

# Autenticación local

## Seguridad en Redes de Ordenadores

Enrique Soriano

LS, GSYC

9 de febrero de 2018



(cc) 2018 Grupo de Sistemas y Comunicaciones.

Algunos derechos reservados. Este trabajo se entrega bajo la licencia Creative Commons Reconocimiento - NoComercial - SinObraDerivada (by-nc-nd). Para obtener la licencia completa, véase <http://creativecommons.org/licenses> También puede solicitarse a Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

- ▶ **Identificación:** Alguien dice ser un sujeto concreto.
- ▶ **Autenticación:** Alguien demuestra su identidad.
- ▶ **Autorización:** Otorgamiento de privilegios a un sujeto para realizar una operación de acceso sobre uno o varios objetos.
- ▶ **Control de acceso:** Permitir o denegar una operación de acceso sobre un recurso en base a la autorización del sujeto.

# Autenticación

- ▶ Un programa se autentica ante el sistema a nombre de un usuario y pasa a ejecutar a su nombre.
- ▶ ¿Cómo te autenticas?
  - ▶ Algo que conoces.
  - ▶ Algo que tienes.
  - ▶ Algo que eres.
  - ▶ Algo que haces.
  - ▶ Donde estás.

- ▶ Idea general: combinar más de un método de autenticación. Por lo general, se usa la contraseña junto con otro método de autenticación adicional (two-factor authentication).
- ▶ Además de requerir la contraseña, usar un método de autenticación adicional.
- ▶ Puede ser:
  - ▶ Una contraseña adicional, p. ej. enviada por SMS.
  - ▶ Un Time-based One-time Password (TOTP), p. ej. Google Authenticator.
  - ▶ Un servicio extra para activar/desactivar la cuenta, p. ej. Latch.
  - ▶ ...

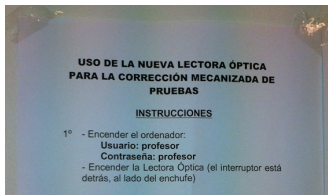
# Contraseñas

- ▶ La forma más común y sencilla de autenticación: el usuario proporciona un secreto que sólo él y el sistema conocen.
- ▶ Peligro: depende del usuario.
- ▶ Riesgos:
  - ▶ Que vean nuestra contraseña.
  - ▶ Que la adivinen.
  - ▶ Spoofing (suplantación).
  - ▶ Que se hagan con el almacén de contraseñas.

# Contraseñas

Que vean nuestra contraseña:

- ▶ No se deben compartir.
- ▶ No se deben apuntar en claro.
- ▶ No se debe usar la misma contraseña para distintos servicios.
- ▶ Aplicaciones *keyring*: Password Safe (Windows), Keychain (OSX), etc.



Que adivinen nuestra contraseña:

- ▶ Ataque de fuerza bruta.
- ▶ Búsqueda inteligente.
- ▶ Ataque de diccionario online.



# Ataque de fuerza bruta

Keyspace:

- ▶ Siendo  $S$  la longitud de la contraseña y  $N$  el número de símbolos que se pueden usar en las contraseñas.

$$Keyspace = \sum_{i=0}^S N^i$$

- ▶ Ejemplo: Contraseñas de sólo minúsculas y de 0 a 8 caracteres:

$$\sum_{i=0}^8 24^i = 114\,861\,197\,401$$

- ▶ Ejemplo: Con mayúsculas, minúsculas, números y símbolos de puntuación, de 0 a 14 caracteres:

$$\sum_{i=0}^{14} (24 + 24 + 10 + 14)^i = 102\,030\,284\,526\,887\,192\,558\,589\,577$$

# Contraseñas: entropía

$$\text{Entropy}(\text{password}) = \log_2(S^L)$$

- ▶  $S$  es el número de símbolos que se utilizan.
- ▶  $L$  es la longitud de la contraseña.
- ▶ Se usa como medida de la fortaleza de una contraseña.
- ▶ Es la entropía teórica, la que se calcula para una **contraseña aleatoria**.

