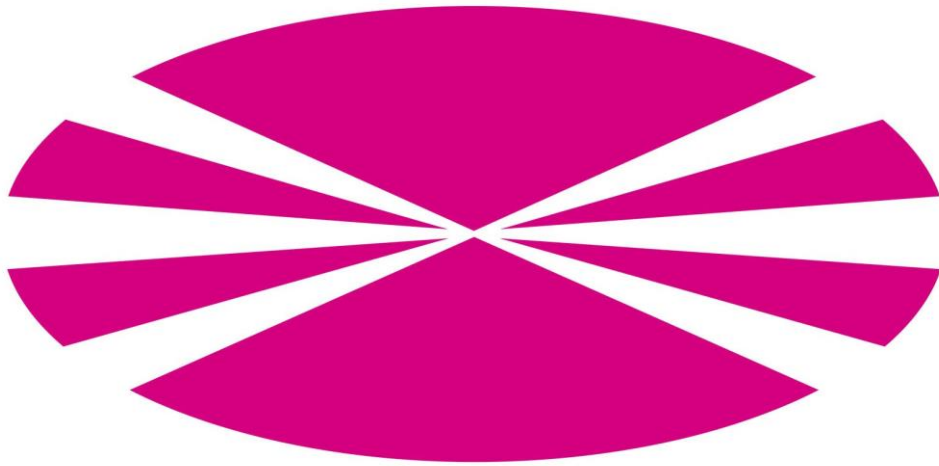


MEMORIA PRÁCTICA 8
ADMINISTRACIÓN DE SISTEMAS OPERATIVOS

Maseda Dorado, Tomé



UNIVERSIDADE DA CORUÑA

Facultade de Informática

Universidade da Coruña

A Coruña, Spain

1. Crear un shell script que crea copias de seguridad de los directorios de los usuarios

Para crear el script usé un editor en mi máquina local (por comodidad), el script creado es el siguiente:

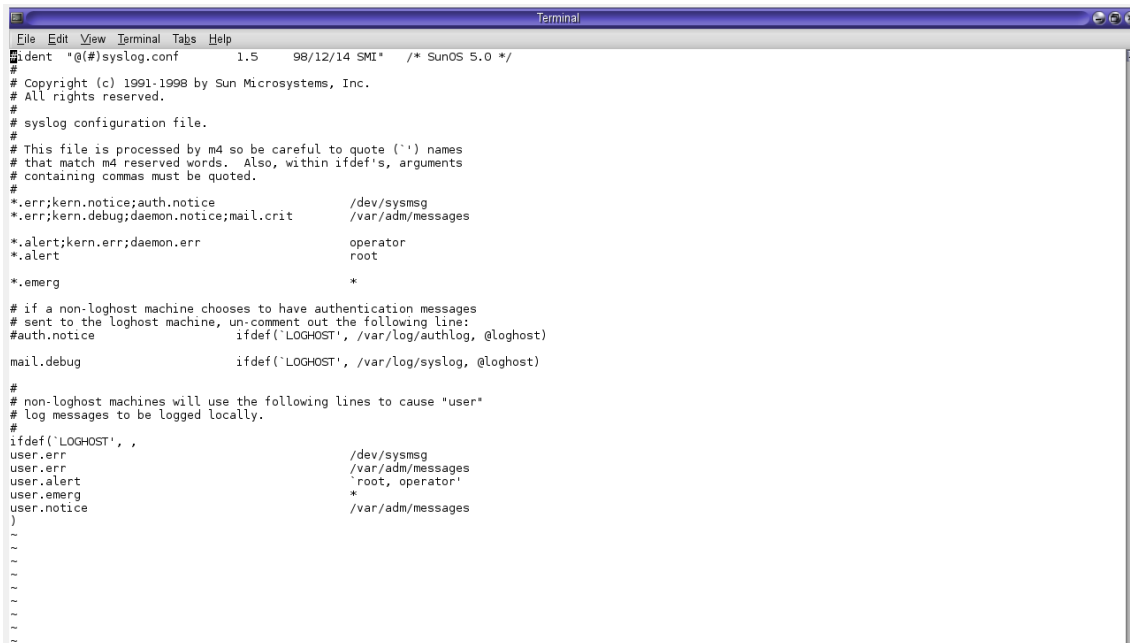
E:\1. GEI\GEI 3\2C\ASO\PRÁCTICAS\Práctica 8\backup_script.sh - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

