

# Administración de Sistemas y Seguridad

Máster en Ingeniería Informática  
Universidad de Granada

(c) Miguel García Silvente

## Tema 2

Control de acceso y seguridad en sistemas operativos.

## Conceptos fundamentales

- La **identificación** es la capacidad de identificar de forma exclusiva a un usuario de un sistema o una aplicación que se está ejecutando en el sistema.
- La **autenticación** es la capacidad de demostrar que un usuario o una aplicación es realmente quién dicha persona o aplicación asegura ser.
- Por ejemplo: login / password

(c) Prof.Miguel García Silvente

3

## Índice

1. Autorización y control de acceso
2. Cifrado
3. Seguridad de un sistema: SELinux
4. Malware
5. Simulación de sistemas

(c) Prof.Miguel García Silvente

4

## Autorización y control de accesos

- Principio del “mínimo privilegio”. Windows, servidor web,...
- Minimizar las relaciones de confianza.
- Accesos:
  - Permitir,
  - Denegar,
  - Limitar (sandbox)
  - Revocar.
- Listas de control de acceso (ACLs)
- Capacidades

(c) Prof.Miguel García Silvente

5

## Acceso a ficheros en Unix

- Cada fichero y directorio tiene un propietario y un grupo.
- Los permisos los define el propietario
  - Lectura, escritura y ejecución (o acceso).
  - Para propietario, grupo y otros.
  - Se utilizan 4 números en código octal.
- Sólo el propietario y root pueden cambiar los permisos.
  - No se puede delegar o compartir ese privilegio.
- Existen “bits de stick” (En Windows existe la “suplantación”)

(c) Prof.Miguel García Silvente

6

## Ejemplo de permisos de acceso

	u	g	o
	7	5	4
acceso	r w x	r w x	r w x
binario	4 2 1	4 2 1	4 2 1
activado	1 1 1	1 0 1	1 0 0
resultado	4 2 1	4 0 1	4 0 0
total	7	5	4

(c) Prof.Miguel García Silvente

7

## Resolución de privilegios en Unix

Si el usuario = **propietario** tiene permiso  
si no,  
si usuario está en el grupo tiene permisos de **grupo**  
si no, tiene los permisos para **otros**

(c) Prof.Miguel García Silvente

8

## ID de usuario efectivo (EUID)

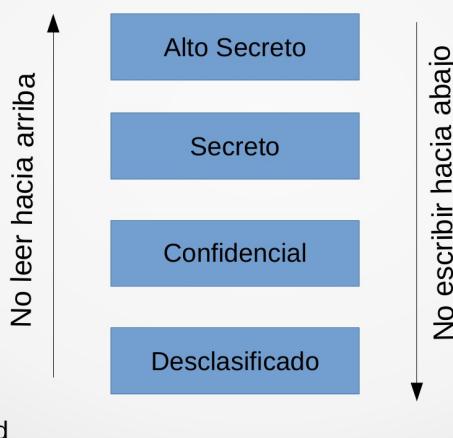
- Cada proceso tiene 3 identificadores (IDs)
  - ID de Usuario real (RUID). Determina el usuario que lanzó el proceso.
  - ID de usuario efectivo (EUID). Puede ser distinto al anterior si se usa el bit de stick o se usa una llamada al sistema.
  - ID de usuario guardado (SUID). Si se bajan privilegios.

(c) Prof.Miguel García Silvente

9

## Otros modelos de acceso (I)

- **Modelo de confidencialidad Bell-LaPadula:** Sólo se puede leer hacia abajo y sólo se puede escribir hacia arriba.



(c) Prof.Miguel García Silvente

10

## Otros modelos de acceso (II)

- **Modelo de integridad de Biba:** Sólo se puede escribir hacia abajo y leer hacia arriba. Integridad
- **Modelo de matriz de acceso.** Con usuarios x permisos.
- **Separación de funciones:** Dos personas diferentes deben tomar una decisión sobre aspectos importantes.
- **Muralla china:** Dos personas de una misma entidad no pueden trabajar para dos partes enfrentadas.

## Otros modelos de acceso (III)

- Discretionary Access Control (**DAC**): Una entidad con privilegios de acceso puede pasar esos privilegios a otras entidades.
- Mandatory Access Control (**MAC**): El control no depende únicamente del propietario. Se definen restricciones a nivel de administrador del sistema o del núcleo del sistema.

