
# Seguridad en Redes

## Introducción a LINUX / UNIX

# Seguridad en UNIX

- Introducción
- System V vs Berkeley
- Kernel y Shells
- Como obtener Ayuda
- File System
- Administración de Usuarios
- Procesos
- Procedimiento de logueo
- Permisos
- Comandos Básicos y Pipes
- Secure Shell
- Scripting

# Seguridad en UNIX

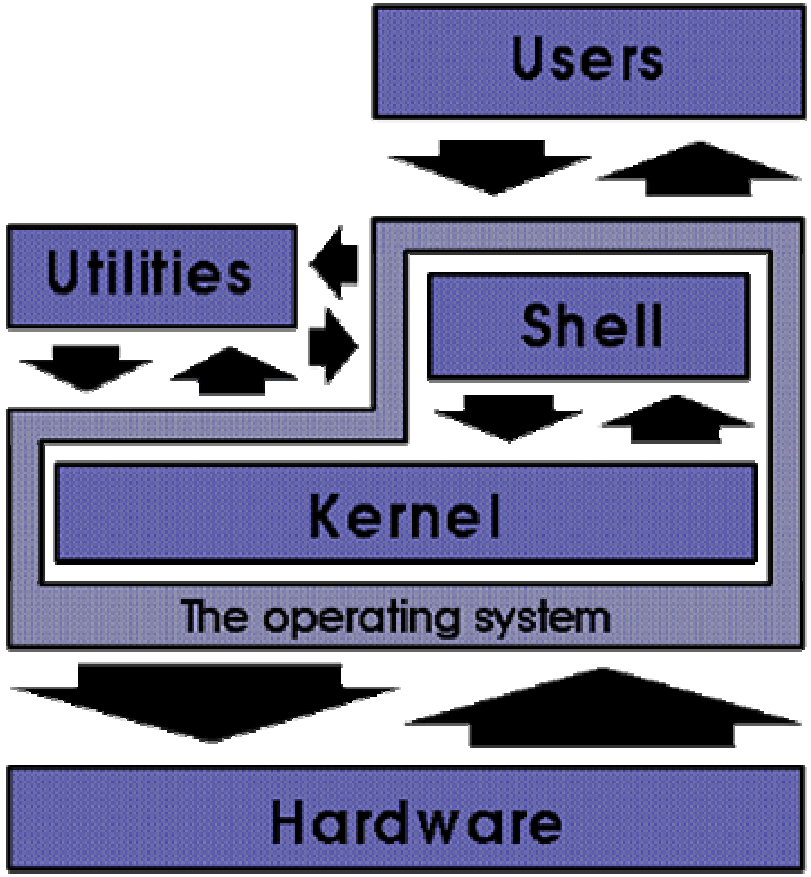
- Fue desarrollado por AT&T a fines de los 60s
- La primera versión se llamo UNICS y fue escrita por Dennis Ritchie
- Programado íntegramente en C
- Recibe distintos nombres hoy en día
  - AIX
  - HP-UX
  - Solaris
  - Linux

# System V vs Berkeley

- Existes basicamente dos lineas de Unix:
  - System V: Desarrollada por AT&T
  - BSD: Desarrollado por la Universidad de Berkeley

- Las interrelaciones entre estas familias son las siguientes, aproximadamente en orden cronológico:
  - La familia BSD surge del licenciamiento del UNIX original de AT&T.
  - Xenix también surge por licenciamiento del UNIX original de AT&T, aunque aún no era propiedad de SCO.
  - AIX surge por licenciamiento de UNIX System III, pero también incorpora elementos de BSD.
  - La familia original AT&T incorpora ilegalmente propiedad intelectual de BSD en UNIX System III r3.
  - La familia AIX vuelve a incorporar propiedad intelectual de la familia AT&T, esta vez procedente de UNIX System V.
  - Linux incorpora propiedad intelectual de BSD, gracias a que éste también se libera con una licencia de código abierto denominada Open-source BSD.
  - Aunque no está demostrado, se cree que Linux incorpora propiedad intelectual procedente de AIX, gracias a la colaboración de IBM en la versión 2.4 de su núcleo (kernel).

