Administración de Sistemas Operativos - 1a Evaluación (RA 4 – CE d, e, f)
Unidad Didáctica 4. Configuración multiusuario centralizada
Jose Manuel Lizana Montero

Ejercicio 1.

Investiga sobre qué es PAM.

La Gestión de Acceso Privilegiado (PAM) es una solución de seguridad de identidad que protege a las organizaciones contra las amenazas cibernéticas al controlar y gestionar el acceso a recursos críticos, como redes, sistemas y datos. Se enfoca en las cuentas con privilegios elevados, como administradores y superusuarios, que pueden tener acceso ilimitado a recursos confidenciales si se ven comprometidas.

Funcionalidades clave

-Control de acceso: Implementar controles estrictos para restringir y monitorear el acceso a cuentas con privilegios.
-Visibilidad: Proporcionar visibilidad sobre el uso de cuentas con privilegios y las actividades realizadas durante su conexión.
-Restricción del número de usuarios con acceso a funciones administrativas: Aumentar la seguridad del sistema al reducir el número de usuarios con acceso a funciones administrativas.
-Capas de protección adicionales: Prevenir la filtración de información por parte de actos malintencionados mediante la adopción de capas de protección adicionales.

Objetivo
El objetivo de la Gestión de Acceso Privilegiado (PAM) es garantizar la protección de los recursos críticos de las organizaciones contra las amenazas cibernéticas, evitando accesos no autorizados y violaciones de seguridad graves.

Ejercicio 2.

Busca información sobre los tipos de módulos y las banderas de control utilizadas en PAM.

Módulos y Categorías en PAM

PAM (Pluggable Authentication Modules) es un sistema modular que permite configurar y personalizar los mecanismos de autenticación en sistemas Linux y Unix. Los módulos se agrupan en cuatro tipos principales o categorías funcionales. Cada categoría está diseñada para manejar un aspecto específico del proceso de autenticación.

auth (AUTENTICACIÓN)

Esta categoría se encarga de **verificar la identidad del usuario**. Es el primer paso en el proceso de autenticación, donde se solicitan y validan credenciales como contraseñas, huellas digitales o tokens.
Funciones principales:
-Validar las credenciales del usuario.
-Solicitar información adicional.
-Permitir o denegar el acceso inicial.

Módulos comunes en auth:

pam_unix.so
Autentica al usuario verificando contraseñas almacenadas en /etc/shadow.
pam_ldap.so
Permite la autenticación contra un servidor LDAP.
pam_fprintd.so
Autenticación mediante huella digital.
pam_google_authenticator.so
Implementa autenticación de dos factores (TOTP).
pam_faildelay.so
Introduce un retraso configurable tras intentos fallidos de autenticación.

account (Gestión de cuentas)

Esta categoría verifica que la cuenta del usuario tenga permiso para acceder al sistema. No se centra en las credenciales, sino en las **restricciones de la cuenta**, como fechas de expiración, horarios permitidos o configuraciones de acceso.

Funciones principales:

Validar que la cuenta está activa y no expirada.
Comprobar si el usuario tiene acceso permitido en ese momento.
Aplicar reglas específicas para el acceso.

Módulos comunes en account:

pam_time.so
Restringe el acceso al sistema basado en horarios configurados.
pam_access.so
Controla el acceso basado en reglas definidas en /etc/security/access.conf.
pam_exec.so
Ejecuta scripts personalizados para decidir si se permite o deniega el acceso.
pam_succeed_if.so
Aplica condiciones lógicas para determinar el éxito o fallo de la autenticación.
pam_limits.so
Establece límites de recursos (CPU, memoria, etc.) por usuario o grupo.

password (gestor de contraseñas)

Este tipo de módulo se utiliza para gestionar el **cambio y actualización de contraseñas**. Es relevante cuando el usuario debe cambiar su contraseña, ya sea porque ha expirado o por políticas de seguridad.