# Ataque inteligente

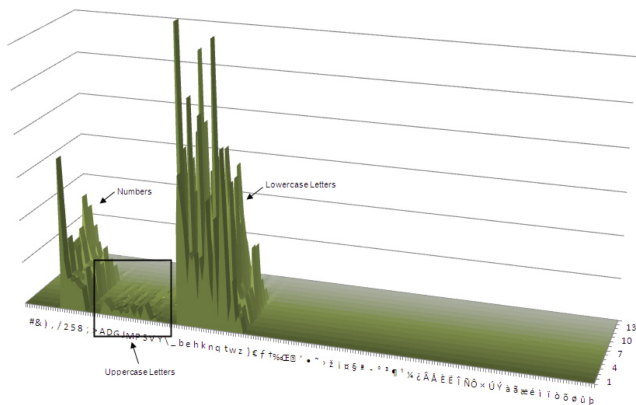


Imagen (C) xato.net

# Contraseñas

Una contraseña puede tener mucha entropía y ser insegura a la vez, p. ej:

- ▶ En un ataque, lo primero que se hace es probar con un diccionario con las contraseñas populares:



Contraseñas de calidad:

- ▶ Nnemotécnicos: TabletMipefaesstwa3 de *“Mi película favorita es star wars 3”*
- ▶ Rimas: SEGURIDAD-0-calamidad@casa
- ▶ Repetición: hey.h0.HEY.H0.6

# Contraseñas

Distintas contraseñas para distintos servicios:

- ▶ Se puede tener un método para generar las contraseñas para distintos servicios.
- ▶ Por ejemplo, poner al principio el número de caracteres del servicio o servidor:
  - ▶ Amazon: A6m1--mej0r--SECRETO
  - ▶ Msn: M3m1--mej0r--SECRETO
  - ▶ Apple: A5m1--mej0r--SECRETO

## Medidas de protección:

- ▶ No permitir contraseñas vacías en el sistema.
- ▶ Obligar a cambiar las contraseñas por omisión.  
Ejemplo: routers adsl.
- ▶ Obligar a que tengan una longitud mínima.
- ▶ Obligar a que no sean palabras del diccionario (o combinaciones básicas): requerimos puntos, números, mayúsculas y minúsculas, etc.

## Medidas de protección:

- ▶ Pasar *crackers* para validarlas.  
Ejemplo: *John the Ripper*, *HashCat*, *Ophcrack*, *Cain/Abel*....
- ▶ **No** es buena idea generar contraseñas aleatorias para los usuarios → las apuntan. Si son aleatorias, no deben ser predecibles.
- ▶ ¿Cambiar las contraseñas periódicamente? ¿Cada cuánto? Lo que importa es tener una password de calidad.



Medidas de protección:

- ▶ Limitar el número de intentos de autenticación.
- ▶ Aumentar el tiempo entre autenticaciones fallidas.
- ▶ *Captchas* para evitar ataques automatizados.
- ▶ **Compromiso: seguridad vs. obstrucción.**

Más riesgos: que nos timen.

- ▶ Spoofing
- ▶ Keyloggers
- ▶ Ingeniería social
- ▶ ...

## Spoofing:

- ▶ Un atacante reemplaza al sujeto ante el que nos autenticamos, y nos roba la contraseña.
- ▶ Seguramente no nos damos cuenta: el resultado de la autenticación parece un error al escribir la contraseña, pero no lo es.
- ▶ Ataque clásico: reemplazar `/bin/login`
- ▶ En el WWW: te conectas a una página que no es la que crees (p. ej. te engañan con *phishing*).

## Medidas de protección:

- ▶ Imprimir el número de intentos reales que se llevan.
- ▶ Trusted Path: estar seguros de que el usuario se comunica con el sistema operativo y no con un programa falso.
- ▶ Autenticación mutua: el programa de login se autentica ante el usuario antes de que el usuario se identifique y autentique.