# Kernel y Shell



# Kernel y Shell

- Es el sistema operativo en si
- Es el encargado de administrar los recursos, procesos y usuarios
- Administra las interfaces de los recursos de E/S
- Maneja los discos y las memorias

# Shell

- Toda la comunicación que tiene el UNIX es a través del shell
- Existen distintos shells (Bash, Bourne, Korn)
- Es una línea de comando.
- Existen varios comandos para interactuar con el sistema

# Manuales de Linux

- man : permite ver el manual de los comandos.
  - EJ: man man
  - Ej2 man cp

- apropos (para ansi: man –k):
  - Busca un string en los titulos de los manuales

- Los manuales tienen distintos capítulos:
  - man –a login  (muestra todos los capítulos)
  - man 1 login (muestra el capitulo 1 del comando login)

# man man

```
MAN(1)                    FreeBSD General Commands Manual                    MAN(1)

NAME
     man -- format and display the on-line manual pages

SYNOPSIS
     man [-adfhkotw] [-m arch[:machine]] [-p string] [-M path] [-P pager]
         [-S list] [section] name ...

DESCRIPTION
     The man utility formats and displays the on-line manual pages.  This ver-
     sion knows about the MANPATH and PAGER environment variables, so you can
     have your own set(s) of personal man pages and choose whatever program
     you like to display the formatted pages.  If section is specified, man
     only looks in that section of the manual.  You may also specify the orde
     to search the sections for entries and which preprocessors to run on the
     source files via command line options or environment variables.  If
     enabled by the system administrator, formatted man pages will also be
     compressed with the ``/usr/bin/gzip -c'' command to save space.

     The options are as follows:

     -M path   Specify an alternate manpath.  By default, man uses manpath(1)
               (which is built into the man binary) to determine the path to
               search.  This option overrides the MANPATH environment vari-
               able.
```

# File System

- Todos son Archivos
  - Archivos
  - Directorios
  - Dispositivos

- Estructura de directorios
  - `/bin  (Binarios del SO)`
  - `/etc  (Configuraciones del SO)`
  - `/dev  (Referencias a dispositivos)`
  - `/home (Directorios de los usuarios)`
  - `/usr  (Binarios de aplicaciones)`
  - `/tmp  (Directorio Temporal del Sistema)`
  - `/var  (Archivos de Datos de las aplicaciones, spool, log, bases, etc)`
  - `/opt  (Binarios de aplicaciones)`

# Comandos para manejo FS

- pwd – muestra el directorio actual
- ls – equivalente al dir
- cd - cambia de directorio
- cp – equivalente a copy
- mv - renombra
- rm – delete file
- mkdir – crea un directorio
- rmdir – borra un directorio
- grep – global regular expresión print
- find - busca un archivo por el nombre u otros parametros
- ln – crea un link al archivo

- Busqueda
  - find –name fstab /

# link

- Link Simbolico
  - Es un “shorcut” del archivo

```
ln -s origen destino
```

- Hard Link
  - Usando este tipo de links en realidad se le asigna al archivo un nuevo nombre que puede estar (o no) en otro directorio. Es indistingible del primero.
  - Si se borra un archivo que tiene links recién se borrara al borrar el ultimo link

```
ln origen destino
```

# Administración de Usuario

- Archivo de usuarios /etc/passwd

  username:password:uid:gid:gcos-field:homedir:login-shell

- Archivo de contraseñas /etc/shadow

  username:password:lastchg: min:max:warn:inactive: expire:flag

- Archivo de grupos /etc/group

  group:passwd:gid:members

# Administración de Usuario

- **groupadd**

Agrega un usuario SO → /etc/group

***groupadd grupo***

- **useradd, usermod, userdel**

Agrega un usuario al SO → /etc/passwd

***useradd –g primgrp –G grp1,grp2,… -c comentario –s shell***

- **passwd**

Cambia el password y otros de un usuario → /etc/shadow

***passwd user***

```
> whoami
admin
> who
admin            ttyp3    Apr 21 12:02 (200.127.33.233)
> id
uid=1001(admin) gid=1001(admin) groups=1001(admin)
>
```