## Control de acceso

- Basado en rol.
  - Un usuario según su categoría.
- Basado en atributos.
  - Debes medir más de X para montar a caballo.
  - Completely Automated Public Turing test to tell Computers and Human Apart. (CAPTCHA)
  - Horario de acceso a un edificio.
  - Reconexión cada, p.ej., 24 horas a VPNs

## ACL en Linux (I)

- Sistema adicional al sistema estándar UNIX.
- Permite definir privilegios específicos a usuarios y grupos concretos.
- Activación con acl en /etc/fstab o por defecto (tune2fs -l ...)
- **setfacl -m g:<grupo>:r <directorio>/**
- **-d** define el comportamiento por defecto
- **-b** elimina todos los acls, **-x** elimina algunos concretos
- **getfacl <directorio>**
- Aparece un + al final de la información del fichero.

## ACL en Linux (II)

Ejemplos:

- Se puede dar permisos a grupos concretos en lugar de a todos
  - **setfacl** -m g:pepe:r \$HOME
- O incluso a usuarios concretos
  - **setfacl** -m u:nobody:r quiensoy
- Importante: chmod puede hacer que los ACLs pasen a #effective:---
- Se pueden copiar los ACLs
  - **getfacl** <fichero1> | **setfacl** -f - <fichero2>

(c) Prof.Miguel García Silvente

15

## Privilegios en Unix

- El usuario root tiene todos los privilegios.
- Se pueden conceder ciertos permisos a usuarios e incluso con opciones concretas: **sudo**
- Se configura en */etc/sudoers*
- El fichero tiene 3 partes: **alias, opciones, reglas**
- **sudo -l** Informa de los privilegios de un usuario.
- Ejemplo:

```
Cmnd_Alias NSHELLS = /bin/sh,/bin/bash  
Cmnd_Alias NSU = /bin/su  
testing ALL= ALL, !NSHELLS, !NSU
```

(c) Prof.Miguel García Silvente

16

## Claves

220 millones de dólares en Bitcoin, una contraseña perdida y dos intentos restantes: la historia de un ingeniero alemán incapaz de acceder a su cartera



– Pueden ser revocadas.

HOY SE HABLA DE

Vacunas — Mejores móviles —  
Smartwatch — Android Auto  
eléctrico — Xiaomi — Oppo

PUBLICIDAD

(c) Prof.Miguel García Silvente

17

## Ataques al uso de claves

Formas de averiguarlos (cracking):

- Probar todas las claves (ataque por fuerza bruta).
- Probar claves habituales (basado en diccionario).
- Las más probables para un usuario (usando ingeniería social).
- Buscar en el fichero que contiene las claves.
- Preguntar al usuario (simulando login). Surgen los IDPs

(c) Prof.Miguel García Silvente

18

## Ejemplo: claves de windows

- Averiguar claves (No detectable):
  - Fichero **SAM** en **%SystemRoot%\system32\config**
  - Herramientas de recuperación de claves: LCP, SAMDump, SAMinside, pwdump, Cain & Abel
- Comprometer sistema:
  - Arrancar con un dispositivo USB
  - Mover el fichero **logon.scr** a otro sitio y copiar cmd.exe a logon.scr.  
De esa forma el sistema arranca sin pedir usuario y clave.
  - Se puede cambiar la clave del administrador con  
**net user administrator password**
- Reiniciar claves: Arrancando el sistema con un dispositivo usb  
conteniendo **Windows Key**

(c) Prof.Miguel García Silvente

19

## Sistemas de autenticación

- Sistemas de autenticación mal diseñados:
  - Informan de que un usuario no es válido o que la clave es incorrecta (de forma independiente).
- Políticas públicas de definición de claves:
  - Ej. Indicar que debe tener longitud 8 y contener números pero no caracteres especiales.

