Seguridad de los sistemas de ficheros

Ampliación de Sistemas Operativos

Trabajo realizado por:

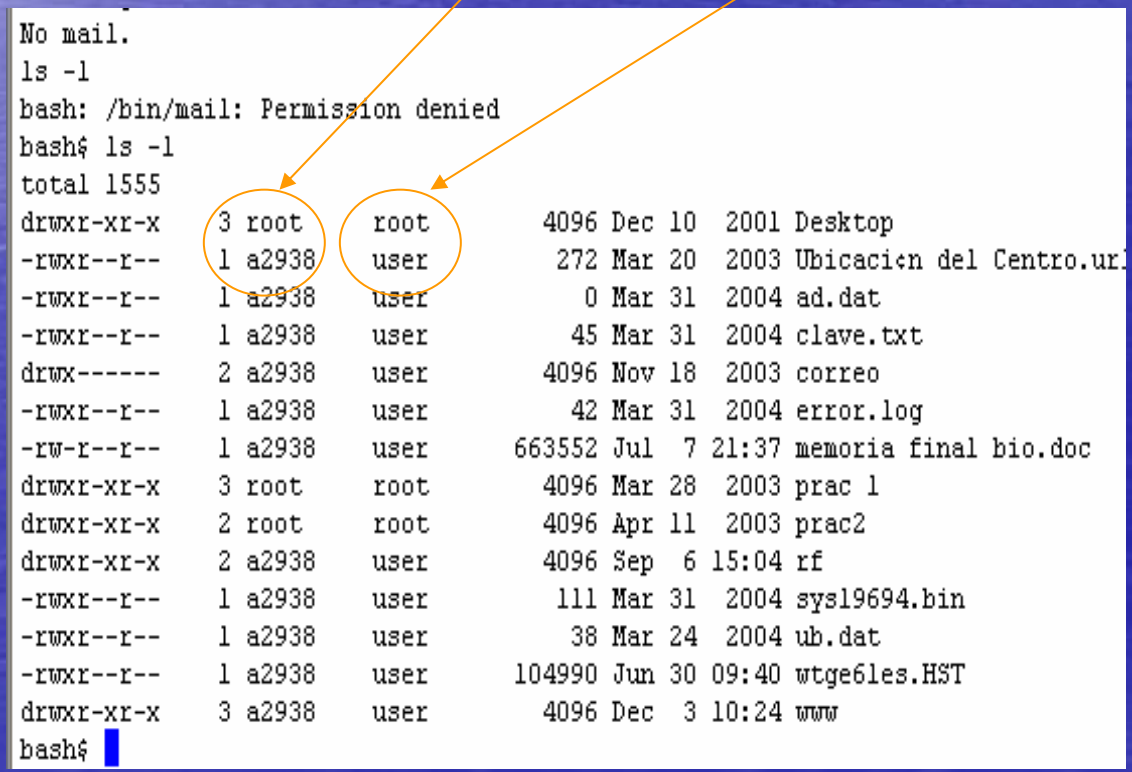
–Jose Yeray Suárez Perdomo

Introducción (I)

- La seguridad en el sistema de ficheros es la más conocida por todos.
  - Muchas de los conceptos y herramientas son manejados con frecuencia.
- Cada fichero posee:
  - Un grupo
  - Un usuario

Introducción (II)

- Se puede visualizar el usuario y el grupo utilizando “ls-l”:



Derechos de acceso (I)

- Los accesos a los ficheros vienen descritos por en bloque de símbolos

Derechos de acceso (II)

- El Primer carácter de la cadena,
  - D -> Directorio
  - B -> Dispositivo de bloque
  - E -> Dispositivo de tipo carácter
  - “-” -> Tipo normal
- Para cada uno de los 3 caracteres de cada permiso,
  - 1º -> Lectura / Listar
  - 2º -> Escritura / Crear o Borrar ficheros
  - 3º -> Ejecución / Ejecutar ficheros en el directorio

Cambiar permisos (I)

- Herramienta **chmod**

  chmod [-R] [ugoa]{ + | - | = }[rwxXstl] file

- Se utiliza para cambiar cualquier permiso de los ficheros a cualquier usuario del mismo.

- U : usuario propietario
- G : grupo propietario
- O : resto de los usuarios
- A : Todos

+ : Añadir
- : Eliminar
= : Establecer

Cambiar permisos (II)

- Utilización “chmod” en modo numérico.

  Chmod NNN
- Cada N se refiere a usuario, grupo y resto.
- Se usa nomenclatura octal (0-7)
- Cuando un permiso está activa se codifica con 1, en caso contrario con 0.

  d r-- -w- --x

  100 010 001
- La orden sería :-> chmod 421

Umask y los permisos por defecto

- Existe una serie de permisos por defecto para las nuevos ficheros.
- Se pueden modificar utilizando “umask”
- Umask utiliza modo numérico inverso al chmod
  - 0 con el “umask” equivale a 1 en chmod
  - Ej: chmod 421 => umask 356
- Los permisos por defecto se almacenan en *etc/profile* y para nuevos usuarios en */etc/skel/.bashrc*
  - Ejemplo: echo ‘ umask 177 ’ >>/etc/profile

Sticky bit

- Es un caso especial que se aplica sobre directorios.
- Modifica los permisos de los otros usuarios
  - Pueden crear y borrar sus propios ficheros sin modificar los de otros usuarios.
- Ejemplo:

  Chmod o+t directorio

  En el bit de ejecución del resto aparece una “t” en lugar de “x”

SUID/SGID (I)