## Keyloggers:

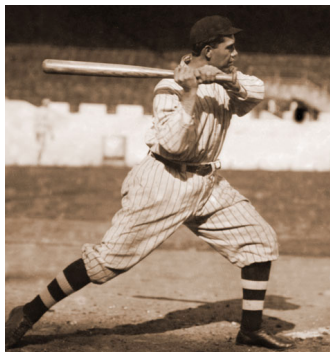
- ▶ Software: un programa captura todo lo que tecleamos.
- ▶ Hardware: espía el cable del teclado o la transmisión inalámbrica de ratones/teclados. Aprox. 50 Euros, 2Gb. Los hay WiFi.





Moraleja:

- ▶ The Big Stick Principle.



Si se hacen con el almacén de contraseñas:

- ▶ Las contraseñas se suelen guardar cifradas.
- ▶ Por lo general, se guarda un hash de la contraseña.
- ▶ Ataque de diccionario offline: mucho más efectivo que online.

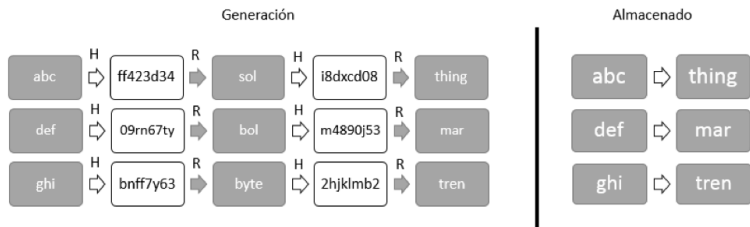


## Ataque de diccionario offline:

- ▶ Buscar la hash que se quiere romper en una tabla precalculada de tuplas (contraseña, hash) → cambias tiempo de computación por espacio.
- ▶ *Rainbow-tables* (p. ej. ophcrack).
- ▶ Uso de Google para encontrar la hash. ¿Qué contraseña tiene esta hash SHA-1?  
99800b85d3383e3a2fb45eb7d0066a4879a9dad0  
¡Búscalos en Google!

# Rainbow Tables

Compromiso espacio/cómputo:



## Medidas de protección:

- ▶ Contraseñas buenas: aumenta el keyspace objetivo de la tabla.
- ▶ Hashes con salt: necesitas una tabla a medida para romper una contraseña específica.
- ▶ Mejor: *peppering* consiste en usar como salt una clave secreta.
- ▶ Todavía mejor: hacer una HMAC de las passwords en lugar de hacer una hash y que la clave esté en el componente que autentica al usuario. En el peor de los casos, es igual de seguro que una hash.
- ▶ Y todavía mejor: almacenar la clave de la HMAC en un HSM si te puedes permitir hardware dedicado.

# Caso: autenticación clásica en Unix

Pasos de arranque:

1. Scripts de arranque (`init`).
2. `/bin/getty`
3. `/bin/login`

# Caso: credenciales en Unix

Un proceso tiene *credenciales*<sup>1</sup>:

- ▶ PID: identificador de proceso.
- ▶ UID: identificador de usuario.
- ▶ GID: identificador de grupo.
- ▶ EUID: identificador de usuario efectivo, el que se usa para comprobar privilegios.
- ▶ EGID: identificador de grupo efectivo, el que se usa para comprobar privilegios.
- ▶ ...

---

<sup>1</sup>Dependen del “sabor” de Unix

# Caso: autenticación básica en Unix

/etc/passwd:

- ▶ Login.
- ▶ Contraseña (“x” cuando tenemos *shadow*)
- ▶ UID.
- ▶ GID.
- ▶ Gecos info: datos sobre el usuario, separados por comas.
- ▶ Path del home.
- ▶ Programa de inicio.