(c) Prof.Miguel García Silvente

20

## Ataques de fuerza bruta (I)

- Probar todas las posibles combinaciones.
- Ejemplos de tiempo necesario:
  - De 1 a 8 letras minúsculas o mayúsculas, números, y caracteres especiales:  $6.1 \times 10^{15}$  posibilidades
- Si se sabe que debe tener longitud 8, se pueden descartar  $3.5 \times 10^{12}$  casos
- La cuestión es el número de encriptaciones por segundo que se pueden realizar para un algoritmo concreto
  - <http://lastbit.com/pswcalc.asp>
  - [https://tmedweb.tulane.edu/content\\_open/bfcalc.php](https://tmedweb.tulane.edu/content_open/bfcalc.php)

(c) Prof.Miguel García Silvente

21

## Ataques de fuerza bruta (II)

Si disponemos de un sistema con capacidad para realizar 2,000,000,000 encriptaciones/s

<https://github.com/lattera/glibc/blob/master/crypt/sha512-crypt.c>

- 5 caracteres (3 minúsculas, 2 números)
  - $36^5 = 60,466,176$  combinaciones
  - $60,466,176 / 2,000,000,000 = 0.03$  s
- 7 caracteres (1 mayúscula, 6 minúsculas)
  - $52^7 = 1,028,071,702,528$
  - $1,028,071,702,528 / 2,000,000,000 =$
  - 514 s. = aprox. 9 minutos

(c) Prof.Miguel García Silvente

22

## Ataques de fuerza bruta (III)

- 8 caracteres (4 minúsculas, 2 caracteres especiales, 2 números)
  - $68^8 = 457,163,239,653,376$  combinaciones  
 $457,163,239,653,376 / 2,000,000,000 = 228,581$  s. =  
aprox. 2.6 días
- 9 caracteres (2 mayúsculas, 3 minúsculas, 2 números, 2 caracteres especiales)
  - $94^9 = 572,994,802,228,616,704$ 
    - $572,994,802,228,616,704 / 2,000,000,000 = 286,497,401$  s = aprox. 9.1 años

(c) Prof.Miguel García Silvente

23

## Ataques de fuerza bruta (IV)

- 12 caracteres (3 mayúsculas, 4 minúsculas, 3 caracteres especiales, 2 números)
  - $94^{12} = 475,920,314,814,253,376,475,136$  combinaciones
  - $475,920,314,814,253,376,475,136 / 2,000,000,000 = 237,960,157,407,127$  s = aprox. 7.5 millones de años

(c) Prof.Miguel García Silvente

24

## Ataques basados en diccionarios

La mayoría de las claves no son secuencias aleatorias sino combinaciones de palabras con otros elementos como, por ejemplo, números:

- Se puede usar un diccionario.
- Se puede usar una lista de palabras junto con los elementos que un usuario podría añadir o que aparecen en la política.

## Ataque basado en conocimiento

- Cada persona tiene características que pueden ser aprovechadas.
- Conociéndola es más fácil adivinar su clave:
  - Relacionado con su vida, intereses o preferencias.
  - Ejemplos: nombres familiares, cumpleaños, series favoritas, alimentos, colores, aficiones, etc
- Unido a un ataque basado en diccionario: reducción de la búsqueda.
- Mucha de esa información está disponible en redes sociales.

## Estimación de tipos de claves

- Dos caracteres: 2%
- Tres caracteres: 14%
- Cuatro caracteres: 14%
- Cinco letras (todas mayúsculas o todas minúsculas): 22%
- Seis letras minúsculas: 19%
- Palabras de diccionarios o listas: 15%
- Otros: 14%

(c) Prof.Miguel García Silvente

27

## Algoritmos de encriptación y hash

- DES
- MD5
- SHA512
- RC5
- En sistemas basados en RedHat (CentOS, Fedora):
  - Algoritmo usado en el sistema: **authconfig** --test | grep hashing
  - Cambiar algoritmo:  
**authconfig** --passalgo=sha512 --update
    - Opciones: decrypt bigcrypt md5 sha256 sha512
- En ubuntu:
  - /etc/login.defs ENCRYPT\_METHOD
  - /etc/pam.d/common-password
    - password [success=2 default=ignore] pam\_unix.so obscure  
use\_authtok try\_first\_pass sha512

(c) Prof.Miguel García Silvente

28

## Claves en linux (I)

- cat /etc/shadow

xxx:\$6\$/0vUbQKa\$Eq2HbkKVz0EBs0KDDYAY88DujzK4T8PLr2Or9tggykTuS7EZL/UZ4UA5jkJfPnStEpCnPEhJdPRa0ChMnMBCs0:16540:0:99999:7:::
- Primer campo identifica el algoritmo (\$6):
  - \$1 = MD5.
  - \$2 = Blowfish.
  - \$2a= eksblowfish
  - \$5 =SHA-256
  - \$6 =SHA-512

