Sistemas Operativos

Permisos y tipos de amenazas

Departamento de Ingeniería en Sistemas y Computación
Universidad Católica del Norte, Antofagasta.

Permiso de archivo y directorios

- Todos los archivos del sistema pertenecen a algún usuario y a algún grupo
- Para visualizar el usuario y grupo propietario: `ls -l`

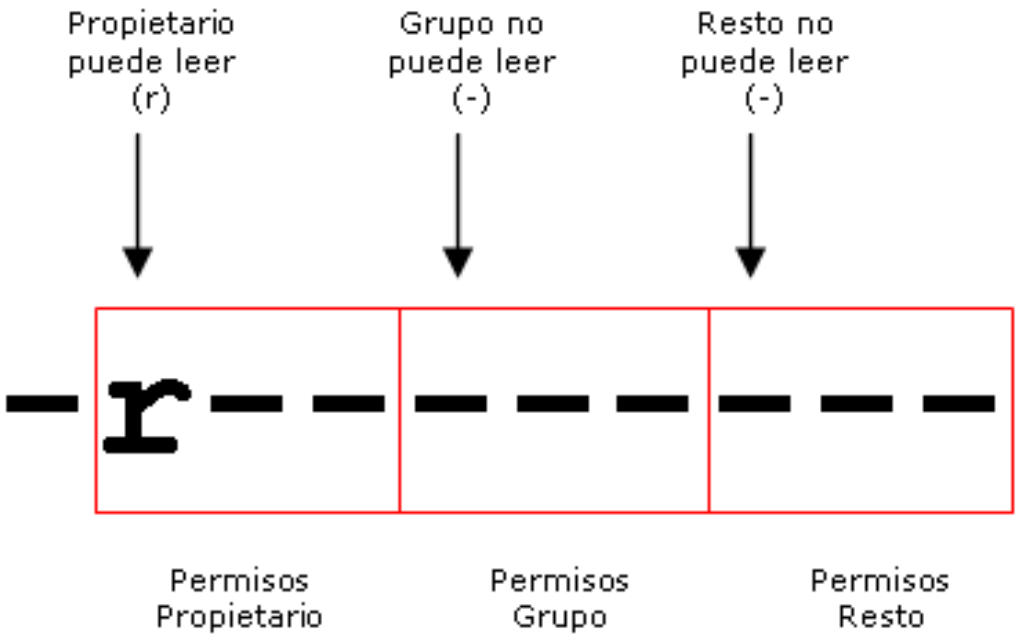
Tipos de permisos

- Permiso de lectura
- Permiso de escritura
- Permiso de ejecución

Permiso de lectura

- El usuario puede leer o visualizar el archivo, mediante aplicación o comandos.
- En carpetas: usuario puede visualizar contenido del directorio