# Procesos

- Un proceso es un programa en ejecución
- Todos los procesos tiene un proceso padre. El padre de todos es el proceso init.
  - PID numero de proceso
  - PPID numero de proceso padre
  - PUID usuario con que esta corriendo
- "ps" Para ver procesos
  - ps –fx
  - ps –fu usuario
  - ps -ef
- fork / exec

# top:

- Muestra un listado de los procesos ordenados con algun criterio

```
last pid: 37029;  load averages:  0.60,  0.48,  0.60    up 6+08:31:40
12:02:29
29 processes:  1 running, 28 sleeping
CPU:  6.8% user,  0.0% nice,  2.7% system,  0.1% interrupt, 90.4% idle
Mem: 8884M Active, 4920M Inact, 1023M Wired, 268M Cache, 399M Buf, 297M Free
Swap: 32G Total, 11M Used, 32G Free

  PID USERNAME  THR PRI NICE   SIZE    RES STATE  C   TIME   WCPU COMMAND
 2987 root        1   4    0 52628K 21928K select 5   3:25  0.00% python2.5
 2927 bind       11   4    0 30108K 18956K select 0   1:45  0.00% named
 2914 root        1   4    0  5688K  1072K select 7   1:45  0.00% syslogd
 3040 root        1   4    0 15044K  2132K select 5   1:30  0.00% sshd
 3032 root        1  44    0   121M 11776K select 7   1:08  0.00% httpd
 4460 www         1   4    0   136M 19936K accept 6   0:08  0.00% httpd
 4462 www         1   4    0   137M 30320K accept 0   0:06  0.00% httpd
 7972 www         1   4    0   126M 14572K accept 4   0:04  0.00% httpd
 3047 root        1   8    0  6744K   976K nanslp 5   0:03  0.00% cron
36986 root        1   4    0 27008K  3300K sbwait 0   0:00  0.00% sshd
37010 admin       1  20    0 10100K  2360K pause  2   0:00  0.00% tcsh
```

# Procesos (cont)

- **Foreground / Background**

```
ls -R / > pepe.txt &
```

- **Ctrl-Z:** Pasa un proceso en foreground
- **jobs:** Muestra los procesos corriendo en background

- **Señales:** Es una manera de pasarle mensajes a un proceso que esta corriendo
- Para reciclar un proceso

```
kill HUP PID
```

- Para solicitar gentilmente que se muera

```
kill PID o kill -15 PID
```

- Para sacarlo de memora

```
kill -9 PID
```

# Procesos (cont)

```
> sleep 800 &
        [1] 38803
> sleep 400 &
        [2] 38817
> jobs
        [1]  + Running                 sleep 800
        [2]  - Running                 sleep 400
> kill %1
> jobs
        [1]    Terminated              sleep 800
        [2]  + Running                 sleep 400
> jobs
        [2]  + Running                 sleep 400
> fg %2
        sleep 400
^Z
        Suspended
> bg %2
        [2]    sleep 400 &
> ps
          PID  TT  STAT      TIME COMMAND
        37010  p3  SsJ    0:00.02 -tcsh (tcsh)
        38817  p3  IJ     0:00.00 sleep 400
        39105  p3  R+J    0:00.00 ps
```

# Variables de Ambiente

- **env**: Muestra las variables de ambiente
  - Definir una variable es dependiente del shell:
  - `NETID=abc123; export NETID` (bourne shell)
  - `setenv NETID abc123` (c-shell)
- Para usar una variable en un comando:

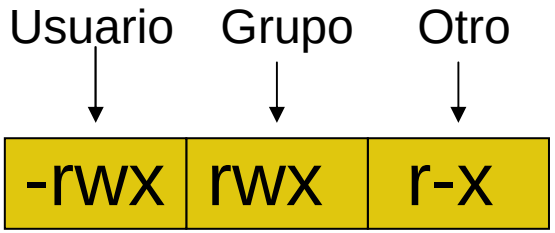
  `echo $NETID`
- Las variables de ambiente se usan para configurar programas y para configurar el funcionamiento del Sistema

  - $PATH define la ruta desde la que se ejecutan programas.
  - $TERM emulador de Terminal usado.
  - $HTTP_PROXY Algunos programas de linea de comando toman el proxy de esta variable de ambiente

