

Bastionado de redes y sistemas

Tarea 3: Configuración de sistemas de acceso y autenticación de personas

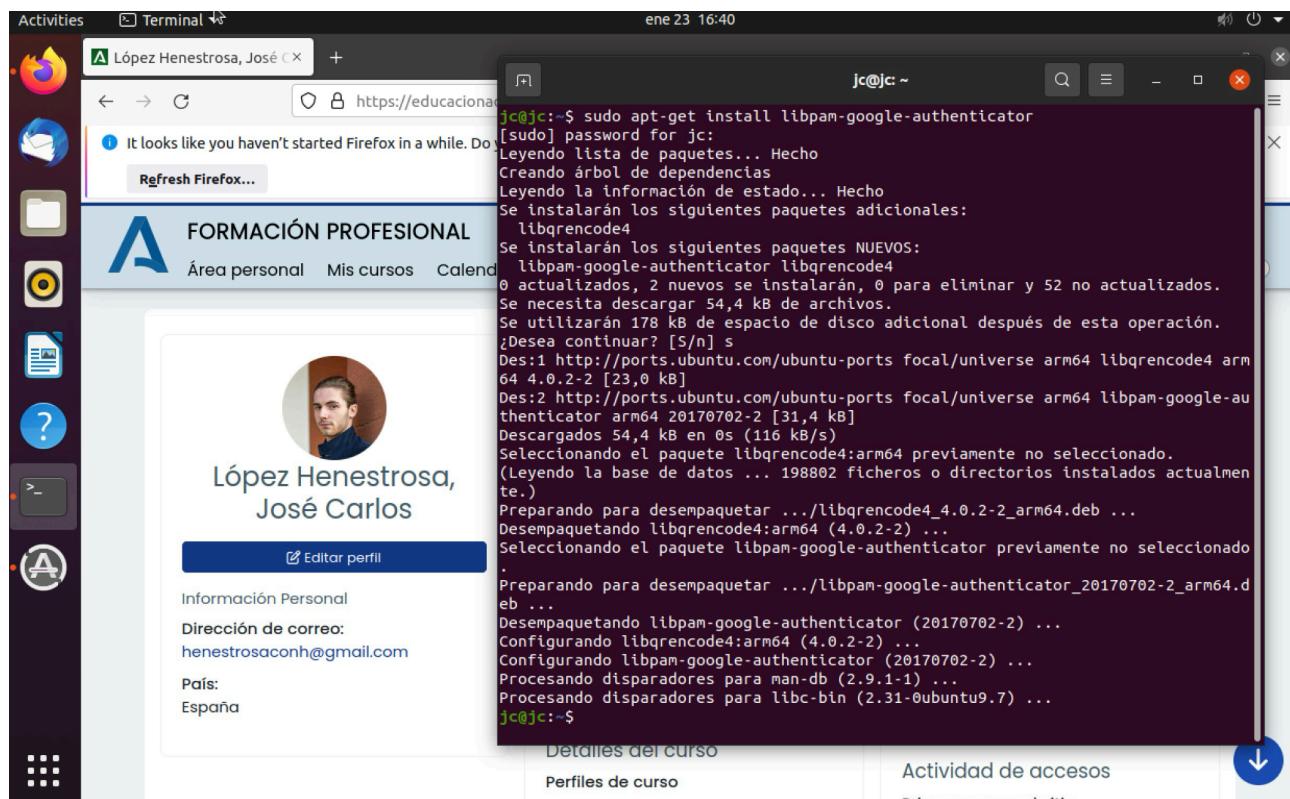
Índice

Instalar libpam-google-authenticator.....	2
Login desde terminal de texto.....	6
Login en entorno gráfico.....	8
Uso del comando sudo.....	11
Preguntas adicionales.....	13
¿Por qué influye el orden en el que se añaden las directivas en el archivo PAM que se está configurando?.....	13
¿En qué otras acciones o servicios sería aconsejable configurar PAM para reforzar la autenticación del usuario?.....	15
¿Cuál es la finalidad de los archivos common-account, common-password, common-auth y common-session?.....	16
¿Por qué se genera un código semilla al ejecutar el comando google-authenticator y cuál es su propósito en la autenticación?.....	17
¿Por qué se proporcionan una serie de códigos preestablecidos al usuario una vez se ha generado el QR y cuál es su finalidad?.....	18
¿En qué ubicación se almacena la configuración de Google Authenticator y los códigos preestablecidos para el usuario? Explica cómo se pueden usar estos códigos en caso de no disponer de la aplicación Google Authenticator, especificando que cada código solo se puede utilizar una vez.....	19

Instalar libpam-google-authenticator

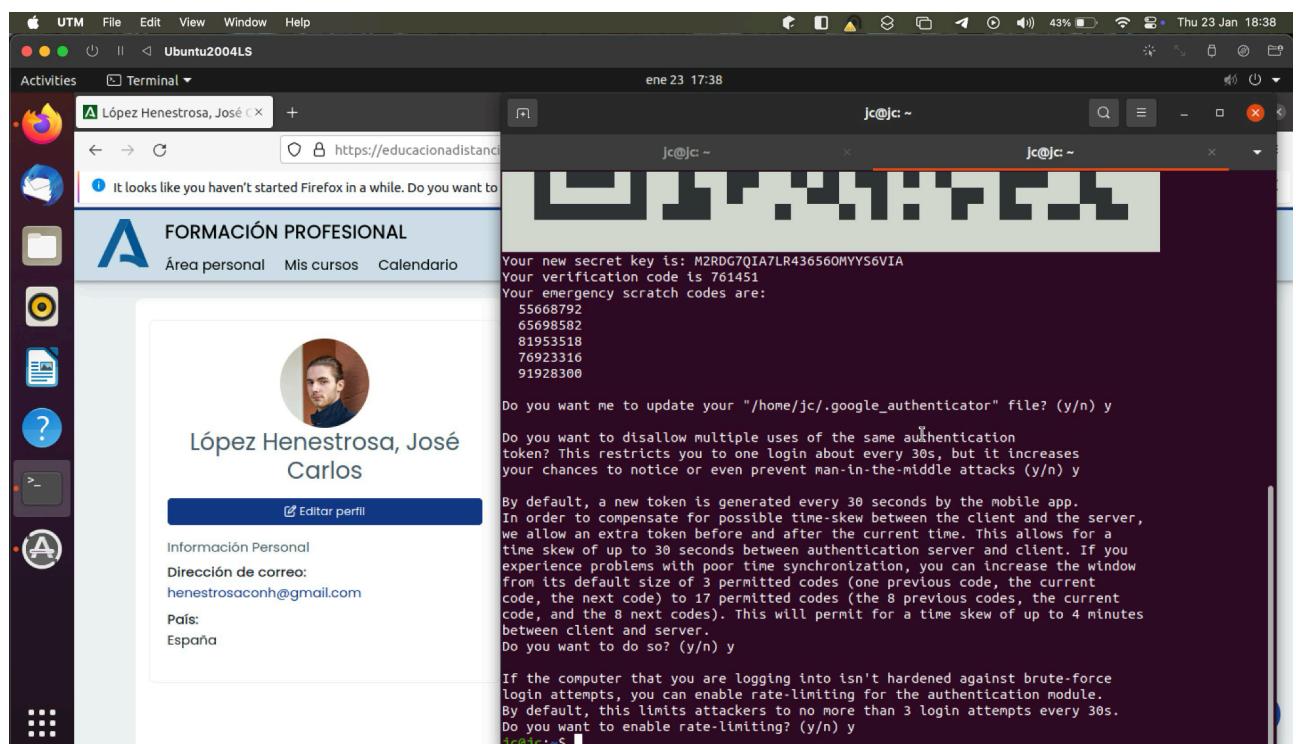
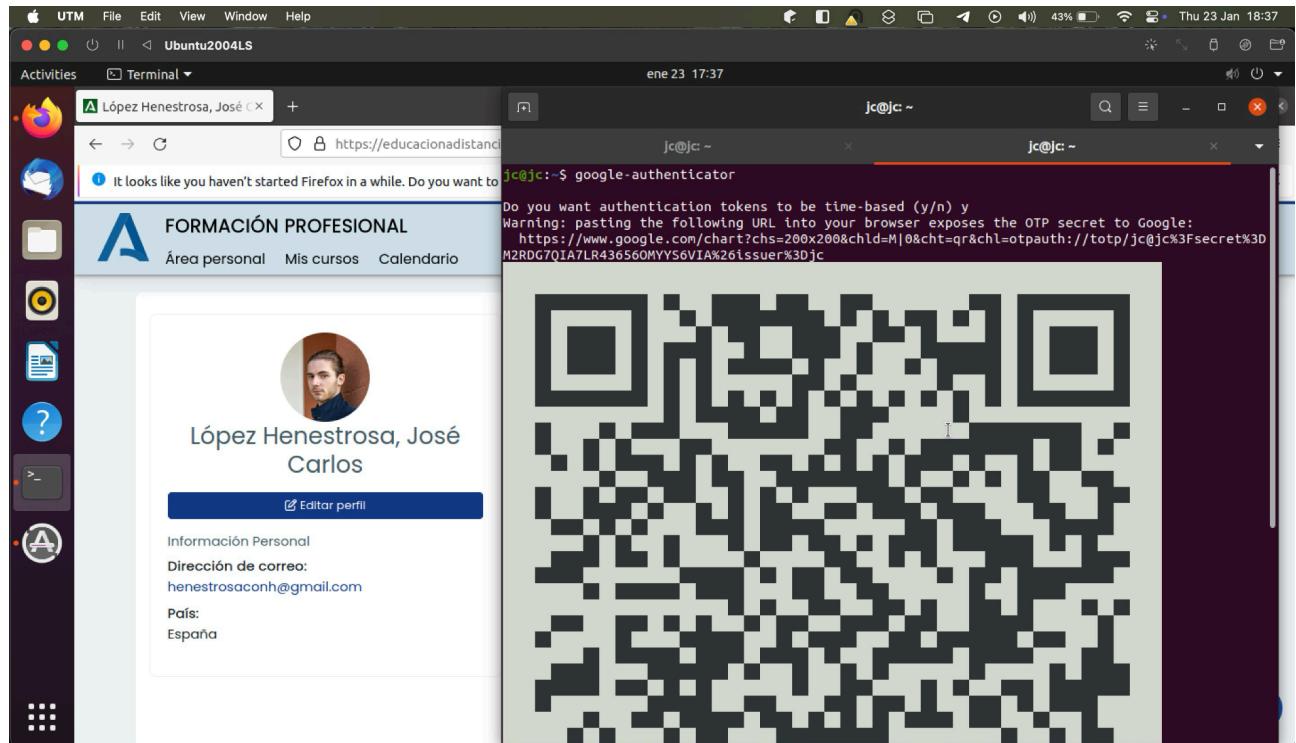
Antes de empezar, es necesario instalar el paquete `libpam-google-authenticator`. Para ello, lo instalamos ejecutando el siguiente comando:

```
sudo apt-get install libpam-google-authenticator
```