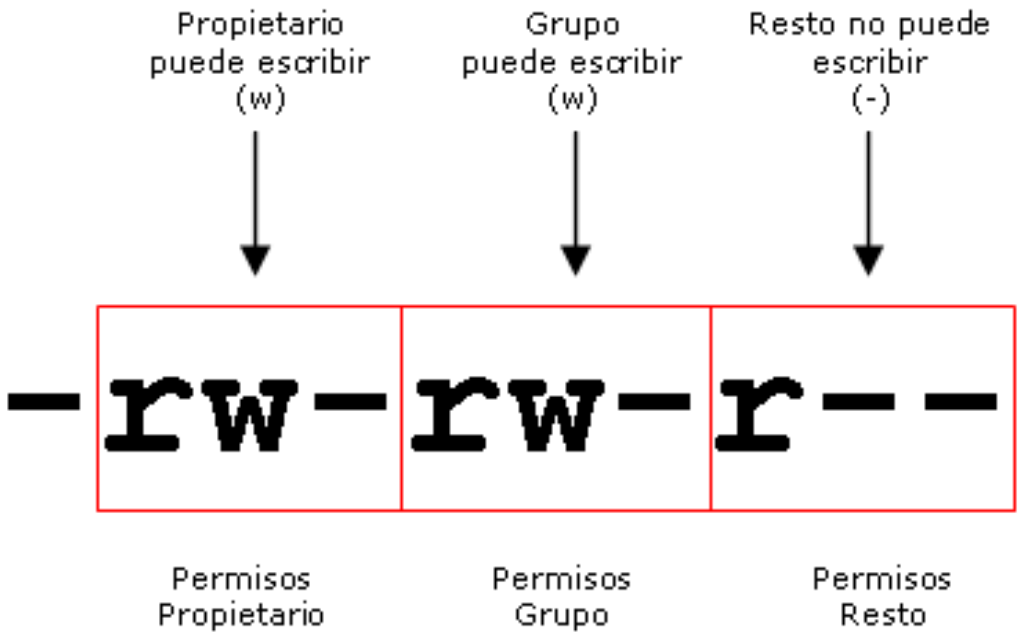
Permiso de lectura



Permiso de escritura

- El usuario puede modificar el contenido del archivo, e incluso borrarlo.
- El usuario también tiene derecho a cambiar los permisos del archivo (`chmod`)
- El usuario puede cambiar su usuario y grupo propietario (`chown`)

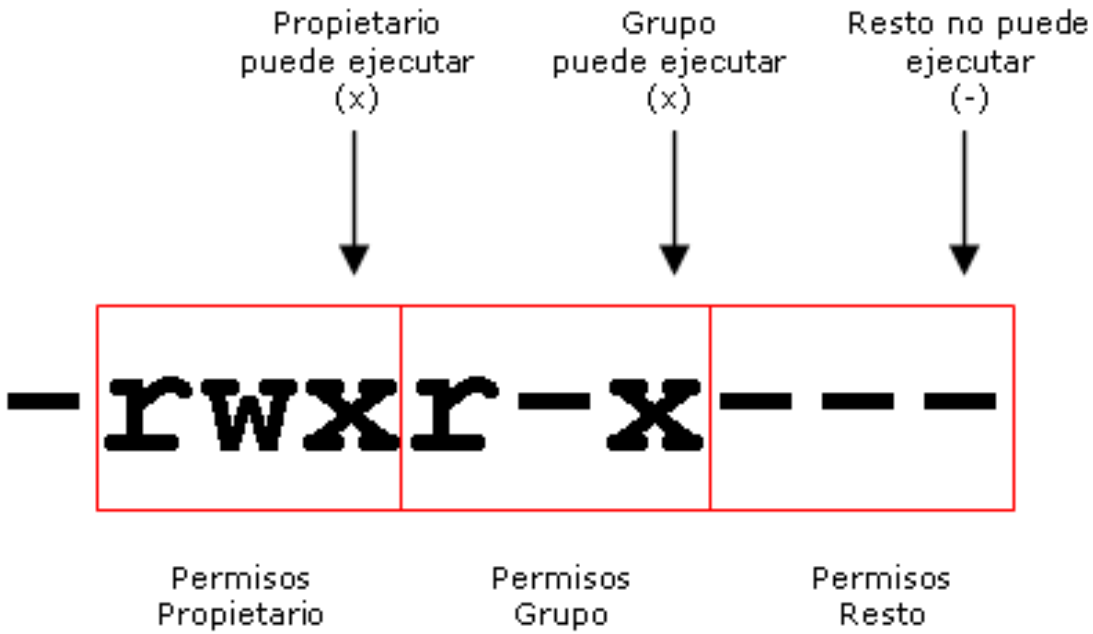
Permiso de escritura



Permiso de ejecución

- El usuario puede ejecutar el archivo, si el usuario no dispone de permiso no podrá ejecutarlo aunque el archivo sea una aplicación
- Los únicos archivos ejecutables son las aplicaciones y los archivos de comandos (scripts)
- En carpeta: el usuario puede entrar a ella mediante comandos o explorador de archivos

Permiso de ejecución



Visualización de permisos

El primer caracter indica de qué tipo de archivo se trata:

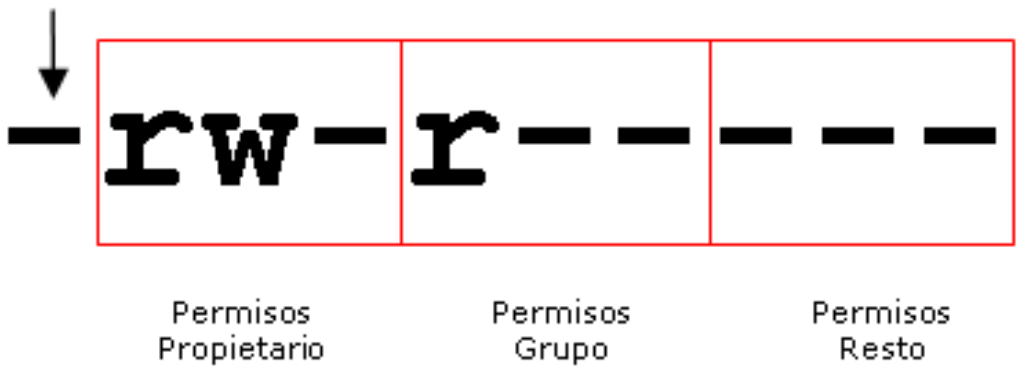
Tipo de archivo:

(-) para archivos normales

(d) para carpetas (directory)

(l) para enlaces (link)

(s)=socket, (p)=tubería (pipe), (b)=dispositivo de bloque.



Cambio de permisos

- Es necesario disponer de permiso de escritura sobre el archivo o carpeta

```
chmod [opciones] permiso nombre_archivo
```

Representación de permisos

- Iniciales de a quién va dirigido el permiso
  - u: usuario
  - g: grupo
  - o: resto (other)
- +: agregar permiso
- -: quitar permiso
- Tipo de permiso
  - r: lectura
  - w: escritura
  - x: ejecución

Representación de permisos

Ejemplos:

- chmod u+w examen.txt
- chmod o-w examen.txt
- chmod u+w,g-r,o-r examen.txt

Representación de permisos

Código numérico:

- compuesto por tres cifras (usuario, grupo, resto), cada cifra entre 0 y 7

Codigo	Binario	Permiso
0	000	—
1	001	–x
2	010	-w-
3	011	-wx
4	100	r–
5	101	r-x
6	110	rw-
7	111	rwx

Representación de permisos

Ejemplos:

- chmod 700 examen.txt
- chmod 550 examen.txt
- chmod 744 *

SUID

- bit SUID: extensión del permiso de ejecución.
- Para activar: `chmod u+s nombre_archivo`
- Cuando un usuario ejecuta una aplicación, ésta se ejecuta con permisos del usuario propietario

Seguridad en el SO

Servicios de seguridad:

- Servicios de autenticación: corresponde al paso previo a la aplicación de cualquier esquema de protección, determinando si el usuario está autorizado y sus privilegios.
- Como medida suplementaria se pueden limitar los recursos a determinadas horas del día, o expulsar al usuario después de un periodo de inactividad

Privacidad

Los derechos de acceso pueden residir en:

- El objeto: indica qué usuarios y qué derechos tiene cada uno
- El usuario: indica qué objetos y qué derechos tiene el usuario sobre estos objetos

Tipos de amenazas

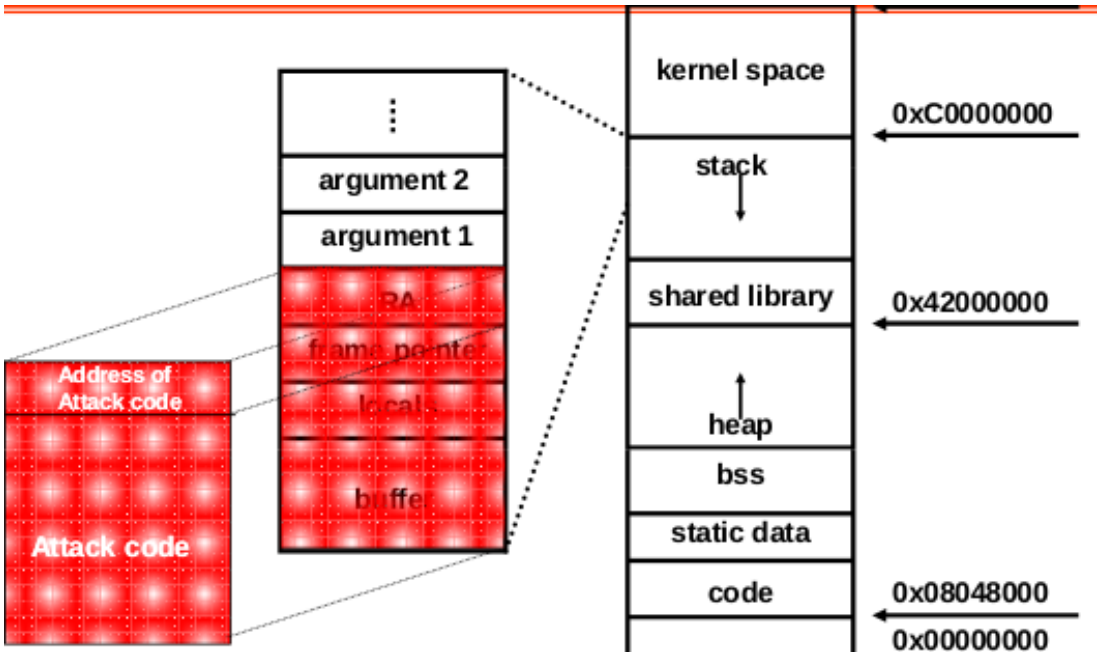
- Exploits
- Caballo de Troya
- Puertas traseras
- Virus
- Gusanos
- Ataques por denegación de servicio