# Caso: autenticación básica en Unix

/etc/shadow:

- ▶ Login.
- ▶ Hash: son tres campos separados por \$.
  - ▶ Algoritmo: p. ej. 1 es MD6, 6 es SHA-512.
  - ▶ Salt
  - ▶ Hash

NOTA: Si tiene un valor que no se corresponde con una posible salida de crypt, el usuario no se podrá autenticar con contraseña (pero sí de otras formas). Por ejemplo, se pone \* o ! para evitar que el usuario entre con contraseña. Si está vacío, el usuario no tiene contraseña (no es recomendable). En Linux recientes, la contraseña de root es !.

# Caso: autenticación básica en Unix

`/etc/shadow` (continúa):

- ▶ Fecha Unix del último cambio de contraseña.
- ▶ Mínima edad: mínimo número de días para permitir un cambio de contraseña.
- ▶ Máxima edad: días tras los que tiene que cambiar la contraseña.
- ▶ Días, antes de la caducidad según el campo anterior, para avisar al usuario.
- ▶ Días de prórroga desde que caduca, cuando pasan se anula la contraseña.



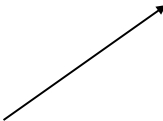
# Caso: autenticación básica en Unix

`/etc/shadow` (continúa):

- ▶ Fecha Unix de caducidad para la cuenta. Si llega, se cancela la cuenta. No es lo mismo que la edad de la contraseña, esta es más dura.
- ▶ Campo reservado para uso futuro.

# Caso: control de credenciales en Unix

El uid y gid de root es 0



- ▶ `/bin/id`: Permite ver tu UID y GID.
- ▶ `/bin/su`: Permite ejecutar un shell con otro UID. Por omisión, intenta ejecutar un shell con UID 0.
- ▶ `/usr/bin/sudo`: Permite ejecutar un comando con otro UID, proporcionando tu propia contraseña. El fichero `/etc/sudoers` especifica quién puede convertirse en quién, y para qué.

# Caso: control de credenciales en Unix

Ejemplo de sudoers (I):

```
jose ALL = (root, bin : operator, system) ALL
```

- ▶ jose puede, en cualquier máquina
- ▶ adquirir el UID de root y bin
- ▶ adquirir el GID de operator y system
- ▶ para ejecutar cualquier comando

# Caso: control de credenciales en Unix

Ejemplo de sudoers (II):

```
ramon mono = NOPASSWD: /bin/kill, PASSWD: /bin/ls,  
/usr/bin/lprm
```

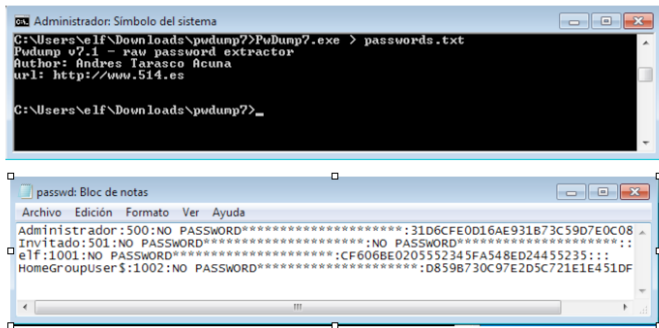
- ▶ ramon puede, en la máquina mono
- ▶ adquirir el UID de root
- ▶ para ejecutar kill sin proporcionar contraseña
- ▶ para ejecutar ls y lprm proporcionando contraseña

# Caso: autenticación en Windows 7

- ▶ Windows almacena las contraseñas en SAM:  
`\windows\system32\config\sam`
- ▶ El fichero está bloqueado en todo momento.
- ▶ Puede contener dos tipos de contraseñas cifradas:
  - ▶ LM (Lan Manager). ¡Inseguro! Windows 7 no la usa por omisión (pero se puede configurar para que las use, ojo).
  - ▶ NTLM (NT Lan Manager): No tan seguro como debería.