Funciones principales:
Validar la calidad o fortaleza de las contraseñas nuevas.
Cambiar contraseñas almacenadas en el sistema o servidores remotos.
Notificar al usuario cuando su contraseña está a punto de expirar.

Módulos comunes en password:

pam_unix.so
Cambia contraseñas almacenadas en el sistema local.
pam_ldap.so
Cambia contraseñas en servidores LDAP.
pam_cracklib.so
Verifica que las nuevas contraseñas cumplan con políticas de complejidad.
pam_pwquality.so
Sustituto moderno de pam_cracklib para validar la fortaleza de contraseñas.
pam_tally2.so
Lleva un registro de intentos fallidos y puede bloquear al usuario tras varios intentos.

sesión (gestión de sesión)

Esta categoría maneja las **acciones realizadas al inicio o cierre de una sesión**. Estas acciones pueden incluir la configuración del entorno del usuario, montaje de directorios o registro de auditorías.

Funciones principales:

Configurar el entorno del usuario (variables, directorios, etc.).
Registrar eventos relacionados con la sesión (auditoría).
Limpiar recursos al cerrar la sesión.

Módulos comunes en session:

pam_limits.so
Aplica límites de recursos como número de procesos o uso de memoria.
pam_env.so
Establece variables de entorno.
pam_mkhomedir.so
Crea automáticamente el directorio home del usuario si no existe.
pam_exec.so
Ejecuta scripts personalizados al inicio o cierre de la sesión.
pam_lastlog.so
Muestra información sobre el último inicio de sesión del usuario.

Aunque cada módulo pertenece principalmente a una categoría, algunos pueden ser usados en múltiples tipos dependiendo de la configuración. Por ejemplo, pam_unix.so puede actuar como **auth**, **password**, o **sesión**.

Las banderas de control admitidas son

requisito: la falla devuelve instantáneamente el control a la aplicación indicando la naturaleza de la falla del primer módulo.
requerido: todos estos módulos son necesarios para que libpam devuelva el éxito a la aplicación.

suficiente: dado que todos los módulos anteriores han tenido éxito, el éxito de este módulo conduce a un regreso inmediato y exitoso a la aplicación (se ignora el fracaso de este módulo).
opcional: el éxito o el fracaso de este módulo generalmente no se registra.

Además de las palabras clave anteriores, existen otras dos banderas de control válidas:
incluir y subpilar: incluye todas las líneas de un tipo dado del archivo de configuración especificado como argumento para este control.

Ejercicio 3

Explica la arquitectura asociada.

La arquitectura asociada a PAM se enfoca en proteger las identidades con privilegios, tanto humanas como no humanas, y garantizar el acceso seguro y autorizado a recursos críticos.

La arquitectura PAM típica consta de las siguientes componentes:

Repository de credenciales: Un almacén seguro y centralizado para almacenar credenciales privilegiadas, como contraseñas, claves SSH, tokens y certificados. Este repositorio debe ser resistente a la manipulación y cumplir con normas de seguridad como PCI-DSS, HIPAA o GDPR.

Portal de autenticación: Un punto de acceso seguro para los usuarios con privilegios para iniciar sesión y acceder a recursos protegidos. El portal debe implementar autenticación multifactor adaptativa según el contexto y validar los usuarios con privilegios.

Motor de autorización: Un componente que evalúa las solicitudes de acceso y verifica si el usuario tiene los permisos necesarios para acceder al recurso solicitado. El motor de autorización debe ser configurable para definir políticas de acceso y permisos.

Sistema de gestión de sesiones: Un componente que monitorea y controla las sesiones de los usuarios con privilegios, detectando y respondiendo a comportamientos anómalos o indicadores de peligro.

Integración con sistemas y aplicaciones: PAM se integra con sistemas y aplicaciones para controlar el acceso a recursos protegidos, como bases de datos, sistemas operativos y aplicaciones empresariales.