Exploits

- Programa que aprovecha un error en otro programa para violar la política de seguridad
- Especialmente peligroso en programas con *setuid*

Exploits

- Corrompe la pila de un programa escribiendo más allá de los límites



Virus

- Secuencia de código que se inserta en un ejecutable
- Etapas:
  - Fase latente: virus está dormido, despertándolo un evento específico
  - Fase de propagación: el virus se replica en otros programas
  - Fase de activación: se activa para realizar las acciones para las que fue concebido
  - Fase de ejecución: La función respectiva que debe realizar se ejecuta

Estrategias de los antivirus

- Prevención y detección
  - Aumento en el tamaño de los ejecutables
  - Integridad de ejecutables (almacenar checksums)
  - Detectar operaciones potencialmente peligrosas

Estrategias de los antivirus

- Si la eliminación no es posible, el antivirus se deshace del programa infectado
- Existen estrategias más sofisticadas, como GD (Generic decryption)

Gusanos

- Su principal misión es reenviarse a sí mismo
- No afectan a la información de los sitios que contagian
- Consumen muchos recursos del sistema y los usan para infectar a otros equipos

Conejos (o bacterias)

- No dañan directamente al sistema
- Se reproducen hasta que la cantidad de recursos consumidos se convierte en una negación de servicio para el sistema afectado

```
main(){
    while(1){
        malloc(1024);
        fork();
    }
}
```

Troyanos

- El programa parece ser útil, pero además hace cosas no autorizadas
- El usuario ejecuta voluntariamente el programa malicioso

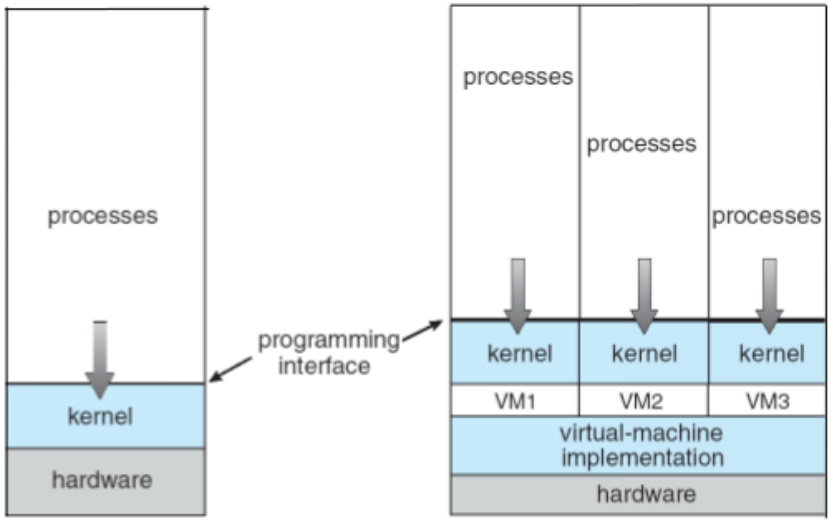
Puertas traseras

- Trozos de código que permiten saltarse los métodos de autenticación
- Generalmente los programadores lo usan para realizar tareas de pruebas
- Puertas traseras en archivos del sistema operativo:
  - Añadir un usuario con UID 0
  - Añadir un nuevo servicio a un puerto