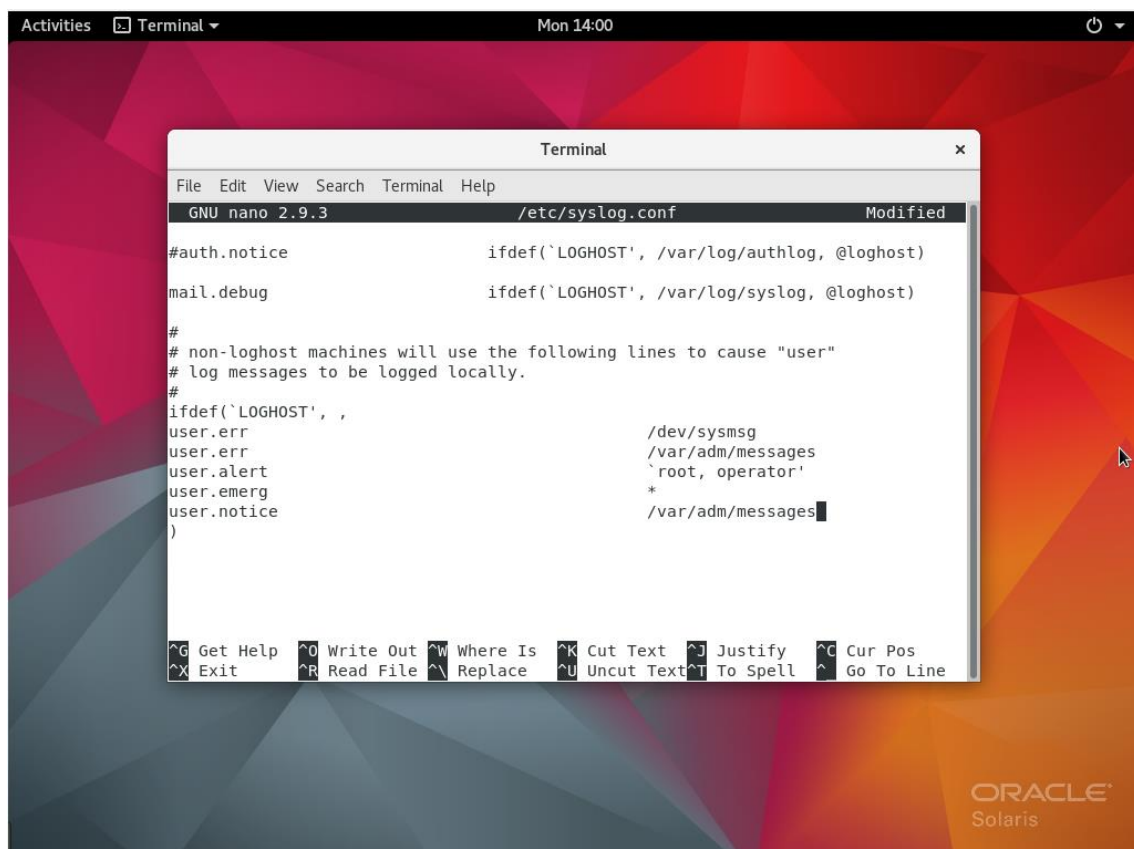
```
backup_script.sh
1  #!/bin/sh
2
3  BACKUP_DIR="/var/backups"
4  BACKUP_LIM=10
5  BACKUP_LOG=YES
6  #DATE=`date +%d-%m-%y_%H-%M-%S`
7  FILE_NAME="$(whoami).tar.gz"
8  SIZE=`du -ks $HOME | cut -f 1`
9  ERROR="/tmp/error.txt"
10
11 # Si el directorio no supera el límite n se hace un backup
12 if [ $DIR_SIZE -le $BACKUP_LIM ];
13 then
14     echo "$HOME does not reach the minimum size: $SIZE kB (< 10 kB)";
15     if [ $BACKUP_LOG = "YES" ]; # Si la variable de log está activa se reporta el log, si no termina la ejecución
16     then
17         logger -p user.notice -t backup_script "$HOME does not reach the minimum size: $SIZE kB (< 10 kB)";
18     fi
19     exit 0;
20 fi
21
22 # Creamos un archivo temporal para redirigir la salida de error del backup
23 touch $ERROR;
24 # Creamos el directorio si aún no existe
25 mkdir -p "$BACKUP_DIR";
26
27 # Hacemos backup del directorio en un archivo comprimido
28 # redirigiendo el error a un archivo
29 tar cf "$BACKUP_DIR/$FILE_NAME" "$HOME" 2> $ERROR;
30 chmod 700 "$BACKUP_DIR/$FILE_NAME"; # La copia solo será accesible por el usuario al que corresponde
31 # Reportamos el log dependiendo de si es un error o el programa
32 # fue ejecutado con éxito
33 if [ $? -eq 0 ];
34 then
35     echo "Backup from $HOME done";
36     if [ $BACKUP_LOG = "YES" ]; # Si la variable de log está activa se reporta el log, si no termina la ejecución
37     then
38         logger -p user.notice -t backup_script "Backup from $HOME done";
39     fi
40 else
41     echo "Error while doing backup. See your system logs for further information.";
42     if [ $BACKUP_LOG = "YES" ]; # Si la variable de log está activa se reporta el log, si no termina la ejecución
43     then
44         logger -p user.notice -t backup_script -f $ERROR;
45     fi
46 fi
47 rm $ERROR;
48 exit 0;
```