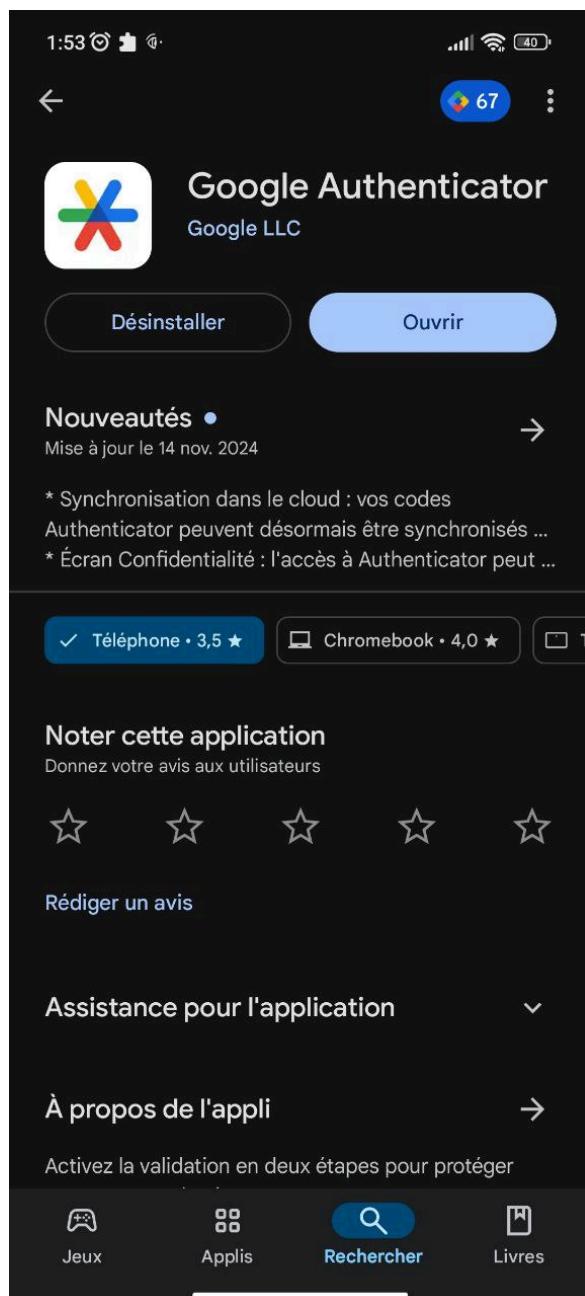
Instalando el paquete `libpam-google-authenticator`

Una vez hecho esto, configuramos Google Authenticator para el usuario actual ejecutando `google-authenticator`.