(c) Prof.Miguel García Silvente

29

## Claves en linux (II)

xxx:\$6\$/0vUbQKa\$Eq2HbkKVz0EBs0KDDYAY88DujzK4T8PLr2Or9tggykTuS7EZL/UZ4UA5jkJfPnStEpCnPEhJdPRa0ChMnMBCs0:16540:0:99999:7:::

- Segundo campo identifica el “salt”: /0vUbQKa
- Para generar ese hash:

```
python -c 'import crypt; print crypt.crypt("1t0r0l0c0", "$6$/0vUbQKa")'
```

(c) Prof.Miguel García Silvente

30

## Rainbow tables (I)

- Tablas de códigos previamente generados usando diccionarios y reglas.
- Ejemplo: <http://project-rainbowcrack.com/>
- Código hash MD5: echo -n "1234" | md5sum
- Crack:  
v1.6: rcrack \*.rt -h 81dc9bdb52d04dc20036dbd8313ed055  
v1.7: rcrack . -h 81dc9bdb52d04dc20036dbd8313ed055
- Creación de tablas:  
rtgen md5 loweralpha 1 7 0 1000 1000 0  
rtgen md5 numeric 1 7 0 1000 1000 0  
rtsort \*.rt

(c) Prof.Miguel García Silvente

31

## Rainbow tables (II)

- Existen tablas ya generadas previamente:
  - <https://www.freerainbowtables.com> (gratuito)
  - <http://rainbowtables.shmoo.com/> (gratuito)
  - <http://www.rainbowtables.net/>
  - <http://www.rainbowcrack-online.com/>

(c) Prof.Miguel García Silvente

32

## Mejoras de seguridad

- Mejoras en el proceso de autenticación.
- Ejemplos:
  - 3 intentos fallidos: bloquear el usuario y/o la IP (fail2ban, denyhost)
  - Incorporando un retraso después de cada intento fallido.

### **Un niño bloquea el iPhone de su madre para los próximos 47 años**

- En el móvil apareció el mensaje “inhabilitado para los próximos 25 millones de minutos”

(c) Prof.Miguel García Silvente

33

## Métodos de autenticación (I)

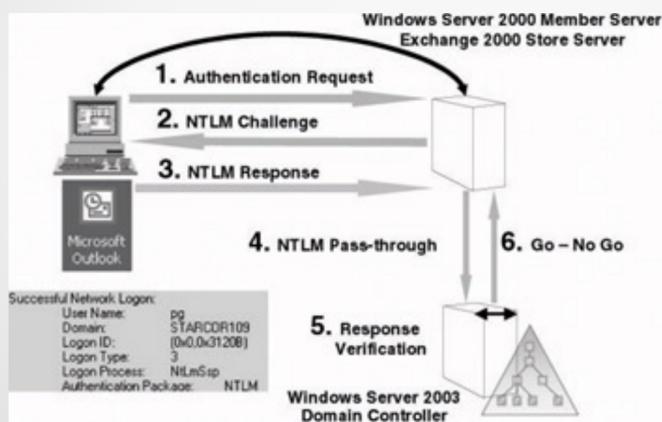
- Certificados digitales: pasaporte digital.
- Token digital: un pequeño dispositivo electrónico, usado un número adicional o una clave diferente cada vez.
- Tarjeta digital: usa un chip
- Kerberos: usa tickets (desarrollado por el MIT)
- LDAP (Lightweight Directory Access Protocol): directorio centralizado de datos y claves (no es adecuado para información muy cambiante)

(c) Prof.Miguel García Silvente

34

## Métodos de autenticación (II)

- NTLM (NT LAN Manager): usa autenticación reto/respuesta. Desarrollado por Microsoft.



(c) Prof.Miguel García Silvente

35

## Métodos de autenticación (III)

- Infraestructura de Clave Pública (PKI)
  - El usuario tiene una clave privada
  - Para autenticar se usa una clave pública.
  - Se encripta y desencripta usando ambas claves
- RADIUS: autenticación centralizada en red.
- Secure Socket Layer (SSL). Información encriptada a través de la red. También para TLS.
  - Ejemplo vulnerabilidad en openssl 1.0.1f: Heartbleed
- Secure Remote Password (SRP). Es un sistema PAKE (Password-Authenticated Key Agreement).