Aclaración: Cuando uso logger para reportar el resultado de la ejecución del script a los logs del sistema específico la prioridad user.notice, como ponía en el enunciado, sin embargo, esto es un poco redundante porque la prioridad por defecto de logger ya es user.notice. Especifiqué esa prioridad de todas formas porque no estaba seguro si la prioridad por defecto dependía del sistema operativo y el script debe ser independiente del sistema operativo.

En Solaris 10 y Solaris 11 tuve que editar la configuración de syslog (fichero /etc/syslog.conf), para que guardase en algún fichero de log los logs con prioridad user.notice, si no simplemente los descartaba.

A terminal window titled "Terminal" showing the contents of the /etc/syslog.conf file. The file contains configuration for syslog, including log destinations for various message priorities and conditional compilation for loghost machines.

```
ident *(@(#)syslog.conf 1.5 98/12/14 SMI* /* SunOS 5.0 */
#
# Copyright (c) 1991-1998 by Sun Microsystems, Inc.
# All rights reserved.
#
# syslog configuration file.
#
# This file is processed by m4 so be careful to quote (') names
# that match m4 reserved words. Also, within ifdef's, arguments
# containing commas must be quoted.
#
*.err;kern.notice;auth.notice /dev/sysmsg
*.err;kern.debug;daemon.notice;mail.crit /var/adm/messages
*.alert;kern.err;daemon.err operator
*.alert root
*.emerg *
#
# if a non-loghost machine chooses to have authentication messages
# sent to the loghost machine, un-comment out the following line:
#auth.notice ifdef('LOGHOST', /var/log/authlog, @loghost)
mail.debug ifdef('LOGHOST', /var/log/syslog, @loghost)
#
# non-loghost machines will use the following lines to cause "user"
# log messages to be logged locally.
#
ifdef('LOGHOST', ,
user.err /dev/sysmsg
user.err /var/adm/messages
user.alert 'root, operator'
user.emerg *
user.notice /var/adm/messages
)
~
~
~
~
~
~
~
```

