GRUPO DE SEGURIDAD INFORMÁTICA

Fundamentos de Seguridad Informática

Seguridad en UNIX

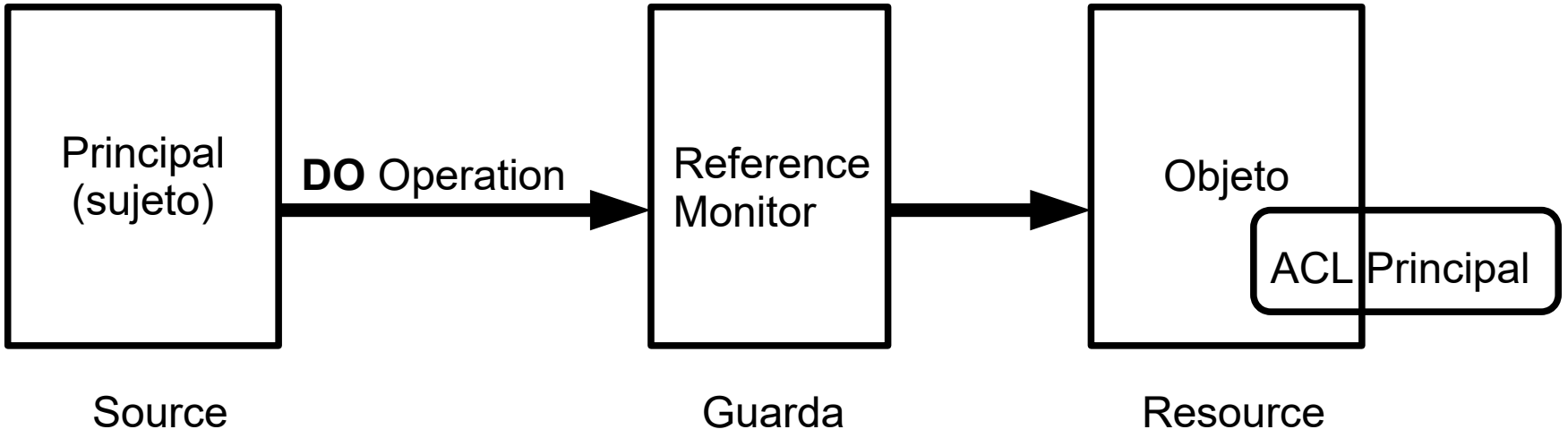
Plan

- Conceptos preliminares
- *Principals (UID / GID)*
- Sujetos (PID)
- Objetos (files)
- Control de acceso
- Políticas de complejidad de claves
- SUID/SGID, sandboxing, logging, hardening

Vamos a hacer foco en ..

- Controles de seguridad en sistemas operativos
  - Identificación, autenticación
  - Control de acceso
  - Auditorías, instalación y configuración

En general tenemos ..



Conceptos preliminares

- Información sobre usuarios (sujetos) se guarda en *cuentas de usuario*
- Cualquier *privilegio* permitido a un usuario puede guardarse en esta cuenta
- La *identificación* y *autenticación* verifican la identidad del usuario
- Esto permite al sistema asociar los privilegios de un usuario a cualquier proceso iniciado por éste

Conceptos preliminares

- Permisos sobre recursos (objetos) pueden asignarse por el administrador o el propietario
- Para decidir acceso a un recurso, el SO puede referirse a la identidad del usuario, a los privilegios establecidos para el sujeto y los permisos sobre el objeto

en UNIX debemos tener en cuenta ..

- Originalmente diseñado para pequeñas redes de computadoras, ambientes multi-usuario
- Desarrollado para ambientes amigables, labs. de investigación y Universidades
- Mecanismos de seguridad elementales; mejorados gradualmente
- Implementa DAC con granularidad a nivel de owner, group, other.
- Existen versiones de Unix “seguras” con soporte multi-level security

Security Architecture

- Los controles de seguridad en UNIX no están contemplados en los objetivos de diseño iniciales
- Nuevos controles de seguridad se han incorporado, otros se fortalecieron a demanda
- Siempre intentando interferir lo menos posible con las estructuras existentes
- La seguridad es gestionada por administradores con mucha experiencia

Principals

- Los principals son llamados *user identities* (UIDs) y *group identities* (GIDs)
- Ambos son enteros de 16 bits
- Algunos UIDs tienen significados especiales
- El *superusuario* en unix es siempre 0

Cuentas de usuario

- Las informaciones sobre los *principals* son guardadas en cuentas de usuario y directorios home (personal)
- Cada individuo debe tener su propia cuenta
- No deben haber cuentas grupales **¡tip!**
- Todas las cuentas deberían autenticarse de la misma forma