# Caso: autenticación en Windows 7

En un terminal (cmd.exe) como administrador:



The screenshot displays two windows on a Windows 7 desktop. The top window is a command prompt titled 'Administrador: Símbolo del sistema'. It shows the execution of 'PuDump? .exe' in the directory 'C:\Users\elf\Downloads', which outputs version information and the file path 'passwords.txt'. The bottom window is a Notepad application titled 'passwd: Bloc de notas', displaying the contents of the password file. The file lists three users: 'Administrador' with a long hexadecimal password, 'Invitado' with a long hexadecimal password, and 'elf' with a long hexadecimal password. The 'HomeGroupUser\$' entry is also visible.

```
Administrador: Símbolo del sistema
C:\Users\elf\Downloads\PuDump?>PuDump?.exe > passwords.txt
PuDump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es

C:\Users\elf\Downloads\PuDump?>_
```

```
passwd: Bloc de notas
Administrador:500:NO PASSWORD*****:31D6CFE0D16AE931873C59D7E0C08
Invitado:501:NO PASSWORD*****:NO PASSWORD*****:
elf:1001:NO PASSWORD*****:CF606BE020552345FA548ED24455235:::
HomeGroupUser$:1002:NO PASSWORD*****:D859B730C97E2D5C721E1E451DF
```

# Caso: autenticación en Windows 7

## Contraseñas LM (Lan Manager):

- ▶ Convierte todos los caracteres de la contraseña a mayúsculas.
- ▶ Rellena con 0s la contraseña hasta llegar a los 14 caracteres.
- ▶ Parte la contraseña en dos bloques de 7 bytes.
- ▶ Usa los dos bloques como claves para cifrar con DES para un bloque de datos conocido: 0x4b47532140232425
- ▶ Concatena el resultado de los dos cifrados DES.

# Caso: autenticación en Windows 7

Contraseñas LM (Lan Manager):

Input interpretation:

$$\left( \sum_{i=0}^7 (24 + 10 + 14)^i \right) \times 2$$

---

Result:

1 199 118 316 130

VS

$$\sum_{i=0}^{14} (24 + 10 + 14)^i = 351\,982\,201\,019\,228\,060\,039\,473$$



# Caso: autenticación en Windows 7

## Contraseñas NTLM (NT Lan Manager):

- ▶ Usa una hash MD4 de 128 bits, que era segura en 1991:
  - ▶ Ataque de colisión en microsegundos.
  - ▶ Ataque de pre-imagen teórico.
- ▶ Diferencia entre mayúsculas y minúsculas.
- ▶ **No usa salt.**

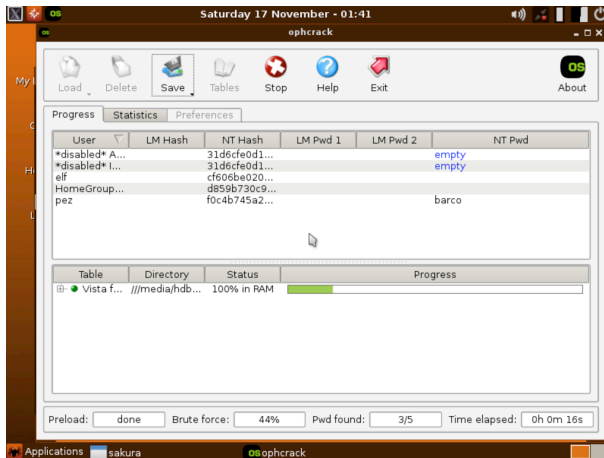
# Caso: autenticación en Windows 7

Syskey:

- ▶ Añade una capa extra de seguridad cifrando el almacén de claves de SAM.
- ▶ Opción I (activada por omisión):
  - ▶ Usa una clave generada para cada sistema y que está oculta en el Registro.
  - ▶ Viola el principio de Kerckhoffs: no sirve de nada.
  - ▶ `bkhive` la encuentra, `samdump2` descifra el archivo de SAM.
- ▶ Opción II: Generar la clave a partir de una contraseña adicional que se introduce en el arranque.
- ▶ Opción III: Almacenar la clave en un disco extraíble.