(c) Prof.Miguel García Silvente

36

## Fortalecimiento de la autenticación

- Biometría (FRR, FAR, EER)
- Tokens hardware.
- Autenticación multifactor (lo que tienes+lo que sabes+lo que eres+algo que haces+dónde estás)

## Biometría

- Usan características físicas o biológicas del cuerpo humano:
  - Asumen que la medida es única para una persona.
  - Iris, huella dactilar, reconocimiento de la voz, escáner de retina, reconocimiento de la cara, reconocimiento de la mano, ADN.
- ¿Se deberían recoger datos biométricos de los usuarios?

## Problemas con la biometría

- Cuestiones psicológicas:
  - Daños físicos
  - Pérdida de privacidad.
- Cuestiones tangibles:
  - Dispositivos caros.
  - No son 100% exactos.
  - Pueden fallar y paralizar el acceso.
  - Es posible la suplantación.

(c) Prof.Miguel García Silvente

39

## Autenticación múltiple

- Incrementar seguridad combinando claves con:
  - Limitaciones de acceso temporal.
  - Limitaciones geográficas de acceso.
- Los mecanismos adicionales pueden incrementar la incomodidad y la burocracia.
- Ejemplo: Autenticación basada en token temporal (un reloj sincronizado genera parte de la clave)

(c) Prof.Miguel García Silvente

40

## Correo electrónico seguro

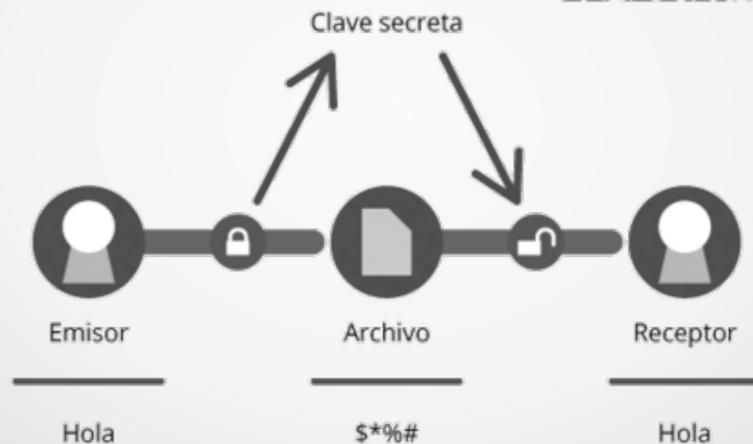
- Internet es una red abierta y por tanto no es segura.
- S/MIME (Secure Multipurpose Internet Mail Extensions)
- En los años 90 aparecen dos sistemas:
  - PEM: Private Enhancend Mail
  - PGP: Pretty Good Privacy
- Ejemplo: enigmail

## Índice

1. Autorización y control de acceso
- 2. Cifrado**
3. Seguridad de un sistema: SELinux
4. Malware
5. Simulación de sistemas

## Cifrado simétrico

GENBETA :dev



(c) Prof.Miguel García Silvente

43

## Cifrado simétrico: GPG

- Cifrado
  - `gpg -c datos.txt`

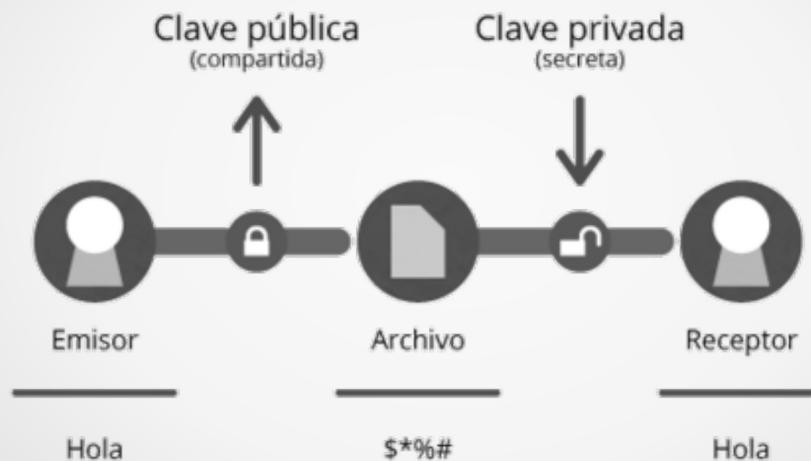
IMPORTANTE: el fichero datos.txt sigue existiendo!
- Descifrado
  - `gpg -d datos.txt.gpg`