Cuentas de usuarios en UNIX

- Cada cuenta se registra con una entrada en el archivo /etc/passwd
- y en el /etc/shadow
- La cuenta *root* es el superusuario
- Existen otras cuentas por defecto
  - **siempre** deberían estar bloqueadas **¡tip!**

UID 0

- El login del usuario root no es lo que determina los permisos de ese usuario
- El UID es el que lo hace
- Si un usuario tiene UID 0, aunque no tenga como login root, es equivalente al superusuario
- Periódicamente hay que revisar /etc/passwd para verificar que hay una única cuenta con UID 0 y que a la vez tiene login root

/etc/passwd

- Es una tabla con los siguientes campos

login:contraseña encriptada:UID:GID:comentario:directorio personal:shell

Dueño: root; grupo: root

Permisos: rw para el dueño, r para el resto

- Ej:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
```

Grupos de usuarios

- Cada usuario está presente en al menos un grupo de usuarios
- Agregar usuarios a grupos es una base conveniente para decisiones de control de acceso
- El GID de /etc/passwd es el grupo primario
- Ejecutando *groups* se pueden saber los grupos secundarios

Sujetos

- Los sujetos son los procesos del sistema
- Cada uno tiene un *process ID* (PID)
- Se crean nuevos mediante *fork* o *exec*
- Cada proceso tiene asociado un UID/GID *real*, y uno *efectivo*
- El UID real es heredado del padre
- El UID efectivo es heredado del padre o del archivo que se está ejecutando

Login y Passwords

- Los usuarios se identifican mediante nombres de usuario y passwords
- Cuando un usuario se loguea, el proceso *login* verifica el usuario y password
- Si la verificación es exitosa, se cambia el UID/GID al del usuario y se ejecuta la shell de login
- Los passwords se cambian mediante el comando *passwd*

/etc/shadow

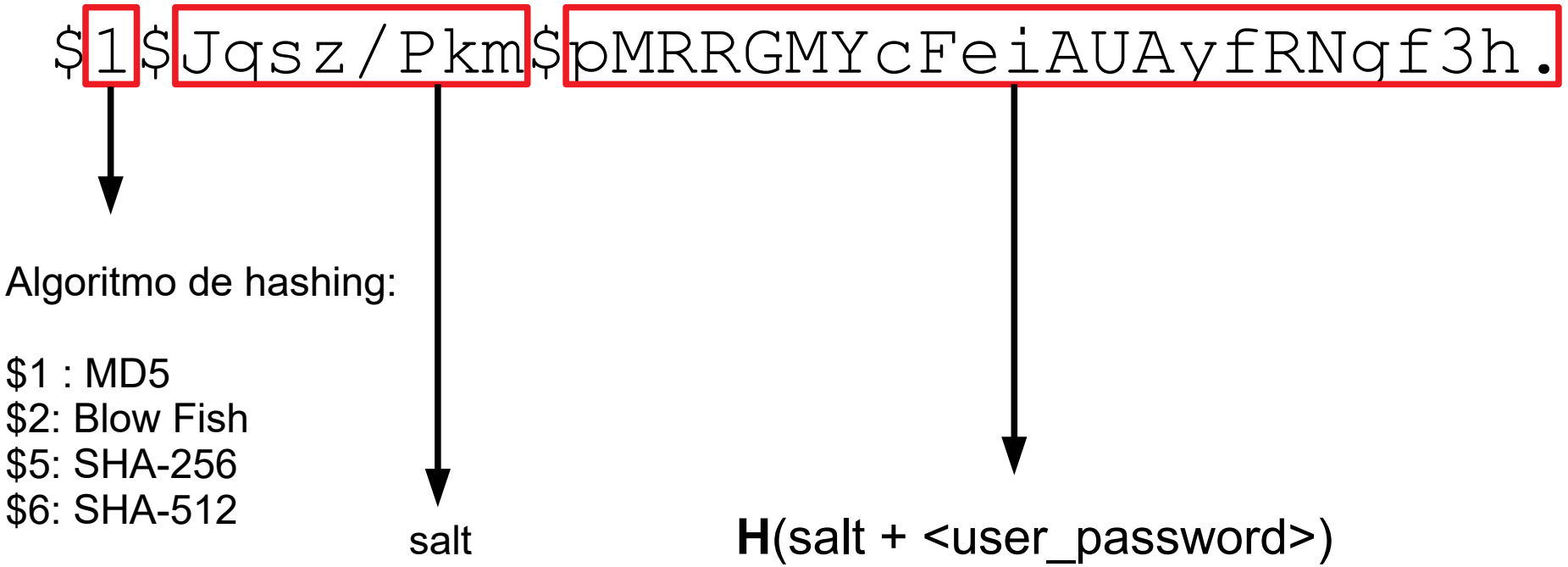
- Si está en el sistema, la contraseña se cambia por un ‘+’ en /etc/passwd
- Dueño: root; grupo: root
- Permisos: r para el dueño, nada para el resto
- Es una tabla con los siguientes campos

Ejemplo: /etc/shadow

- Los restantes campos están vacíos pues son valores por defecto