# Caso: autenticación en Windows 7

Ophcrack rompe contraseñas débiles de Windows NT (hashes MD4) con rainbow tables:



# Caso: control de credenciales en Windows

En Windows es común usar una cuenta de administrador. User Account Control (UAC) permite controlar los privilegios:

- ▶ Un administrador tiene un token de usuario y token de administrador.
- ▶ Idea: evitar el uso del token de administrador para operaciones que no lo requieren.
- ▶ Cuando se intenta usar el de administrador (elevación) se presenta una ventana de diálogo UAC para pedir permiso o credenciales.

# Caso: control de credenciales en Windows

Los diálogos de elevación usan un código de colores:

- ▶ Rojo (con escudo) si el firmante de la aplicación está bloqueado, es una aplicación sin firma bajada de Internet.
- ▶ Azul (con escudo) si es una aplicación administrativa de Windows (p.ej. Panel de Control).
- ▶ Azul (con interrogación) si la aplicación está firmada por terceros con Authenticode y se puede verificar.
- ▶ Amarillo (con escudo) si la aplicación no está firmada o está firmada pero no se puede verificar.

# Caso: control de credenciales en Windows



# Caso: control de credenciales en Windows

## UAC niveles

- ▶ Sólo el nivel más alto impide autoelevación silenciosa.
- ▶ Los niveles más bajos no usan el escritorio seguro.

# Caso: control de credenciales en Windows

## Ventajas de UAC:

- ▶ No hay que usar dos cuentas (usuario y administrador).
- ▶ No hay que entender para qué hay que usar la cuenta de administrador y para qué la otra.
- ▶ No hay que gestionar dos perfiles distintos (carpetas, escritorios, etc.).
- ▶ Aplica el *principio de mínimo privilegio*.

Inconvenientes: No ha sido bien recibido por ser molesto ¿El problema es la interfaz?



# Algo que eres

Nuestro cuerpo nos puede autenticar (biometría):

- ▶ Escaner de iris.
- ▶ Escaner de retina.
- ▶ Huellas dactilares.
- ▶ Palma de la mano.
- ▶ Reconocimiento de voz.
- ▶ Reconocimiento facial.

# Algo que eres

- ▶ Aumenta el riesgo para la integridad física del usuario.
- ▶ Son más propensos a fallar:
  - ▶ Falso positivo: autenticación errónea.
  - ▶ Falso negativo: aumenta la obstrucción.
- ▶ ¿Cómo se revoca una huella dactilar o una cara?

# Algo que tienes

- ▶ **Tu SmartPhone: Google Authenticator.**
- ▶ **SecureID**
- ▶ **Smart Cards**
- ▶ **RFID / NFC**
- ▶ **USB tokens**
- ▶ ...

# Algo que tienes

Los dispositivos de autenticación:

- ▶ Pueden disminuir la obstrucción, como NFC.
- ▶ Pueden aumentar la obstrucción, como SecureID.
- ▶ Hay que fiarse del HW lector: igual que con las contraseñas, puede existir spoofing.
- ▶ Nuevo riesgo: puedes perder el objeto, o te lo pueden robar.
- ▶ Si lo piensas, corremos los mismos riesgos a diario con las llaves de nuestra casa o las tarjetas de crédito.

Por ejemplo, el DNI electrónico (DNle):

- ▶ Se usa para cualquier tipo de tramitación telemática con el estado.
- ▶ Tiene la misma validez que el DNI normal.
- ▶ El DNle 3.0 incorpora NFC.