- Cada Proceso tiene un usuario y un grupo definido en su ejecución.
- Dependiendo de ello tiene unos permisos.
- Utilizando SUID/SGID los usuarios tienen los permisos del propietario del fichero
- Se utiliza para dar más privilegios
- Puede llegar a ser una ventaja o un problema

SUID/SGID(II)

- Se puede modificar con el “chmod” cambiando la “x” por una “s”.
- Es más usual en modo numérico:
  - Se utilizan 4 dígitos en lugar de 3
  - Los 3 últimos son los permisos normales
  - El 1º modifica SUID/SGID
    - 4 -> SUID
    - 2 -> SGID
    - 6 ->Ambos

Ejemplo

- *Almacenar en un archivo todos aquellos ficheros que posean dichos bits activados. Mirar periódicamente si existen otros ficheros y comprobar si se encuentran en la lista, en caso negativo, mostrar un mensaje que avise al administrador:*
  - 1.- Redireccionar la salida al fichero que contendrá el listado
  - 2.- Encontrar todos aquellos ficheros con el “find”:
    - “find / -perm -6000”
  - 3.- Periódicamente ejecutar un script que utilice el “grep”(busca ristras en ficheros) para comprobar si el fichero está controlado.
    - “find / -perm -6000 –exec grep fichero”
  - 4.- Devuelve los ficheros que si están controlados, si existe alguno que no se encuentre en este listado, se manda al administrador un mensaje de aviso para cada archivo encontrado.

chattr

- Permite también modificar los permisos
- Sólo es accesible para los superusuarios
- Chattr [+|-|=] [a|i] fichero
  - a: permite sólo añadir contenido al fichero
  - i: no permite modificaciones

Herramientas de seguridad de sistema de ficheros

- Mecanismos ACL (access control list):
  - Permiten modificar permisos que no se encuentran disponibles en el sistema de ficheros tradicional de Unix
- Herramientas para el borrado permanente:
  - Que asegura que los datos de los archivos eliminados son realmente borrados del soporte

Necesidad de Posix ACL para Linux

- Es aconsejable en sistemas donde la flexibilidad es necesaria
- Los permisos pueden ser modificados por tres roles:
  - El Usuario
  - El Grupo
  - El Resto

No permite trabajar con permisos para usuarios diferentes!!!!

Posix ACL para Linux

- Se instala mediante un paquete de Linux
- Se encuentra en la versión de Linux con la que trabajamos
- Después de la instalación, el núcleo puede almacenar una cantidad extra de metadatos para cada fichero.
  - De esta forma, se almacena información sobre los permisos provistos por la utilidades ACL

Paquete ACL

- Encontramos información de ayuda en su instalación en al Cap. 8 del Linux Security Basic.
- El paquete de instalación además posee otros componentes como:
  - Atributos adicionales para almacenar información del ACL
  - Extiende las utilidades del ext2 para soportar los metadatos
  - Herramientas para configurar las propiedades de ACL

Comandos, setfacl

- Se utiliza para modificar los permisos para cada usuario o grupo.
  - setfacl [-m|-x] u:[user]:[+|^] permissions file [file...]
  - setfacl [-m|-x] g:[group]:[+|^] permissions file [file...]
  - setfacl [-m|-x] o:[+|^] permissions file [file...]

- m: modifica o añade permiso - ^ : Elimino permisos
- x : elimina permiso - + : Añade permisos

Ejemplo: *setfacl –m g:alumno:+w archivo*

setfacl –m u:yeray:^x archivo

setfacl –x u:yeray archivo

Comandos, getfacl

- Se utiliza para mostrar los permisos para cada usuario o grupo.
  - getfacl file [file...]

Ejemplo: *getfacl archivo*

- # file : archivo
- # owner:yeray
- # group:alumno
- user::rwx
- user:juan:r--
- group::r--
- group: profesores:rwx
- other:---
- $

Permisos por defecto

- Similar a la utilidad umask
- Se consigue utilizando el setfacl sobre directorios
  - setfacl [-m|-x] u:[user]:[+|^] permissions dir [dir...]
  - setfacl [-m|-x] g:[group]:[+|^] permissions dir [dir...]
  - setfacl [-m|-x] o:[+|^] permissions dir [dir...]
- Todos los archivos creados en ese directorio heredan los permisos por defecto

Ejemplo

- ¿Cómo quitar todos aquellos permisos configurados con el ACL que no son el del usuario, grupo u “others”?
  - Realmente no se eliminan, sino se utiliza el setfacl para modificarlo.

setfacl –m u:yeray:^x archivo

Problemas de compatibilidad

- Puede existir problemas de prioridades entre los permisos tradicionales y el ACL.
- Solución: Algoritmo de reglas de prioridad
  1. Los permisos se chequean de izquierda a derecha
  2. Prioridades:
     1. Usuarios frente a grupos
     2. Grupos frentes a resto de usuarios
     3. Los tradicionales frente a los ACL en los acceso
     4. Los ACL frente a los tradicionales en el resto

Consideraciones

- Hay núcleos que no soportan la estructura ACL
- El sistema de ficheros NFS es incompatible con ACL, lo ignora.
- Algunas utilidades de copias de seguridad no guardan ACL.
- Las utilidades setfacl y getfacl no son soportadas por sistemas de archivos no.ext2

Borrado permanente

- Cuando se borra un archivo este no es borrado de la memoria física.
  - Sólo se pierde la dirección

  Solución:

  Herramientas *wipe* y *bcwipe*
  - No incluidas en la mayoría de las versiones de Linux

Bibliografía

- Linux Security Basic
  - *Capítulo 7 : File System Security*
  - *Capítulo 8 : Extra File System Security Tools*