Políticas y configuración: Un componente que permite definir y configurar políticas de seguridad para el acceso privilegiado, como la complejidad de contraseñas, la frecuencia de rotación y qué usuarios pueden acceder a qué recursos.

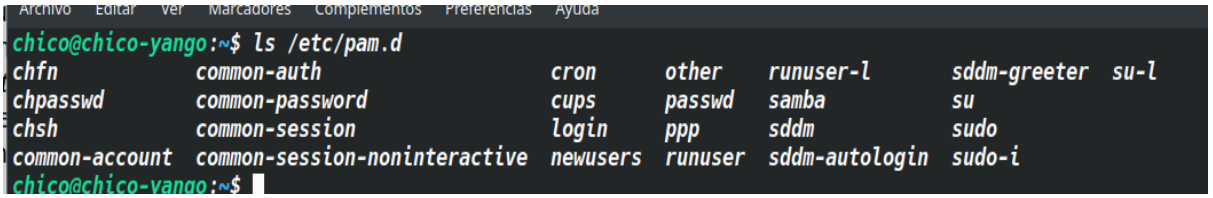
Reporting y auditoría: Un componente que proporciona informes y auditoría detallados sobre las actividades de los usuarios con privilegios, ayudando a los administradores a identificar y mitigar riesgos.

Ejercicio 4.

Expón qué directorios y ficheros son de gran importancia.

El archivo de configuración principal para PAM es **/etc/pam.conf** y el directorio **/etc/pam.d/** contiene los archivos de configuración de PAM para cada aplicación/servicio compatible con PAM.
Por defecto Linux, usará la ruta */etc/pam.d/* para localizar los ficheros necesarios que le indiquen las operaciones que debe de ejecutar para cada servicio que requiera autenticación.
En caso de que no encuentre nada allí, usará la configuración que exista en el fichero */etc/pam.conf*

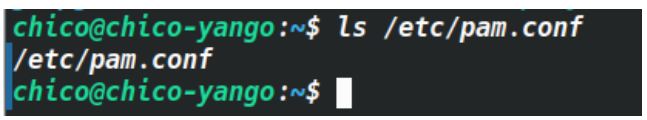
Dependiendo del software instalado, en la ruta **/etc/pam.d** si listamos su contenido veremos una serie de ficheros:



Lo primero que debemos saber es que existen 3 tipos de ficheros:
common: Son los ficheros base para la autenticación de casi todos los demás servicios.
other: Este fichero se usa cuando a un programa no se le haya indicado ninguna configuración.
Resto de ficheros: Cada programa instalará una configuración especial si necesita algún tipo de autenticación.

/etc/pam.d/common-auth: Define las políticas comunes de autenticación, como el uso de contraseñas.
/etc/pam.d/common-account: Contiene configuraciones para la gestión de cuentas, como la validación de las cuentas de usuario.
/etc/pam.d/common-password: Establece las políticas relacionadas con las contraseñas, como la longitud mínima o los requisitos de complejidad.
/etc/pam.d/common-session: Gestiona la configuración de las sesiones de usuario, como la creación de variables de entorno o el manejo de sesiones activas.
/etc/pam.d/sudo: Configura la autenticación para el uso de privilegios elevados mediante sudo.

/etc/pam.conf
Este archivo es una alternativa a la configuración en el directorio /etc/pam.d/. Aunque menos común, **si está presente**, define una configuración global de PAM para todos los servicios. En sistemas más modernos, se prefiere el uso de los archivos en /etc/pam.d/, pero si existe este archivo, se aplica como una configuración predeterminada a nivel global.



Ejercicio 5.

Crea una tabla que contenga los 10 módulos de PAM que más te hayan llamado la atención. Cada fila deberá contener el nombre del módulo, descripción, opciones y varios ejemplos de uso.