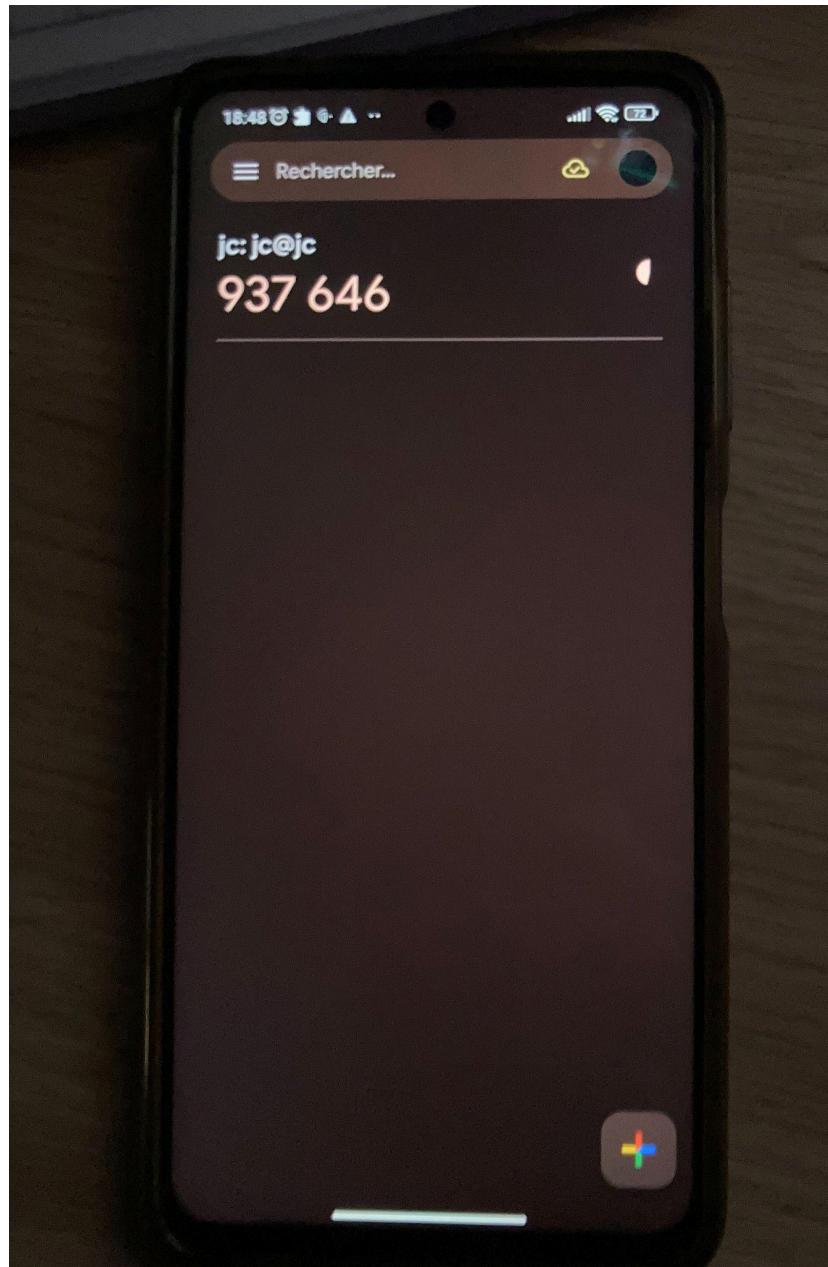
Configurando Google Authenticator

Para habilitar la autenticación 2FA, necesitamos escanear el QR generado en un dispositivo móvil para tener acceso a los códigos de verificación. Para ello, instalamos la aplicación Google Authenticator. Como tengo un dispositivo móvil Android, la descargaré desde la [Play Store](#).



Página de la app de Google Authenticator en la Play Store.
Mi dispositivo está en francés.

A continuación, accedemos a la aplicación y escaneamos el QR. Al hacerlo, nos aparecerá la siguiente pantalla con los códigos de verificación correspondientes a nuestro equipo:



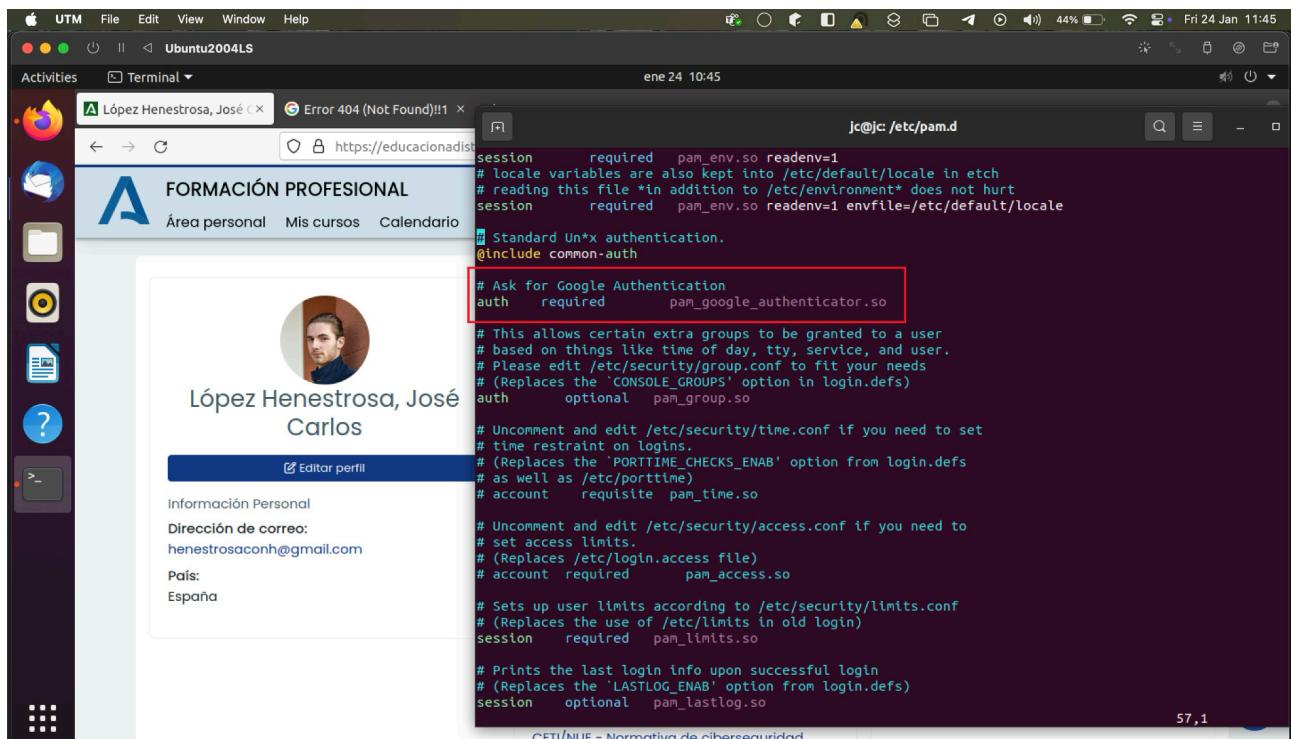
Pantalla principal de la app de Google Authenticator tras escanear el QR. La aplicación no permite capturar la pantalla, por lo que tuve que tomar la imagen desde otro dispositivo móvil.

Login desde terminal de texto

Configurando el archivo `/etc/pam.d/login` para que el sistema pida un código de autenticación cuando se intente iniciar sesión en una terminal sin entorno gráfico.

En primer lugar, tenemos que editar el archivo `/etc/pam.d/login` para añadir la siguiente línea tras incluir los contenidos del archivo `common-auth`:

```
auth      required      pam_google_authenticator.so
```



```
session      required      pam_env.so      readenv=1
# locale variables are also kept into /etc/default/locale in etch
# reading this file *in addition to /etc/environment* does not hurt
session      required      pam_env.so      readenv=1      envfile=/etc/default/locale

# Standard Unix authentication.
@include common-auth

# Ask for Google Authentication
auth      required      pam_google_authenticator.so

# This allows certain extra groups to be granted to a user
# based on things like time of day, tty, service, and user.
# Please edit /etc/security/group.conf to fit your needs
# (Replaces the 'CONSOLE_GROUPS' option in login.defs)
auth      optional      pam_group.so

# Uncomment and edit /etc/security/time.conf if you need to set
# time restraint on logins.
# (Replaces the 'PORTTIME_CHECKS_ENAB' option from login.defs)
# as well as /etc/porttime
# account      requisite      pam_time.so

# Uncomment and edit /etc/security/access.conf if you need to set
# access limits.
# (Replaces /etc/login.access file)
# account      required      pam_access.so

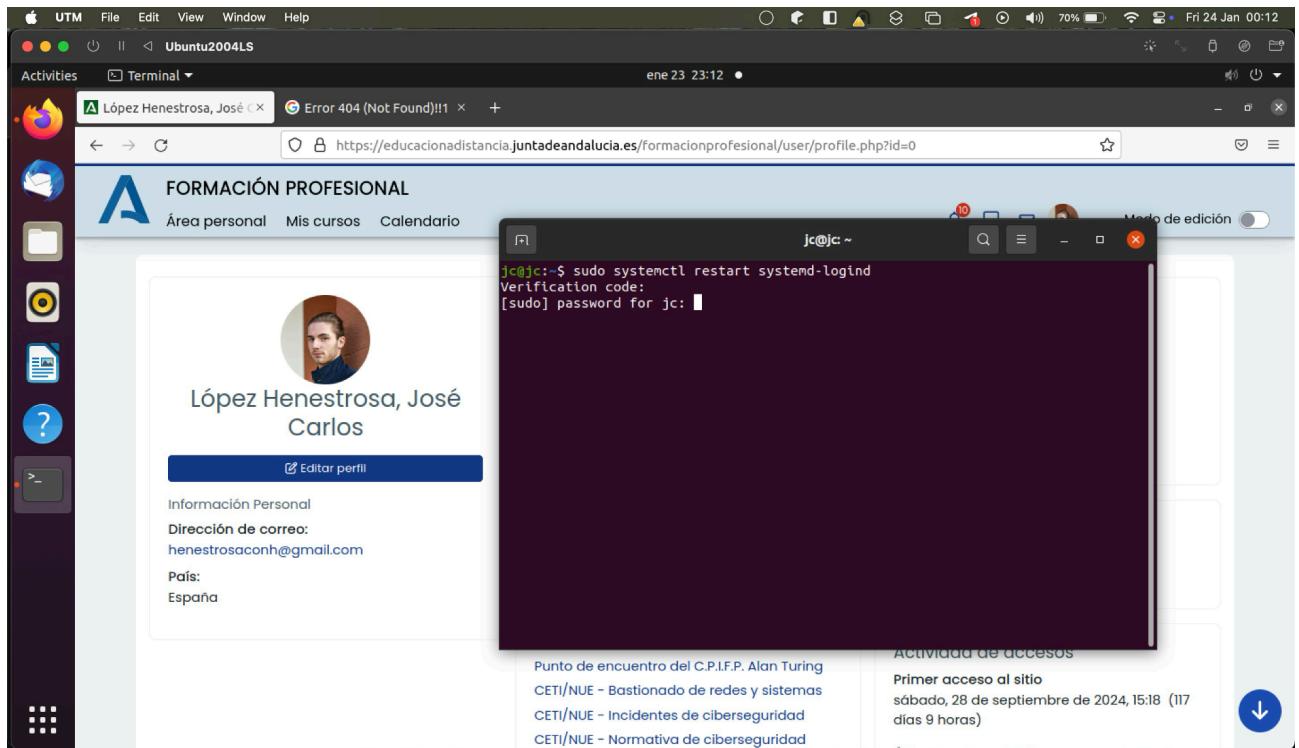
# Sets up user limits according to /etc/security/limits.conf
# (Replaces the use of /etc/limits in old login)
session      required      pam_limits.so

# Prints the last login info upon successful login
# (Replaces the 'LASTLOG_ENAB' option from login.defs)
session      optional      pam_lastlog.so
```