El DNI electrónico (DNle) es una SmartCard *tamper-resistant* con los siguientes ficheros dentro:

- ▶ Datos de filiación del titular
- ▶ Imagen digitalizada de la fotografía.
- ▶ Imagen digitalizada de la firma manuscrita.
- ▶ Plantilla de la impresión dactilar de los dedos índice de cada mano.
- ▶ Certificado X.509 para autenticación, y la clave privada correspondiente.
- ▶ Certificado X.509 para firma, y la clave privada correspondiente.
- ▶ Certificado X.509 de la autoridad emisora.

La generación de claves, el cifrado/descifrado, la firma, etc. se realiza en el chip de la tarjeta.

- ▶ Se entrega junto con un PIN (sic) (es una contraseña con 8-16 caracteres): algo que tienes + algo que sabes.
- ▶ Se puede cambiar la contraseña desde cualquier PC si sabemos la contraseña vieja.
- ▶ Se puede reestablecer la contraseña (sin conocer el viejo) en un terminal especial (Punto de Actualización del DNle). Es necesario proporcionar la huella dactilar (la verificación se hace en el propio chip): algo que tienes + algo que eres.



- ▶ Algo que haces: por ejemplo, gestos ante una cámara, en un touchpad, etc.
- ▶ Dónde estás: el simple hecho de encontrarte en una localización física (p.ej., la sala de los servidores) te autentica.



# Combinación de métodos de autenticación

## SUN's PAM (Pluggable Authentication Modules):

- ▶ Objetivos:
  - ▶ Combinar distintos métodos de autenticación proporcionados por distintos módulos independientes.
  - ▶ Usar distintos métodos de autenticación sin modificar el código de las aplicaciones.
  - ▶ Modificar la autenticación sin parar el sistema.
- ▶ PAM está disponible para distintos *sabores* de Unix.

Seis primitivas englobadas en cuatro *grupos*:

- ▶ Auth: tareas para autenticar al usuario.
- ▶ Account: tareas de gestión de los datos de una cuenta.
- ▶ Password: tareas para administrar los mecanismos de la autenticación.
- ▶ Session: tareas a realizar al crear y destruir sesiones.

## Auth:

- ▶ `pam_authenticate`: sirve para autenticar al sujeto.
- ▶ `pam_setcred`: sirve para establecer y gestionar las credenciales. Tiene que llamarse después de `pam_authenticate` y antes de establecer la sesión.

## Session:

- ▶ `pam_open_session`: realiza las tareas asociadas con el inicio de sesión. Por ejemplo, actualizar logs de entrada, etc.
- ▶ `pam_close_session`: se encarga de las tareas asociadas con el fin de sesión. Por ejemplo, actualizar logs de salida, etc.

## Account:

- ▶ `pam_acct_mgmt`: verifica si la cuenta está en buenas condiciones. Por ejemplo si la contraseña caducó, si la cuenta está cancelada, etc.

Password:

- ▶ `pam_chauthtok`: cambia el objeto de la autenticación (p.ej., la contraseña), posiblemente comprobando que es suficientemente bueno, etc.

- ▶ En `/etc/pam.d/` se crean ficheros con las políticas para los distintos servicios.
- ▶ Un fichero de políticas define una pila de reglas que determina el éxito de una operación:

*type control module-path module-arguments*

- ▶ `type`: grupo (auth, account, password, session).
- ▶ `control`: cómo afecta al resultado de la operación.
- ▶ `module-path`: módulo.
- ▶ `module-arguments`: argumentos.

## Control:

- ▶ `requisite`: si la operación retorna error, se acaba la autenticación con fallo.
- ▶ `required`: si retorna error, el flujo continua aunque el resultado final ya está decidido: será un fallo.
- ▶ `sufficient`: si retorna éxito, termina inmediatamente la con éxito. Si retornar error, se sigue llamando al resto (y si el resto funcionan, la operación acaba con éxito).
- ▶ `optional` se ejecuta, pero lo que retorne (éxito o fallo) no se tiene en cuenta.



# Linux-PAM: ejemplo