Ransomware

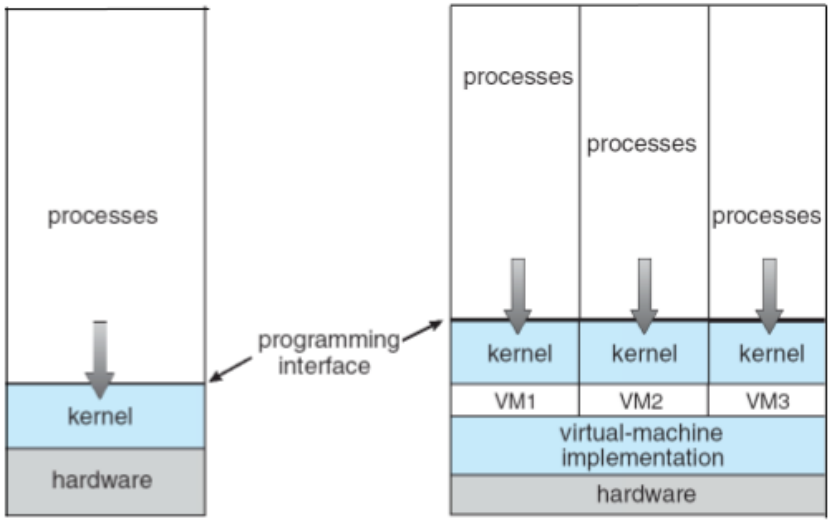
- Se camufla dentro de otro archivo o programa, transmitiendose como un troyano o un gusano.
- Restringe el acceso a determinados sectores o archivos del sistema infectado, pidiendo un rescate a cambio de quitar esta restricción.
- Algunos tipos de ransomware cifran los archivos del sistema operativo, inutilizando el dispositivo.
- Su uso creció en el último tiempo, haciéndose populares el 2013

Máquinas virtuales



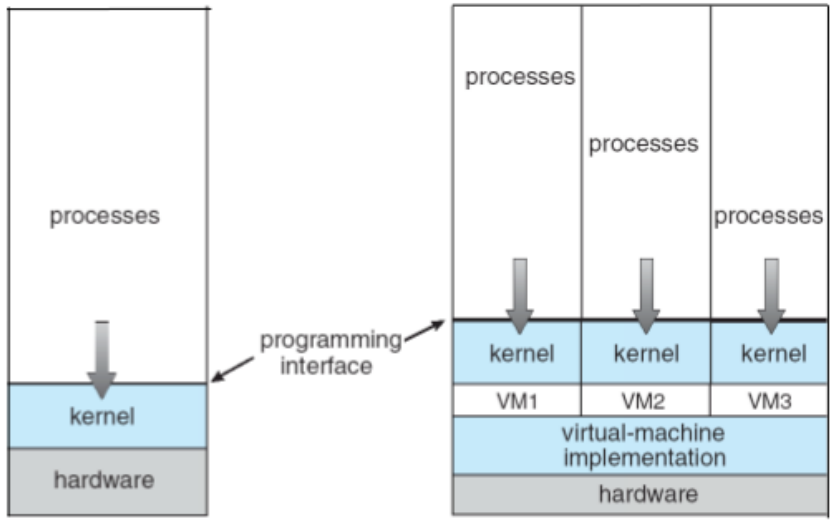
- Se estructuran en base a la aproximación por capas o niveles

Máquinas virtuales



- Trata el hardware y el kernel del SO como si todo fuese hardware

Máquinas virtuales



- El **SO host** crea la ilusión que un proceso tiene su propio procesador y memoria (virtual)

Windows

- SO multitarea, basado en diseño de 32 bits
- Proporciona espacios de memoria separados para cada proceso
- Soporta multiprocesamiento simétrico en máquinas con 2 procesadores
- Se puede ejecutar en procesadores con CISC o RISC

Principios de diseño de Windows

- Tiene un diseño de tipo microkernel (o micronúcleo)
- Se desarrolló desde cero, fusionando y optimizando características ya probadas en sistemas como UNIX, VMS o MACH
- Se mantuvieron algunas ideas desarrolladas en MS-DOS y Windows 3.x