# Permisos de Archivo

- Existen tres tipos de niveles de acceso
  - Usuario
  - Grupo
  - Otros

- Existen tres tipos de accesos
  - Lectura
  - Escritura
  - Ejecución




**-rwxrwxr-x 1 user01 staff 320 Jan 22  docfile**

# Permisos

- Cambio de Permisos → chmod
  - +: agrega permiso
  - -: elimina permiso
  - =: fija un permiso
  - u: aplica al usuario
  - g: aplica al grupo
  - o: aplica a otros
  - a: aplica a todos
- En números
  - 4: lectura  2: write  1: ejecución
- Cambio de Dueño → chown
- Cambio de Grupo → chgrp

```
chmod 751 archivo
chmod u=rwx g=rx o=x archivo
chown user:group archivo
chgrp group archivo
```

# Permisos de un Proceso

- Corre con un usuario valido del unix
- Por lo tanto tiene acceso a los recursos a los que el usuario tiene acceso.
- Cuando un usuario ejecuta un proceso, el mismo corre con los permisos del usuario

# Procesos que suceden durante un login

- Proceso getty

  /sbin/mingetty tty1

- Pasos de login
  - Se ingresa el usuario
  - Se ingresa el password
  - Se valida el usuario y la contraseña
  - El getty toma del passwd el shell, UID, GUD y el Home Directory ($HOME) del usuario.
    - Inicia el shell
    - Ejecuta el archivo de /etc/profile
    - Ejecuta el archivo $HOME/.profile (.bashrc u otros según shell)
    - Muestra el prompt

# Comando su

- Permite entrar con los permisos de otro usuario.

```
su usuario
```

- Ejecuta el ambiente del usuario destino

```
su - usuario
```

# Permiso "SUID bit"

- Permite que un proceso ejecute con las credenciales del dueño del proceso en lugar de las credenciales de quien lo ejecuta.

- Ejemplo el comando passwd necesita el bit "S" para poder modificar el archivo de claves:

```
[root@desa hpg123]# ls -al /usr/bin/passwd
-r-s--x--x  1 root root 92757 Feb 15  2004 /usr/bin/passwd
```

# Combinación de comandos con pipe

- Pipe "|" es un método de comunicación de procesos
  - El output de un proceso es usado como input por el otro
  - Ej: Para buscar los datos de un usuario en el passwd

```
cat /etc/passwd | grep user1
```

- Ver si hay un proceso corriendo

```
ps -ef | grep vi
```

- Espacio de disco ocupado por usuarios

```
du -sk /home/* | sort -nr | head -10 > reporte.txt
```

# Algo de Redes

- Configuración de la interface de red
  - ifconfig eth0 10.0.0.1 netmask 255.255.255.0
- Archivo de hosts
  - /etc/hosts
- Nombre del host
  - hostname nombre del host
  - hostname –i (solo en linux)
  - hostname –f (solo en linux)

- **ping / traceroute:** Ambos comando sirven para verificar un ruta hacia un host
- **netstat**: Sirve para obtener información de las conexiones TCP/IP
- **route**: sirve para configurar la tabla de ruteo
- **iptables**: sirve para controlar el flujo y conexiones de red

# Secure Shell

- Permite loguearse a un sistemas de manera segura
- Se intercambia un par de claves RSA
- Se negocia una clave compartida
- Se encripta todo el trafico con 3DES
- Los archivos importantes

  `$HOME/.ssh/id_rsa.pub`

  `$HOME/.ssh/id_rsa`

  `$HOME/.ssh/authorized_keys`

```
ssh user@host
scp origen user@host:destino
```

# Scripting en shell

- Permite combinar comandos para agrupar secuencias de comandos.
- Es un lenguaje de programación que tiene por ejemplo “if, for, while, read, etc”
- Ejemplo

```
touch /var/log/mysqld.log
chown mysql:mysql /var/log/mysqld.log
chmod 0640 /var/log/mysqld.log
if [ ! -d $datadir/mysql ] ; then
    echo "Inicializando MySQL database: "
    /usr/bin/inicia_base
    ret=$?
    chown -R mysql:mysql $datadir
    if [ $ret -ne 0 ] ; then
        exit $ret;
    fi
else
  echo "falta el directorio del Mysql
fi
```