Módulo PAM	Descripción	Opciones	Ejemplos de uso
pam_unix.so	Proporciona autenticación basada en el sistema Unix local, usando contraseñas almacenadas en /etc/shadow.	nullok, try_first_pass, use_authtok	**auth required pam_unix.so** (Autenticación estándar mediante contraseñas). **password required pam_unix.so nullok** (Permite contraseñas vacías).
pam_tally.so	Lleva un registro de los intentos de acceso, útil para detectar ataques de fuerza bruta.	deny, reset, file, onerr	**auth required pam_tally.so deny=3** (Deniega acceso después de 3 intentos fallidos). **auth required pam_tally.so reset** (Resetea el contador tras éxito).
pam_google_authenticator.so	implementa la autenticación de dos factores usando Google Authenticator.	secret, force, window	**auth required pam_google_authenticator.so** (Autenticación de dos factores). **auth required pam_google_authenticator.so secret=/path/to/secretfile** (Ruta del archivo de secretos).
pam_securetty.so	Restringe el acceso a usuarios en terminales no seguros (especificados en /etc/securetty).	ttyfile	**auth required pam_securetty.so** (Solo permite el acceso desde terminales definidos en **/etc/securetty).**
pam_env.so	Permite cargar variables de entorno desde archivos específicos de PAM.	file, user, reset	**session required pam_env.so** (Carga las variables de entorno del archivo predeterminado).
pam_listfile.so	Permite bloquear o	item, sense, file,	**auth required**

	permitir el acceso a ciertos usuarios basándose en un archivo de listas.	onerr	**pam_listfile.so item=user sense=deny file=/etc/pam_blocked_users** (Bloquea el acceso si el usuario está en la lista).
pam_limits.so	define límites de recursos para los usuarios, como el número de procesos o el tamaño de archivos.	limit_type, value, domain	**session required pam_limits.so** (Aplica límites definidos en **/etc/security/limits.conf)** .
pam_pwquality.so	Permite aplicar políticas de calidad de contraseñas, como longitud y complejidad.	minlen, minclass, maxrepeat	**password requisite pam_pwquality.so minlen=8 minclass=3** (Requiere contraseñas de al menos 8 caracteres y 3 clases de caracteres).
pam_exec.so	Permite ejecutar comandos externos durante el proceso de autenticación o sesión.	executable, chroot, umask	**auth required pam_exec.so /path/to/script.sh** (Ejecuta un script durante la autenticación).
pam_ldap.so	proporciona autenticación a través de un servidor LDAP (Lightweight Directory Access Protocol)	uri, base, ldap_version	**auth required pam_ldap.so uri=ldap://ldapserver.com** (Usa un servidor LDAP para la autenticación).

Ejercicio 6.

Realiza las siguientes configuraciones:

• Deshabilita el acceso vía SSH de cualquier usuario que no sea el usuario root. El sistema deberá de mostrar un mensaje cuando un usuario intente acceder.

Paso 1: Edito el archivo de configuración de PAM para SSH
Añado la siguiente línea al final del archivo /etc/pam.d/sshd

```
# Standard Un*x password updating.
@include common-password

auth required pam_listfile.so item_user sense=deny file=/etc/ssh/allowed_users onerr=succeed
```

pam_listfile.so: Permite bloquear o permitir el acceso a ciertos usuarios basándose en un archivo de listas.

Paso 2: Creo el archivo de usuarios permitidos

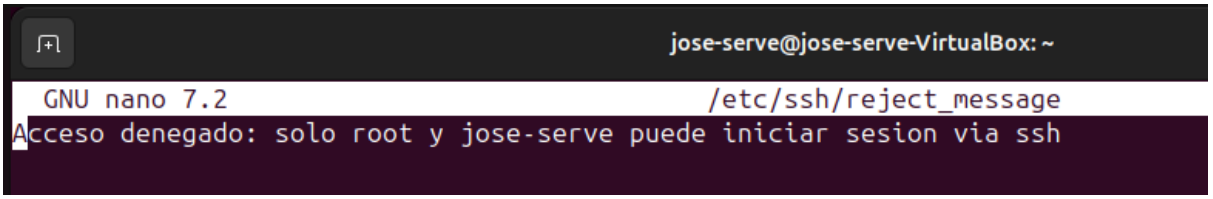
```
jose-serve@jose-serve-VirtualBox:~$ sudo nano /etc/ssh/allowed_users
```