A terminal window titled "Terminal" showing the GNU nano 2.9.3 editor editing the /etc/syslog.conf file. The editor interface includes a menu bar, a status bar, and a list of keyboard shortcuts at the bottom. The file content is the same as in the previous image, but with some modifications to the user-related log destinations.

```
GNU nano 2.9.3 /etc/syslog.conf Modified
#auth.notice ifdef('LOGHOST', /var/log/authlog, @loghost)
mail.debug ifdef('LOGHOST', /var/log/syslog, @loghost)
#
# non-loghost machines will use the following lines to cause "user"
# log messages to be logged locally.
#
ifdef('LOGHOST', ,
user.err /dev/sysmsg
user.err /var/adm/messages
user.alert 'root, operator'
user.emerg *
user.notice /var/adm/messages
)
~
~
~
~
~
~
~
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

ORACLE
Solaris

2. Hacer que las copias de seguridad se realicen 1 vez al día.

En todos los S.O. fui haciendo lo siguiente.

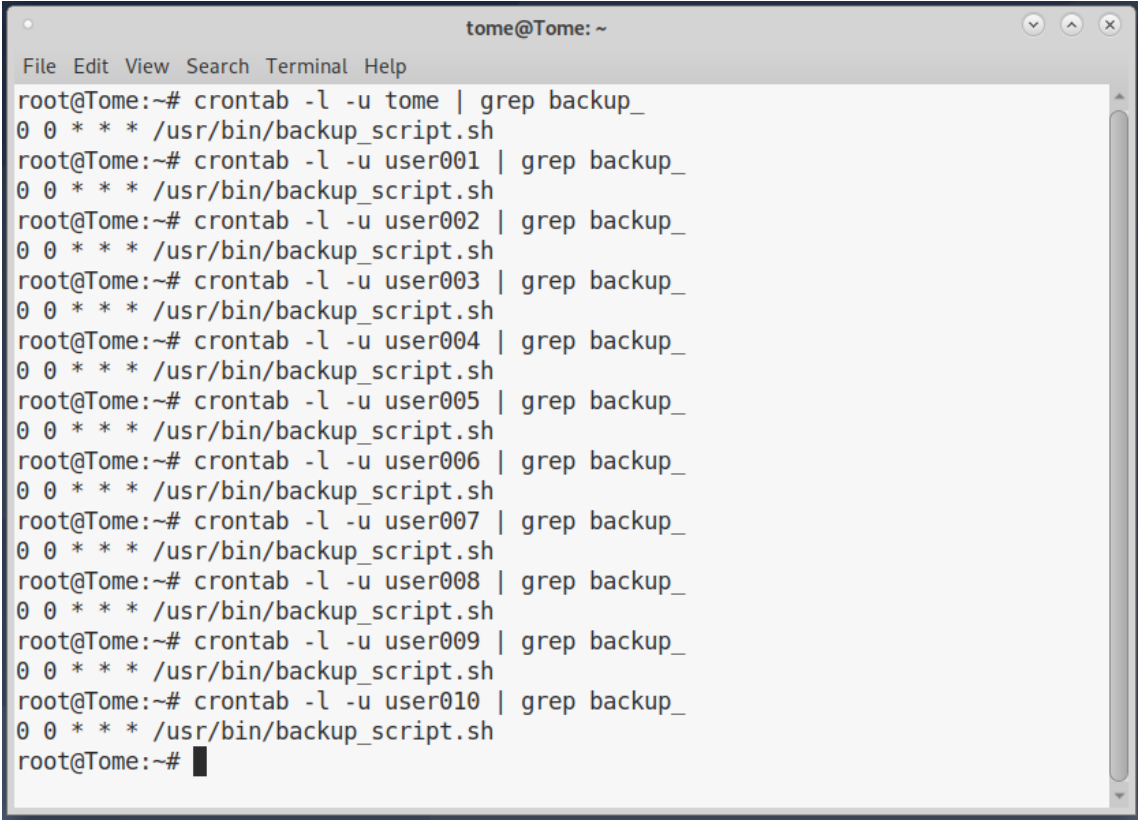
Para cada usuario del sistema ejecuto `crontab -u <usuario> -e` y añado la siguiente línea dentro del editor:

```
0 0 * * * /usr/bin/backup_script.sh
```

Con esto indico que se ejecute el script `/usr/bin/backup_script.sh` todos los días del mes (independientemente del día de la semana) de cualquier mes a la 00:00.

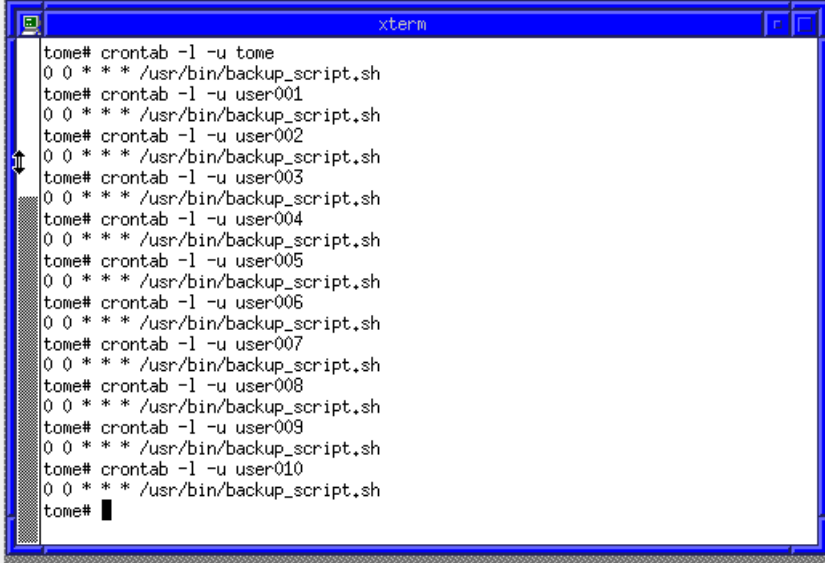
A continuación muestro el crontab de cada usuario con `crontab -l -u <usuario>`.

Debian Linux