Archivo `/etc/pam.d/login` con los cambios marcados en rojo

Esto le indica al sistema que, además de la contraseña, se debe pedir el código TOTP (Time-based One-Time Password) de Google Authenticator como un segundo factor.

Para comprobar que funciona correctamente, reiniciamos el servicio `systemd-logind` con el comando `sudo systemctl restart systemd-logind` para que se vuelvan a pedir las credenciales en la terminal y se pida el código de verificación.



Reiniciando el servicio `systemd-logind`

Como podemos apreciar, se nos pide el código de verificación de Google Authenticator correctamente para poder iniciar sesión con nuestro usuario desde la terminal, por lo que la configuración se ha aplicado con éxito.

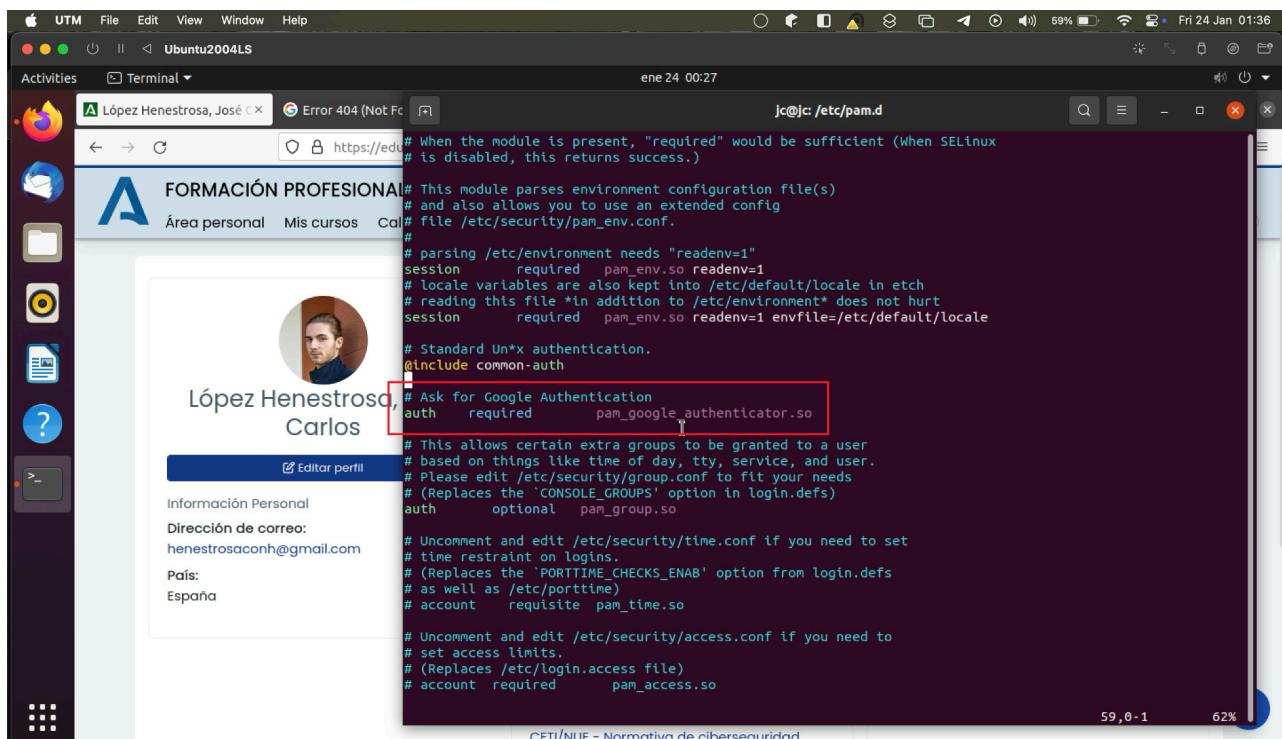
Login en entorno gráfico

Configurando el archivo `/etc/pam.d/lightdm` (o equivalente) para que el sistema solicite un código de autenticación en el inicio de sesión gráfico.

Mi distribución de Ubuntu usa GNOME como *display manager*, por lo que voy a configurar el archivo `/etc/pam.d/gdm-password` en lugar de `/etc/pam.d/lightdm` para cumplir con lo que pide el enunciado.

Para ello, es necesario editar dicho archivo para añadir la siguiente línea tras incluir los contenidos del archivo `common-auth`, tal y como hicimos con el archivo `/etc/pam.d/login`:

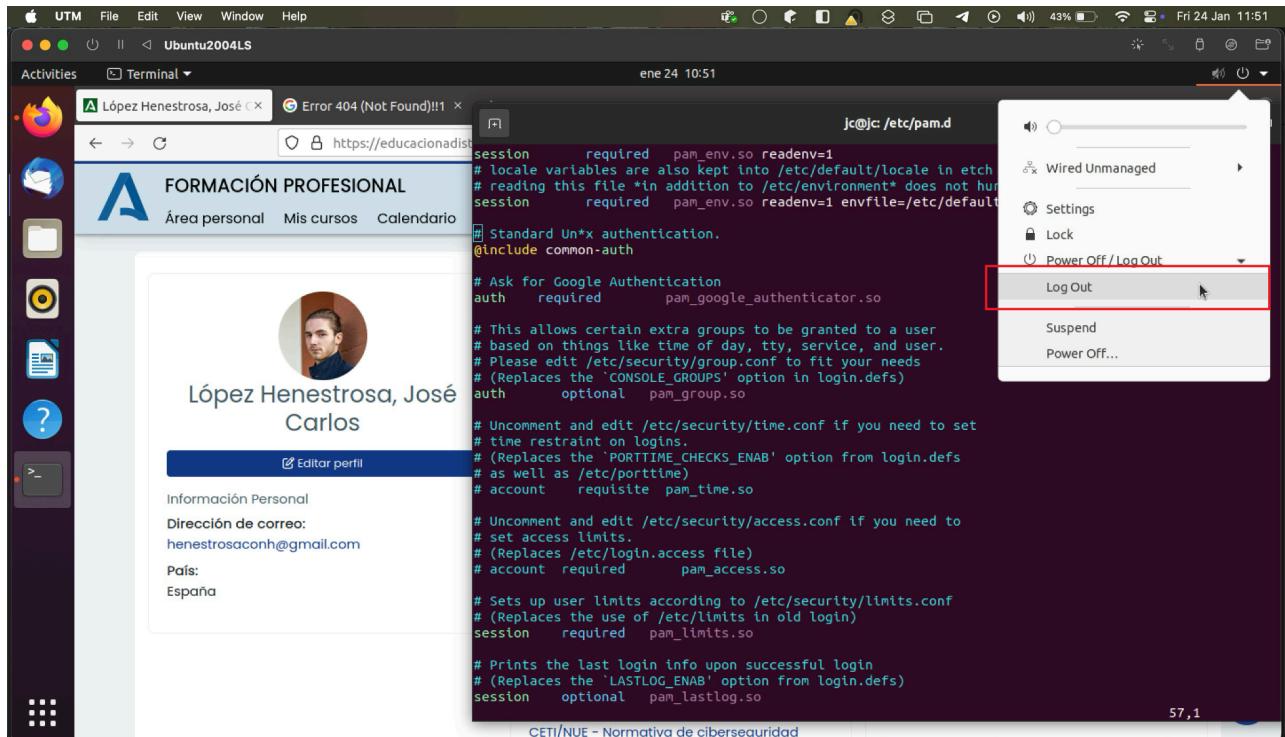
```
auth      required      pam_google_authenticator.so
```