Principios de diseño de Windows

Principios de diseño fundamentales

- **Compatibilidad**: con sistemas anteriores (sistema de archivos FAT de Windows 3.x) como con otros SO (POSIX) y con distintos entornos de red, a través de subsistemas que emulan los servicios de los distintos SO.
- **Transportabilidad**: para facilitar el transporte a distintos computadores con procesadores CISC (Intel) y RISC (MIPS, Digital Alpha), construye una capa de abstracción de hardware (HAL) para proporcionar toda la funcionalidad dependiente del hardware al núcleo del sistema.

Principios de diseño de Windows

Principios de diseño fundamentales

- **Escalabilidad**: se diseñó de forma modular sobre la base de un micronúcleo, permitiendo repartir elementos del sistema sobre distintos procesadores y extender el sistema con nuevos componentes.
- **Fiabilidad y robustez**: se han incluido servicios para dar más robustez al sistema tanto en ámbito de procesos como en el sistema de archivos, como por ejemplo los sistemas de archivos con puntos de recuperación.

Historia de Windows

- **MS-DOS**: MicroSoft Disk Operating System, sistema operativo monousuario de 16 bits, manejado por línea de comandos, consistía en 8 KB de código residente en memoria.
- **MS-DOS 2.0**: dos años después se incluyó un procesador de líneas de comandos con varias características tomadas de UNIX.
- **Windows 1.0**: incluye una interfaz gráfica de usuario a MS-DOS, inspirado por Apple Lisa (anterior a Apple Macintosh)

Historia de Windows

- **Windows 3.0**: tuvo gran éxito comercial, pero principalmente consistía de una interfaz gráfica de usuario puesta encima de MS-DOS, quien seguía al control de la máquina y el sistema de archivos. Al ejecutar todos los programas en el mismo espacio de direcciones, un error de programación en cualquiera de ellos podía paralizar el sistema.
- **Windows 95**: incluyó funciones como gestión de memoria virtual, administración de procesos y multiprogramación.
- **Windows 98**: MS-DOS seguía presente en su versión 7.1 y ejecutando código de 16 bits, la diferencia principal con Windows 95 radicaba en la interfaz de usuario integrando de forma más estrecha el escritorio e Internet.

Linux

- hace referencia al núcleo libre de un SO basado en Unix
- problema con controladores específicos para cierto tipo de hardware ha mejorado con el tiempo
- Ejemplos:
  - Ubuntu
  - Debian
  - Arch Linux
  - Fedora
  - Elementary

SO para Servidor

- Corresponde a un software utilizado como plataforma que soporta programas multiusuarios y aplicaciones en red entre otros ejemplos de herramientas críticas.
- Algunos de la familia Microsoft: Windows 2000 Server, Windows Home Server
- Windows Server 2012 integra características como redes, virtualización, computación en la nube, almacenamiento y automatización entre otras.

Unix

- Desarrollado a fines de los 60, intentando realizar una copia de MULTICS
- Utiliza sistema de archivos y directorios jerárquico
- Administración del sistema, uso de recursos de red, administración BD, manejo de servicios web
- Alto nivel de seguridad, estabilidad de los procesos, flexibilidad en los tipos de configuración

SO para servidor Unix

- CentOS
- Ubuntu Server
- Red Hat Enterprise Linux Server

Servidores Apple

- Utilizan Mac OS X Server, el cual tiene componentes de Unix
- Cuenta con varias herramientas administrativas en modo gráfico, como para administración de usuarios, redes, servicios.

Comparación de arquitectura Windows-Linux

- Para ambos se tiene un área compartida en la que se corren los servicios requeridos formando parte de un archivo de configuración
- La interfaz se maneja diferente: en Windows se basa en un sistema de ventanas modo kernel, mientras que en Linux se corre en modo usuario utilizando un sistema X-Windows

Otras aplicaciones

- **ROS**: Robot Operating System
- desarrollado originalmente en 2007 en el Laboratorio de Inteligencia Artificial de Stanford
- provee servicios estándar de un SO, como abstracción del hardware, control de dispositivos de bajo nivel, paso de mensajes entre procesos.