(c) Prof.Miguel García Silvente

44

## Cifrado asimétrico

GENBETA:dev



(c) Prof.Miguel García Silvente

45

## Cifrado asimétrico: GnuPG (I)

- Generar claves:  
`gpg --generate-key`
- Comprobar claves:  
`gpg -k`
- Exportar clave pública:  
`gpg --output <fichero clave gpg> --export <ID usuario>`
- Subir clave pública a servidor de claves:  
`gpg --send-keys --keyserver pgp.mit.edu <ID usuario>`

(c) Prof.Miguel García Silvente

46

## Cifrado asimétrico: GnuPG (II)

- Importar clave pública:

```
gpg --import <fichero clave gpg>
```

```
gpg --receive-keys --keyserver pgp.mit.edu <ID usuario>
```

- Cifrar con la clave pública:

```
gpg --encrypt --recipient <ID usuario> <fichero>
```

- Descifrar con la clave privada:

```
gpg -d <fichero gpg> > <fichero>
```

- Firmar un fichero:

```
gpg -u <ID usuario> --output <fich. gpg> --sign <fichero>
```

## Cifrado simétrico: OpenSSL

- Encriptar:

```
openssl enc -aes-256-cbc -in <fichero> -out  
<fichero.enc>
```

- Desencriptar:

```
openssl enc -d -aes-256-cbc -in <fichero.enc> -out  
<fichero>
```

## Cifrado asimétrico: OpenSSL

- Generar certificado:

```
openssl req -x509 -nodes -days 365 -newkey rsa:1024 -  
keyout mycert.pem -out mycert.pem
```

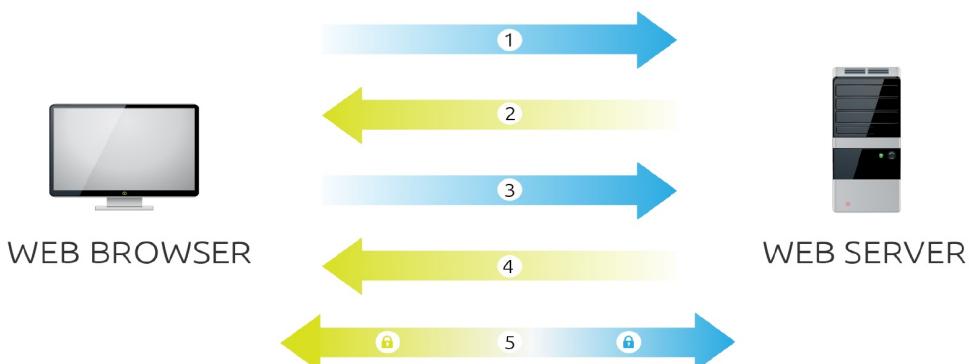
- Encriptar:

```
openssl smime -encrypt -binary -aes-256-cbc -in <fichero>  
-out <fichero.enc> -outform DER mycert.pem
```

- Desencriptar:

```
openssl smime -decrypt -binary -in <fichero.enc> -inform  
DER -out <fichero> -inkey <private.key> -passin  
pass:<password>
```

## Encriptación en comunicación web (I)



## Encriptación en comunicación web (II)

1. El navegador se conecta al servidor web con ssl y le solicita que se identifique.
2. El servidor envía una copia de su certificado SSL, incluyendo su clave pública.
3. El navegador comprueba el certificado raíz contra una lista de CAs de confianza, así como que no ha expirado y que no ha sido revocado y que su "common name" es válido para ese sitio. Si es todo correcto, crea una clave, la encripta usando la clave pública del servidor y la envía al servidor.
4. El servidor desencripta la clave simétrica de sesión usando su clave privada y devuelve el reconocimiento de conexión usando la clave simétrica de sesión.
5. Todos los datos transmitidos entre cliente y servidor se encriptan con la clave de sesión.