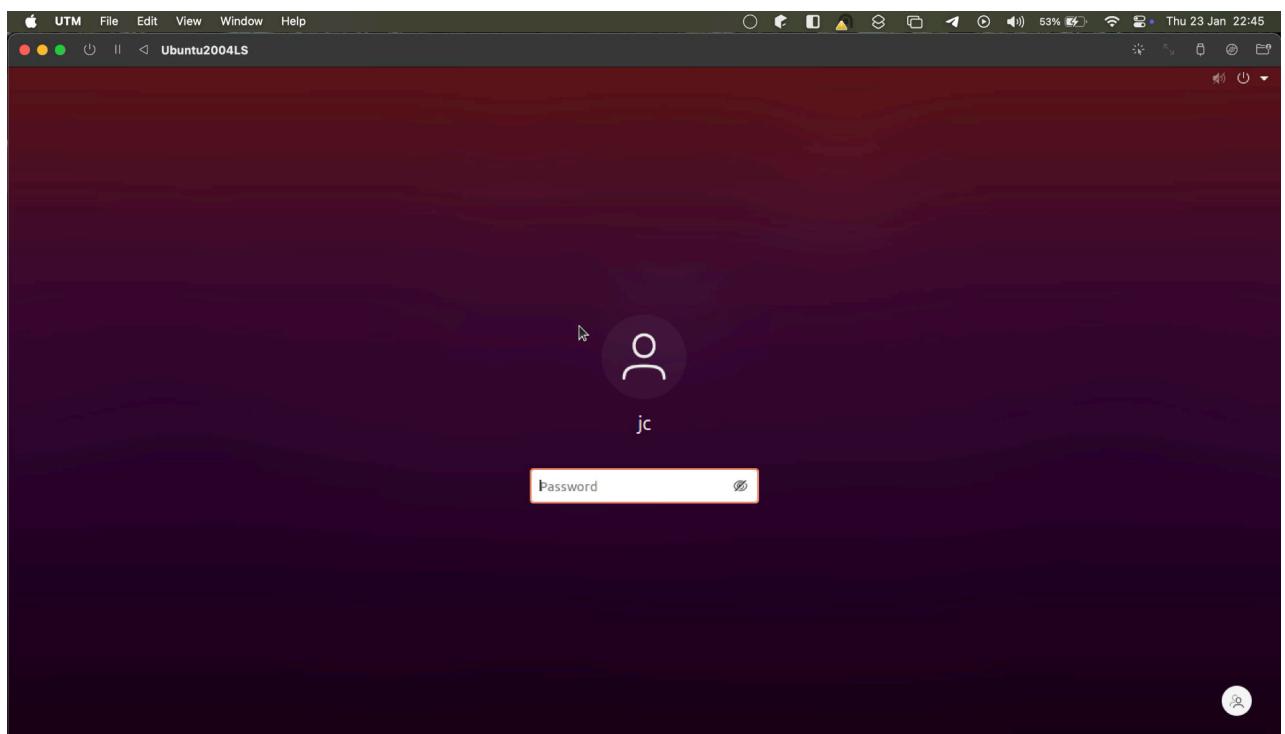
Archivo `/etc/pam.d/gdm-password` con los cambios marcados en rojo

Al realizar este proceso, ejecutamos `sudo systemctl restart gdm` para que los cambios surtan efecto.

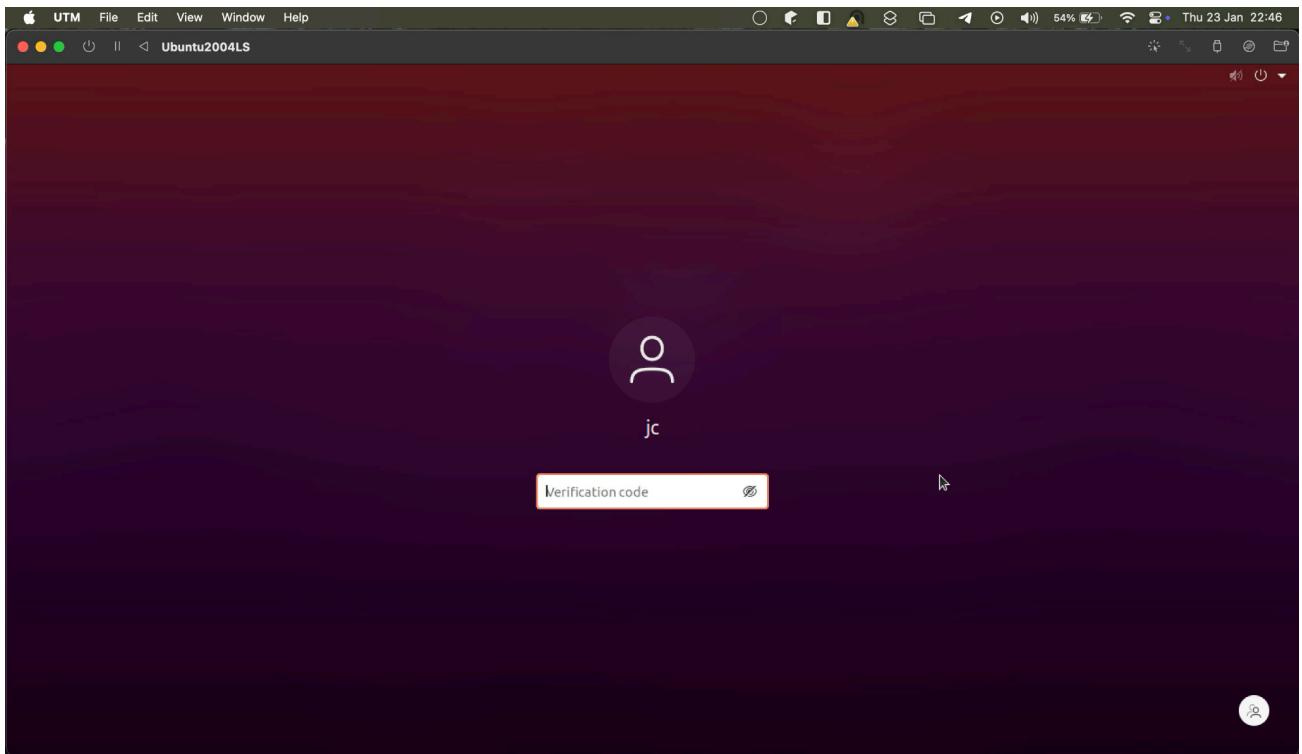
Tras esto, cerramos la sesión y la volvemos a iniciar para comprobar que, efectivamente, se pide el código de verificación al intentar iniciar sesión de nuevo.