Paso 3: Configura el mensaje de rechazo
Edito el archivo /etc/pam.d/sshd nuevamente:
Añado la siguiente línea después de la línea de pam_listfile.so:

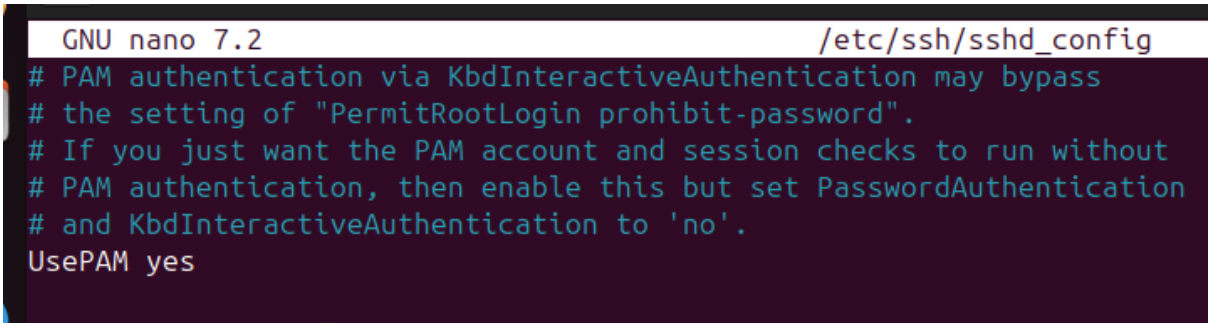
```
# Standard Un*x password updating.
@include common-password

auth required pam_listfile.so item_user sense=deny file=/etc/ssh/allowed_users onerr=succeed
auth optional pam_echo.so file=/etc/ssh/reject_message
```

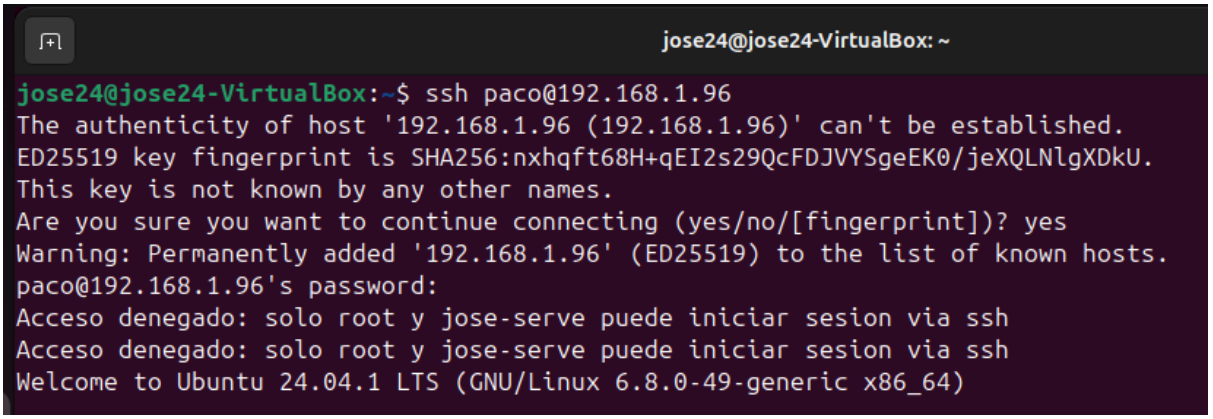
Paso 4: Creo el archivo /etc/ssh/reject_message
Escribo el mensaje de rechazo que se mostrará al usuario