```
tome@Tome: ~  
File Edit View Search Terminal Help  
root@Tome:~# crontab -l -u tome | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user001 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user002 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user003 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user004 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user005 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user006 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user007 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user008 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user009 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~# crontab -l -u user010 | grep backup_  
0 0 * * * /usr/bin/backup_script.sh  
root@Tome:~#
```

OpenBSD



```
tome# crontab -l -u tome
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user001
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user002
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user003
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user004
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user005
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user006
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user007
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user008
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user009
0 0 * * * /usr/bin/backup_script.sh
tome# crontab -l -u user010
0 0 * * * /usr/bin/backup_script.sh
tome#
```

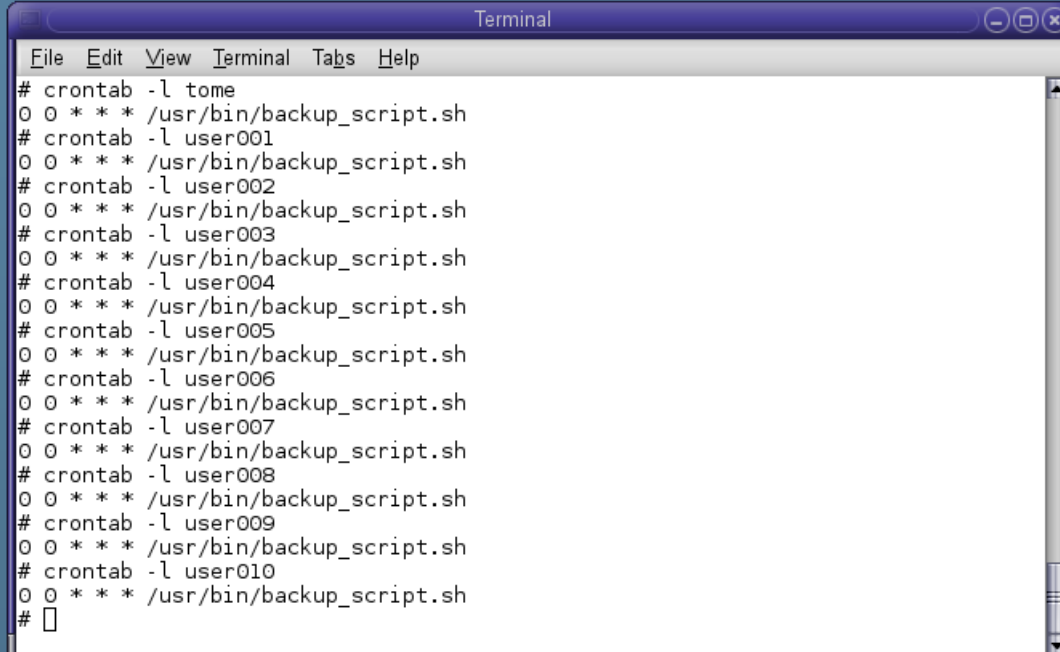
Solaris 10

En Solaris tuve que establecer el editor del sistema previamente (variable de entorno EDITOR).