Cerrando sesión en el entorno gráfico



Petición de contraseña al intentar iniciar sesión en entorno gráfico



Petición de código de verificación tras introducir la contraseña al intentar iniciar sesión en entorno gráfico

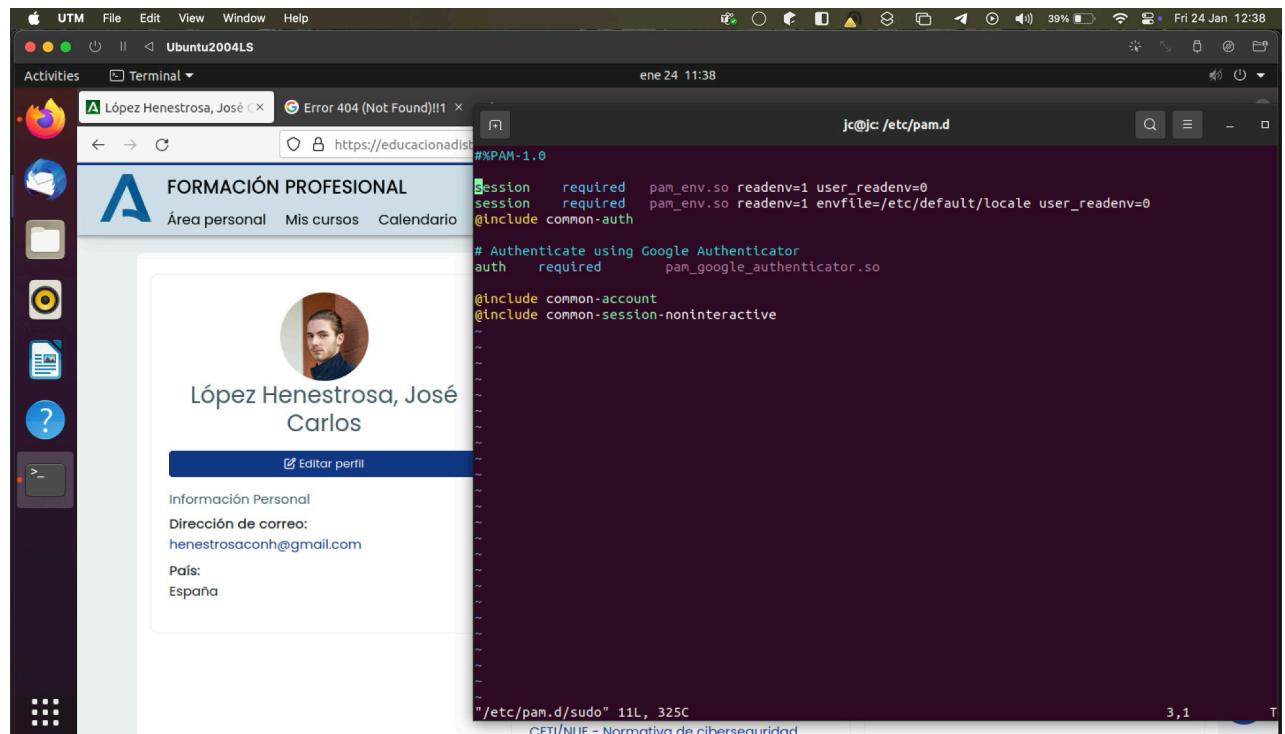
Como podemos observar, la configuración se ha aplicado con éxito.

Uso del comando sudo

Configurando el archivo `/etc/pam.d/sudo` para que el sistema requiera un código de autenticación adicional cuando se ejecute un comando con privilegios elevados.

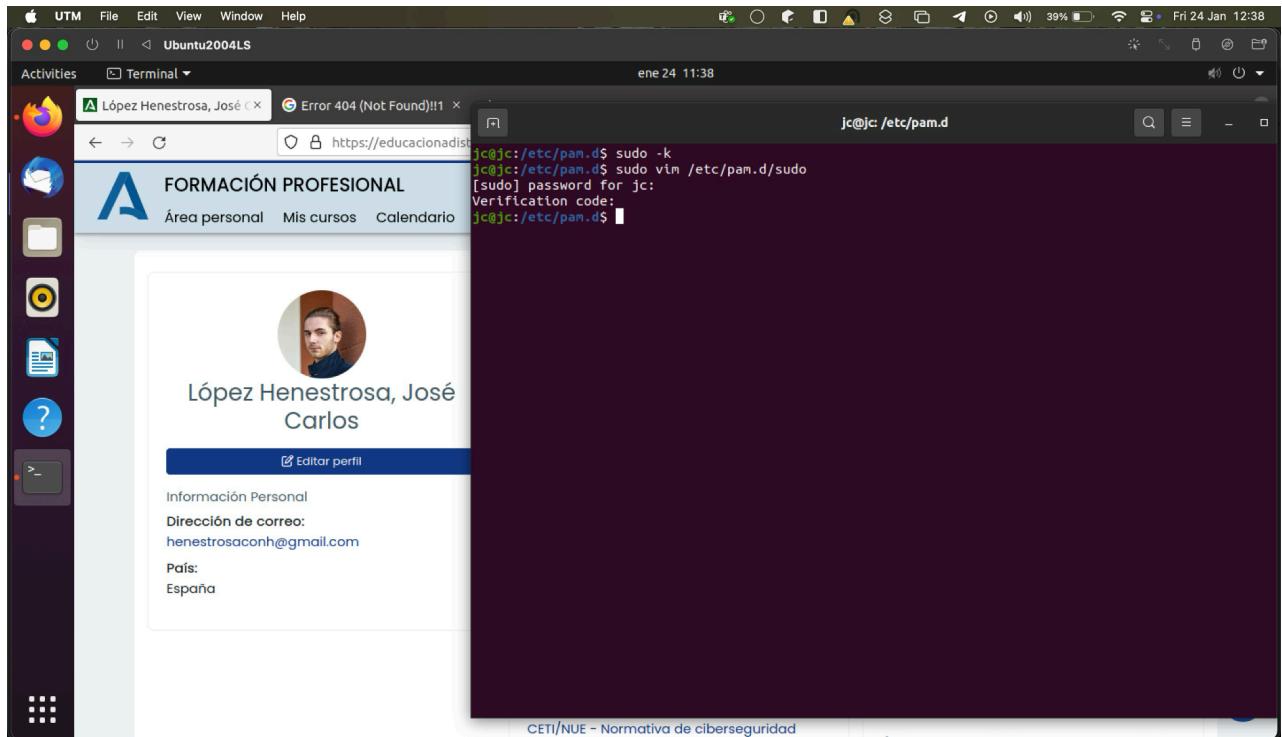
Para ello, tenemos que editar el archivo `/etc/pam.d/sudo` para añadir la siguiente línea después de incluir los contenidos del archivo `common-auth`, tal y como hicimos en los procesos de configuración anteriores:

```
auth      required      pam_google_authenticator.so
```



Archivo `/etc/pam.d/sudo` con los cambios marcados en rojo

Tras esto, borramos la caché de autenticación de sudo con el comando sudo -k y ejecutamos cualquier comando con sudo para realizar la acción con permisos elevados:



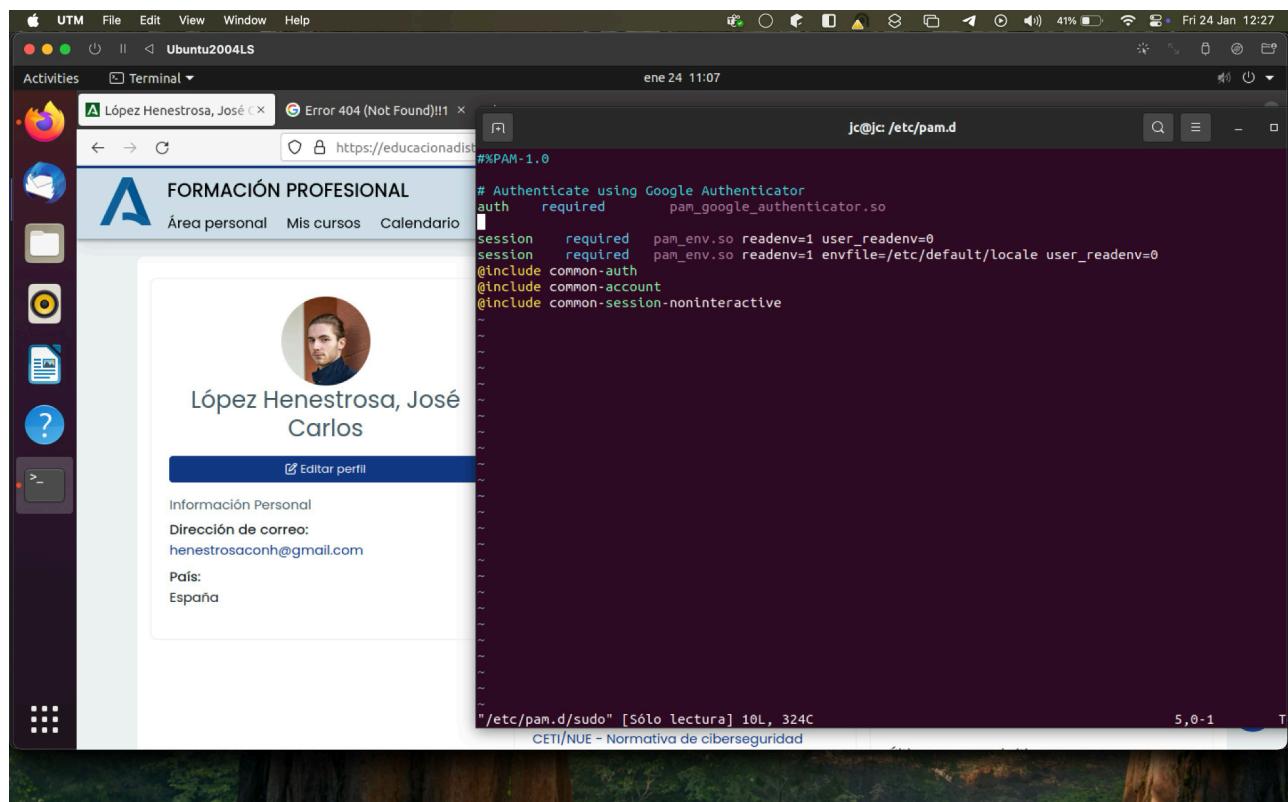
Petición de código de verificación tras ejecutar un comando con privilegios de superusuario

Como podemos comprobar, se pide la contraseña y el código de verificación correctamente.

Preguntas adicionales

¿Por qué influye el **orden** en el que se añaden las directivas en el archivo PAM que se está configurando?

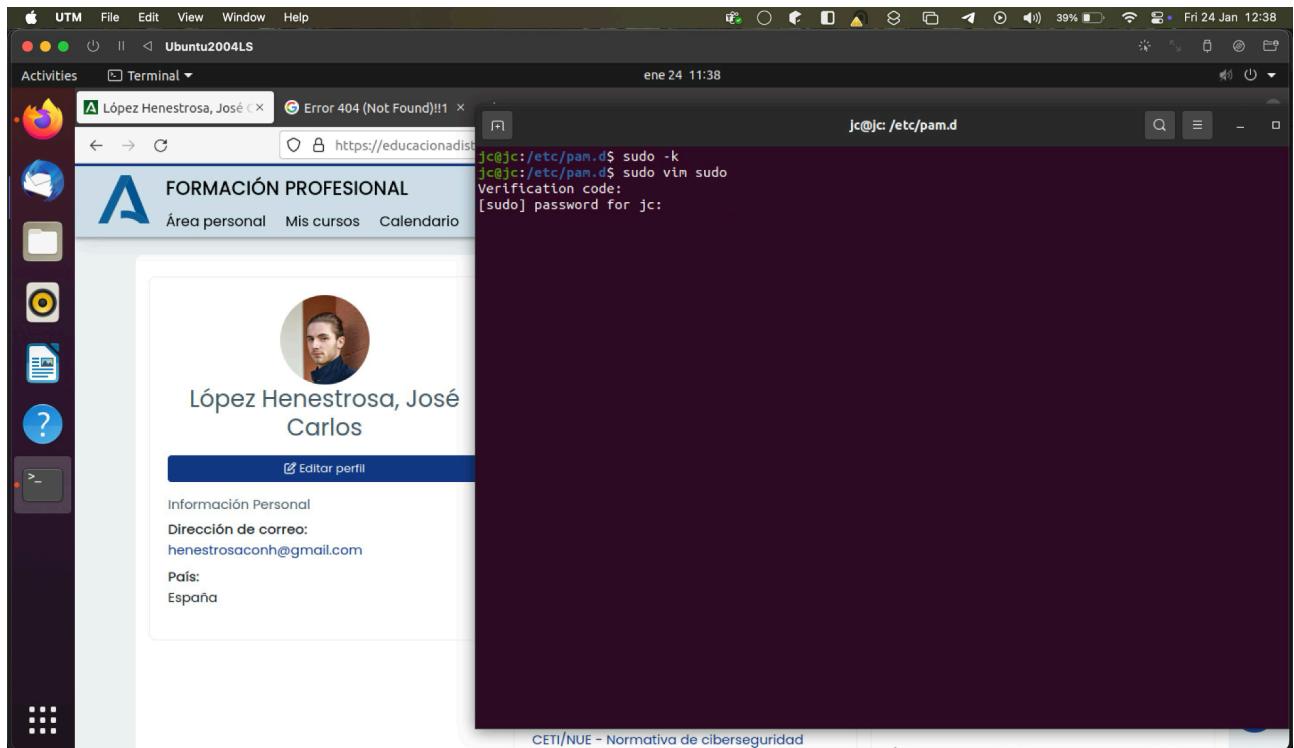
El orden es crucial porque PAM evalúa las directivas de arriba hacia abajo, lo que influye significativamente en el proceso de autenticación de los usuarios. Por ejemplo, si movemos la línea `auth required pam_google_authenticator.so` al principio del archivo `/etc/pam.d/sudo`, tal que así:



```
#%PAM-1.0
# Authenticate using Google Authenticator
auth required pam_google_authenticator.so
session required pam_env.so readenv=1 user_readenv=0
session required pam_env.so readenv=1 envfile=/etc/default/locale user_readenv=0
@include common-auth
@include common-account
@include common-session-noninteractive
```

Archivo `/etc/pam.d/sudo` con los cambios marcados en rojo

Como podemos apreciar, el orden de petición del código de verificación se invierte; es decir, ahora se solicita el código de verificación de Google Authenticator antes que la contraseña, tal y como se muestra en la siguiente captura:



Petición de código de verificación invertido respecto al apartado [Uso del comando sudo.](#)

¿En qué otras acciones o servicios sería aconsejable configurar PAM para reforzar la autenticación del usuario?

Se puede utilizar en multitud de situaciones. Por ejemplo, para proteger conexiones sensibles de transferencia de datos y similares, como VPN (Virtual Private Network), SSH (Secure Socket Shell), FTP (File Transfer Protocol) o SMTP (Simple Mail Transfer Protocol). Asimismo, también se podría usar para integrar servicios en red con sistemas de autenticación centralizados, como LDAP (Lightweight Directory Access Protocol) o NIS (Network Information Service).

Por otra parte, mediante el uso de módulos, se puede emplear para restringir el acceso dependiendo de la hora del día o de la semana y la dirección IP del dispositivo, así como bloquear cuentas e intentos fallidos de inicio de sesión a consecuencia de ataques de fuerza bruta con el módulo [pam_tally2](#). Además, gracias al módulo [pam_ecryptfs](#), también es posible configurar el desbloqueo de particiones cifradas al iniciar sesión e imponer políticas estrictas de contraseñas, como la caducidad de las mismas y la prevención de la reutilización de contraseñas previamente usadas.

Por último, cabe destacar que PAM también puede integrarse en sistemas donde se requiere cumplir con normativas que exigen mantener un seguimiento de quién accede al sistema y cuándo, configurar alertas en caso de accesos sospechosos y conocer qué intentos de inicio de sesión han fallado. Esto resulta muy útil de cara a auditorías.