```
root:$1$Jqsz/Pkm$pMRRGMYcFeiAUAyfRNgf3h.:13228:0:99999:7:::
shutdown:*:13228:0:99999:7:::
mail:*:13228:0:99999:7:::
news:*:13228:0:99999:7::
nscd:!!:13228:0:99999:7:::
named:!!:13228:0:99999:7:::
netdump:!!:13228:0:99999:7:::
sshd:!!:13228:0:99999:7:::
rpc:!!:13228:0:99999:7:::
apache:!!:13228:0:99999:7:::
ntp:!!:13228:0:99999:7:::
mysql:!!:13230:::::
glest:$1$A3TUG3YS$DT4le2QA7FEiric28Nh94.:13305:0:99999:7:::
```

Password hash



Objetos

- Los objetos para el control de acceso incluyen: archivos, directorios, dispositivos, I/O, etc
- Están organizados en un sistema de archivos con estructura de árbol

Inodos

- Cada entrada de archivo en un directorio es un puntero a una estructura de datos llamada *inodo*
- Cada directorio tiene un puntero a sí mismo, el archivo '.', y un puntero a su padre, el archivo '..'
- Cada archivo tiene un dueño, usualmente el usuario que lo creó
- Cada archivo pertenece a un grupo

Permisos en archivos

- Los permisos de los archivos (*bits de permisos*), se agrupan en tres tripletas
- Definen permisos de *read*, *write*, y *execute*, para el propietario, el grupo, y otros
- También se pueden especificar como números octales, separando los nueve permisos en 3 grupos de 3
- Cada derecho de acceso está representado por un bit, el cual si está prendido (en 1), permite el acceso

Permisos en archivos (2)

- La combinación de derechos es la suma de los números correspondientes
- El permiso 777 da todos los accesos al dueño, grupo y mundo

400	Lectura por el owner
200	Escritura por el owner
100	Ejecución por el owner
040	Lectura por el grupo
020	Escritura por el grupo
010	Ejecución por el grupo
004	Lectura por el mundo
002	Escritura por el mundo
001	Ejecución por el mundo

Permisos por defecto

- Las utilidades UNIX, como editores o compiladores, usan típicamente los permisos 666 al crear un nuevo archivo
- Deben ajustarse con la ***mask***
- Los permisos por defecto son derivados de un **AND** entre los bits por defecto y el inverso de la máscara
- Ver comando umask

Permisos para directorios

- Cada usuario tiene un directorio *home*
- Se crean con el comando *mkdir*
- Para poner archivos en ese directorio, se deben tener los permisos adecuados.
  - Permisos de lectura permiten a un usuario encontrar archivos en el directorio
  - Permisos de escritura permiten a un usuario agregar y borrar archivos al directorio
  - Permisos de ejecución se requieren para hacer el directorio el actual y para abrir archivos

Control de acceso

- Está basado en atributos de los sujetos (procesos) y de los objetos (recursos)
- Las sistemas Unix clásicos asocian tres conjuntos de derechos de acceso, con cada recurso correspondientes a owner, grupo y world
- El superusuario no está sujeto a estos chequeos

Chequeo de permisos

- Si el *uid* indica que se es dueño del archivo, los bits de permisos de *owner* (o dueño) deciden si se tiene acceso
- Si no es dueño del archivo, pero el sujeto pertenece *grupo* dueño del archivo, se aplican los permisos del grupo
- Si no, se aplican los permisos de *world* (*mundo*)
- Se podría dar el caso en que el dueño del archivo tenga menos permisos que el *grupo* dueño o (world) el resto del *mundo* !!

Las contraseñas de los usuarios

- Son fáciles
- En general son palabras presentes en un diccionario o similares
- No nos gusta cambiarlas y solemos usar la misma contraseña para todo ...
- En un estudio del '90 se demuestra que un 24,2% de los passwords se quiebran con un simple diccionario (Klein90)

Contraseñas

- Algunas medidas que podemos adoptar para mitigar este riesgo:

1) Limitar la cantidad de login fallidos
2) Evitar la repetición de contraseñas
3) Obligar a que los usuarios las cambien
4) Impedir que las contraseñas sean obvias
5) Eliminar las cuentas sin uso
6) Limitar el horario de uso del sistema

Valores por defecto en cuentas **Linux**

- /etc/default/useradd
  Se modifica mediante el comando *useradd -D*
- /etc/logindefs
  Sirve para controlar los parámetros de aging de los passwords, uids, gids, etc
- usermod

```
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=no
```

Valores por defecto en cuentas **Solaris**

- /etc/default/passwd