```
# EDITOR=vi
```

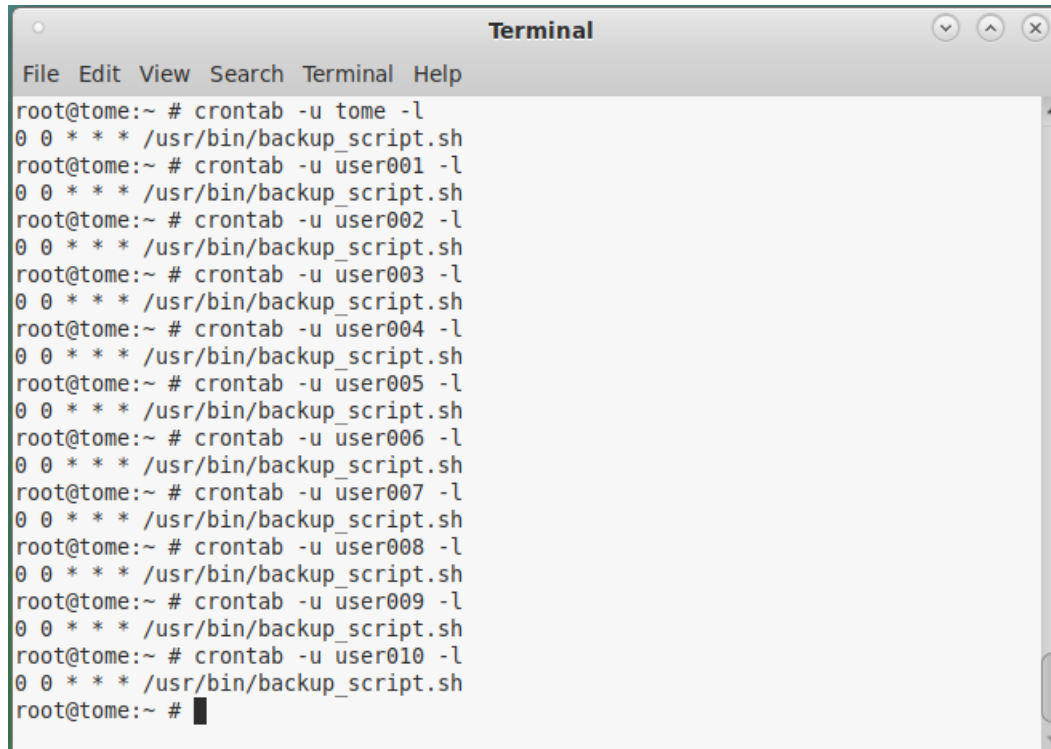
```
# export EDITOR
```

Para editar el crontab de cada usuario en este caso es con `crontab -e <usuario>`.



```
# crontab -l tome
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user001
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user002
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user003
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user004
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user005
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user006
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user007
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user008
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user009
0 0 * * * /usr/bin/backup_script.sh
# crontab -l user010
0 0 * * * /usr/bin/backup_script.sh
#
```

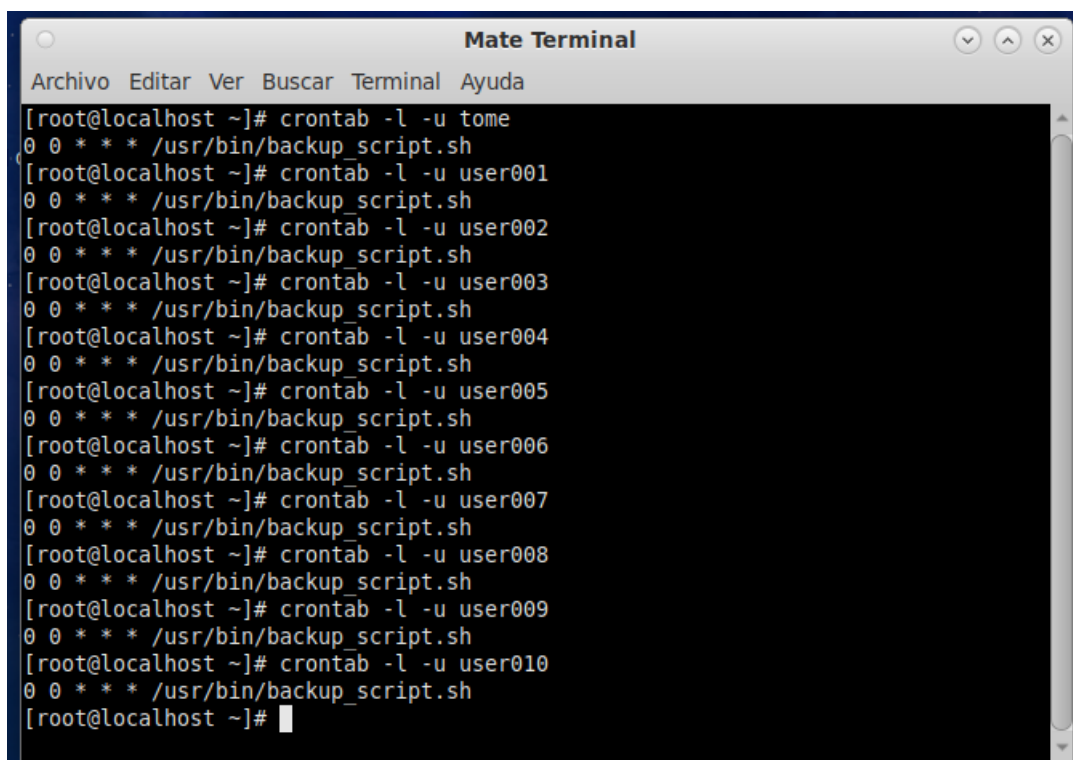
FreeBSD



A terminal window titled "Terminal" with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows a series of commands to set up crontab for a user named 'tome' and ten other users (user001 to user010). Each user's crontab is configured to run the script /usr/bin/backup_script.sh at 00:00 every day.