Como podemos apreciar, este módulo va más allá de añadir la capa de 2FA al proceso de autenticación del usuario, ya que también se puede utilizar en otros contextos esenciales para garantizar la seguridad del acceso al sistema.

¿Cuál es la finalidad de los archivos common-account, common-password, common-auth y common-session?

Dichos archivos son configuraciones compartidas y centralizadas que definen políticas comunes relacionadas con la autenticación, la autorización y la gestión de sesiones y contraseñas. Estos archivos están, por lo general, ubicados en el directorio `/etc/pam.d/` y su propósito es evitar la duplicación de configuraciones entre diferentes servicios o aplicaciones.

He aquí una explicación breve sobre cada uno:

- `common-account`: Contiene la configuración de las políticas de autenticación del usuario. Se utiliza, por ejemplo, durante el inicio de sesión, el uso de `sudo` o el acceso SSH.
- `common-password`: Gestiona las políticas relacionadas con las contraseñas, por ejemplo, al usar el comando `passwd`.
- `common-auth`: Define si un usuario tiene permiso para acceder al sistema o utilizar un recurso del mismo. En un caso de uso real, este archivo determina si un usuario está autorizado a continuar cuando se verifica su identidad.
- `common-session`: Administra las políticas relacionadas con las sesiones de un usuario, específicamente al iniciar o cerrar una sesión, ya sea de forma local o remota.

En un caso de uso genérico, el orden de ejecución de los archivos sería el siguiente:

1. En primer lugar, se verifica la identidad del usuario (`common-auth`).
2. Luego se comprueba si el usuario está autorizado a acceder al recurso (`common-account`).
3. Si se requiere un cambio de contraseña, se aplican las políticas definidas en `common-password`.
4. Finalmente, se gestionan las tareas relacionadas con la sesión (`common-session`).

¿Por qué se genera un código semilla al ejecutar el comando google-authenticator y cuál es su propósito en la autenticación?

El código semilla que se genera es una parte fundamental del mecanismo de autenticación, ya que establece un punto de partida compartido entre el servidor y el cliente para generar códigos TOTP.

Estos son los propósitos principales de un código semilla en el proceso de autenticación:

- Ser la base, junto con la hora actual, para que los códigos TOTP generados sean únicos.
- Proteger la cuenta con una segunda capa, además de la contraseña.
- Sincronizar el servidor y la app para que generen los mismos códigos sin necesidad de conexión constante.

Cabe destacar que este código no debe compartirse con nadie, ya que cualquiera con acceso a él puede generar los mismos códigos TOTP.

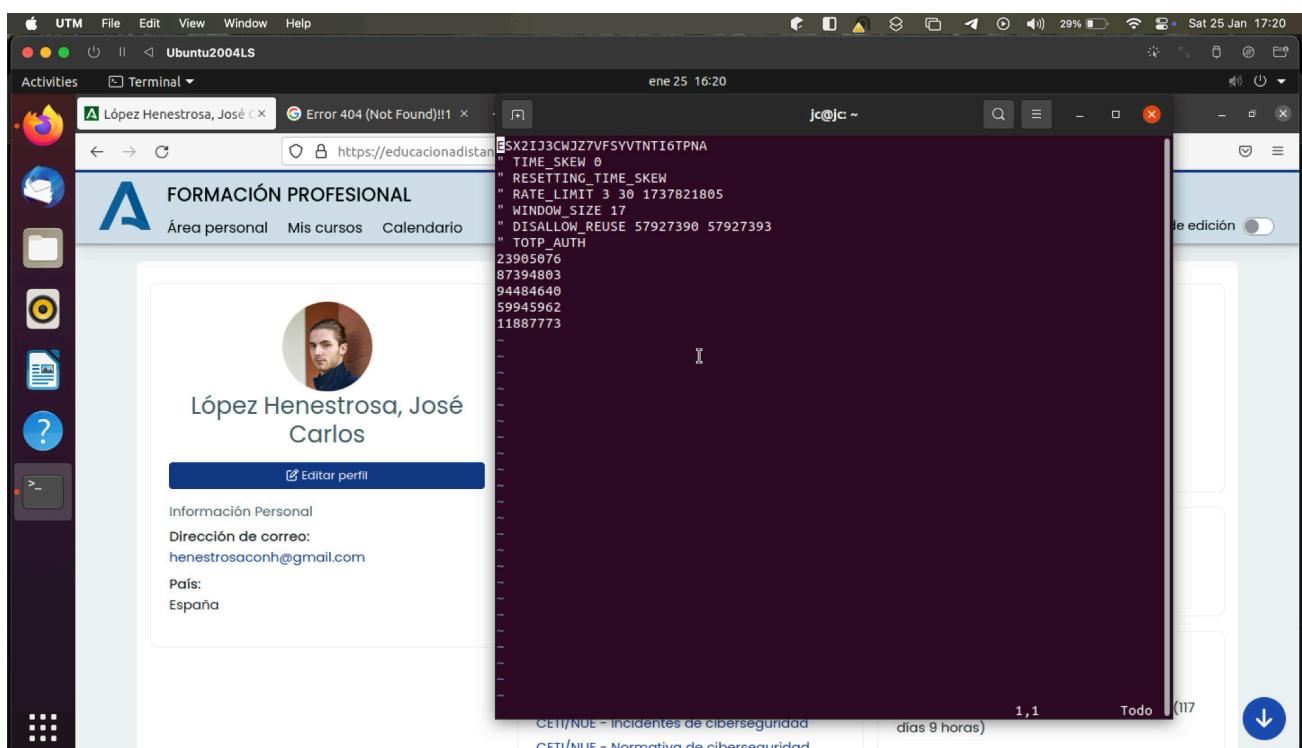
¿Por qué se proporcionan una serie de **códigos preestablecidos** al usuario una vez se ha generado el QR y cuál es su finalidad?

Se proporcionan porque sirven como códigos de recuperación en caso de que el usuario pierda acceso a la app de Google Authenticator o a su dispositivo. Tienen diversas finalidades, como:

- Permiten acceder a la cuenta sin necesidad de utilizar un nuevo dispositivo o de configurar el sistema desde cero.
- Cada código preestablecido se puede usar solo una vez, lo que impide que se puedan reutilizar si alguien más tiene acceso a uno que ya haya sido usado.
- Actúan como un mecanismo seguro para restablecer el acceso, ya que el usuario tiene que almacenarlos manualmente (por ejemplo, en papel o en un gestor de contraseñas).

¿En qué ubicación se almacena la configuración de Google Authenticator y los códigos preestablecidos para el usuario? Explica cómo se pueden usar estos códigos en caso de no disponer de la aplicación Google Authenticator, especificando que cada código solo se puede utilizar una vez

La configuración de Google Authenticator y los códigos de recuperación se almacenan en el archivo `~/.google_authenticator` del directorio personal del usuario (`~`). En él se almacenan el código semilla, las opciones de configuración y los códigos de recuperación generados durante la configuración inicial, tal y como se puede comprobar en la siguiente captura:



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "Terminal" and the command entered is "cat ~/.google_authenticator". The output of the command is a long string of characters: "SX2IJ3CWJZ7VFSYVTNTI6TPNA TIME_SKew 0 RATE_LIMIT 3 30 1737821805 WINDOW_SIZE 17 DISALLOW_REUSE 57927390 57927393 TOTP_AUTH 23965076 87394803 94484640 59945962 11887773". Below the terminal window, a web browser is visible showing a profile page for "López Henestrosa, José Carlos".

Contenidos del archivo `~/.google_authenticator`

En caso de no disponer de la aplicación Google Authenticator, se pueden utilizar los códigos preestablecidos durante el proceso de autenticación (por ejemplo, al iniciar sesión), cuando el sistema solicita el código TOTP. En lugar de introducir un código generado por la app, se introduciría uno de los códigos de recuperación proporcionados.

Después de usar uno de los códigos preestablecidos, este queda invalidado automáticamente, mientras que los códigos restantes siguen siendo válidos. Si se usan todos los códigos de recuperación, se necesitará reconfigurar Google Authenticator.