```
# passwd performs dictionary lookups if DICTIONLIST or DICTIONDBDIR
# is defined. If the password database does not yet exist, it is
# created by passwd. See passwd(1), pam_authtok_check(5) and
# mkdict(1) for more information.
#
DICTIONLIST=/var/adm/wordlists/english,/var/adm/wordlists/spanish,/var
/adm/wordlists/french,/var/adm/wordlists/words
DICTIONDBDIR=/var/adm/passwd
```

Valores por defecto contraseñas

- Generador de contraseñas pronunciables

  existen: pwgen, mkpasswd

```
./pwgen
Idah5eez ye6Ih3Po eojaid0D Ahg8voob uS5ohGhe Cahpae2v aVeiCh2a eejei1Ee
EiQu7aik tohXe6oh nahN9nah Ool1uphu aiF8ooce Me6ugeiw ahR6eejo Zo2Aeque
ohShuu2r eoR2gai0 Zo8queth daF1peih eiF9ao1o eeBaht8C Iephoo1r aS7Ahl6u
Ca5So3je LeuV9aev Nee8Aolu seiy0eiW wuuzeeF1 ga9Ak6En iepho4Vi jooth7Oo
zoW0ohth Egha4xoo Uwae3bai zeeVonu4 Eiciish9 Angeid0W KaeZahG1 Oophah3o
Eelei5te ba5aiFie jao7cieG Bah6fi3k ohz1Eixe Iepoo6Ki se9Kishe PariFua5
```

- Longitud de contraseñas

  /etc/default/passwd: PASSLENGTH=6

Valores por defecto contraseñas

- Tiempo para bloquear cuenta sin uso

```
-f inactive        Specify the maximum number
                   of days allowed  between
                   uses of a login ID before
                   that  login  ID  is  declared
                   invalid.  Normal values are
                   positive integers. A value of 0
                   defeats the status
```

- No permitir cuentas sin contraseñas, o con passwords fácilmente “crackeables”
  - pam_cracklib, pam_cracklib2, etc

SUID/SGID

- Necesitamos poder ejecutar procesos con privilegios de root
- Cuando? por ejemplo:
  - para escuchar en puertos < 1024
  - Para modificar la contraseña
- Solución adoptada en Unix:
  Set UserID (SUID) o Set GroupID (SGID)
- Programas con el SUID o SGID ejecutan con el UID o GID efectivo del usuario o grupo dueño del archivo

SUID a root

- En archivos con SUID root, el usuario que lo ejecuta obtiene estos privilegios durante la ejecución
- Algunos ejemplos

`/bin/passwd`	cambio de contraseña
`/bin/login`	programa de login
`/bin/at`	ejecución de programas batch
`/bin/su`	cambiar el UID (switch User ID)

- Debemos cuidar que estos programas hagan **solo** lo que deben hacer

Riesgos de SUID

- Si logramos engañar a un programa con SUID de root, estamos logrando acceso de root
- Estos programas deben procesar con mucho cuidado los parámetros de entrada
- Usemos SUID/SGID sólo cuando es estrictamente necesario
- Debemos controlar la integridad de estos programas (Tripwire, AIDE, Yafic)

Invocaciones controladas

- Combinando ownership, permisos y programas SUID
  - Ver caso de estudio ....

  o también con técnicas complementarias:

- **Jail root**,
- Containers o **sandboxing**,
- **wrappers**

Sandbox con chroot

- Implementamos restricciones de control de acceso, restringiendo los procesos a un ambiente de *sandbox*
- No se permite el acceso a objetos fuera del sandbox
- El comando chroot cambia la raiz del file System

```
chroot <directory> <command>
```

- Solo accedemos a los objetos bajo la nueva raiz

Otros mecanismos de sandbox

- Virtualización a nivel de Sistema Operativo
  - Solaris containers, Linux-VServers, OpenVZ, etc
- para-virtualización o full virtualization
  - VMware, QEMU, KVM, Xen, VirtualBox
- Otros ejemplos interesantes
  - Qubeos, https://www.qubes-os.org

Otros mecanismos de control

- Variables de ambiente
- Searchpath
- Wrappers

Además necesitamos ..

- El registro de eventos de auditoría del sistema y las aplicaciones del S.O
  (**logging**, syslog, gestión de logs, etc)
- Instalación y configuración inicial del sistema (**Hardening**)
- **Gestión de configuración y**
  actualizaciones de seguridad (**patches**) durante todo el ciclo de vida

Bibliografía y material de referencia

- **D. Gollman**, *Computer Security*, Wiley, 2006
- **Daniel Klein,** *A survey of, and improvements to, password security* In Proceedings of the USENIX Second Security Workshop, Portland, Oregon, 1990
- ***The Center for Password Sanity***, https://cryptosmith.com/password-sanity/
- **Password crackers**, http://www.password-crackers.com