```
root@tome:~ # crontab -u tome -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user001 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user002 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user003 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user004 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user005 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user006 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user007 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user008 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user009 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ # crontab -u user010 -l
0 0 * * * /usr/bin/backup_script.sh
root@tome:~ #
```

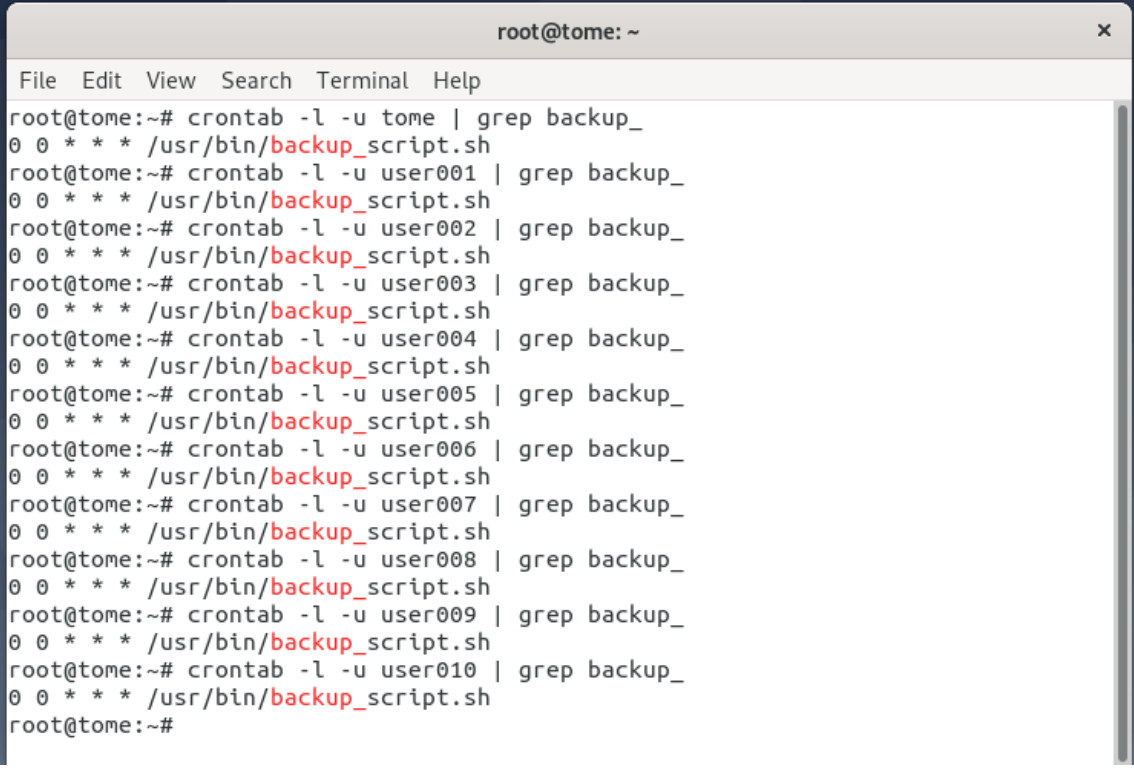
Linux Fedora



A terminal window titled "Mate Terminal" with a menu bar (Archivo, Editar, Ver, Buscar, Terminal, Ayuda). The terminal shows a series of commands to set up crontab for a user named 'tome' and ten other users (user001 to user010). Each user's crontab is configured to run the script /usr/bin/backup_script.sh at 00:00 every day.

```
[root@localhost ~]# crontab -l -u tome
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user001
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user002
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user003
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user004
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user005
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user006
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user007
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user008
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user009
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]# crontab -l -u user010
0 0 * * * /usr/bin/backup_script.sh
[root@localhost ~]#
```

Ubuntu Server

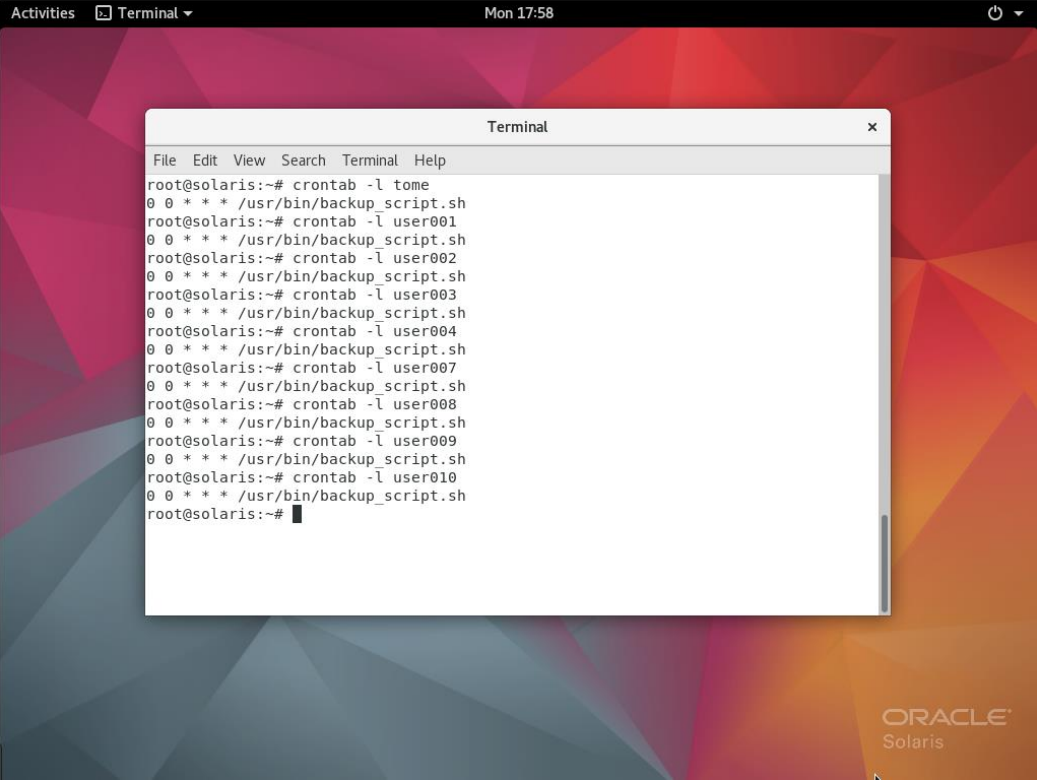


```
root@tome: ~
File Edit View Search Terminal Help
root@tome:~# crontab -l -u tome | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user001 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user002 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user003 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user004 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user005 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user006 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user007 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user008 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user009 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~# crontab -l -u user010 | grep backup_
0 0 * * * /usr/bin/backup_script.sh
root@tome:~#
```

Solaris 11

Para editar el crontab de cada usuario en este caso es con `crontab -e <usuario>`.

(No edite los usuarios user005 y user006 porque no me lo permitía al haber expirado las cuentas -> Práctica 4)



```
Activities Terminal Mon 17:58
Terminal
File Edit View Search Terminal Help
root@solaris:~# crontab -l tome
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user001
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user002
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user003
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user004
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user007
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user008
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user009
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~# crontab -l user010
0 0 * * * /usr/bin/backup_script.sh
root@solaris:~#
```