Paso 5: Me aseguro de que PAM está habilitado verificando esta línea:

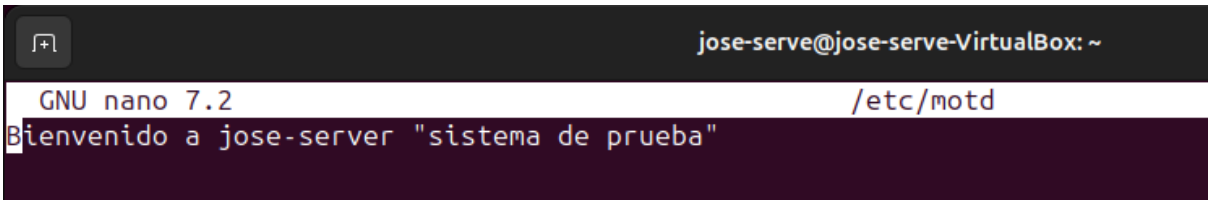


Pruebo la conexión:

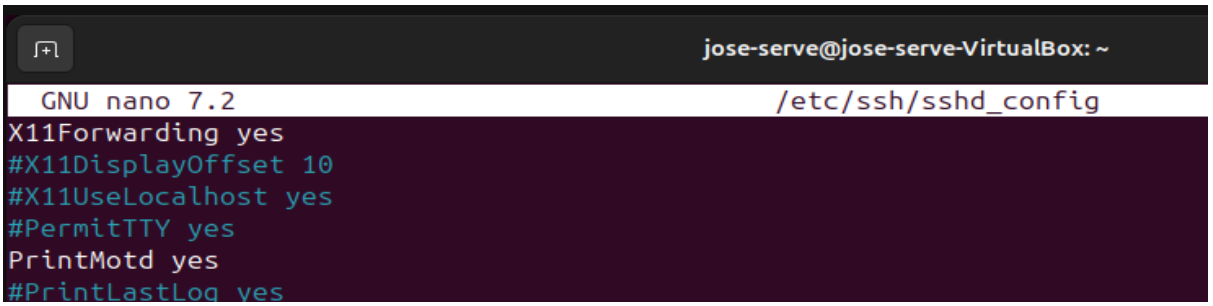


• Cambia el mensaje del día cuando un usuario se conecte vía SSH.

El mensaje del día se encuentra en el archivo: /etc/motd



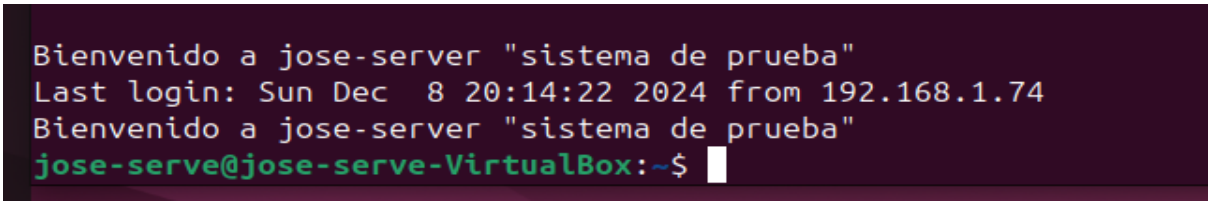
Me aseguro de que el archivo de configuración del servidor SSH permite mostrar el MOTD:sudo nano /etc/ssh/sshd_config (printmotd yes)



Guarda los cambios y reinicio el servicio SSH:

sudo systemctl restart sshd

Prueba el mensaje del día



• Configura franjas horarias en las que los usuarios pueden conectarse al sistema.

Configurar franjas horarias para el acceso de los usuarios al sistema se logra utilizando PAM (Pluggable Authentication Module) con el módulo **`pam_time.so`**. Este módulo permite definir restricciones basadas en horarios y días de la semana, tanto para sesiones locales como remotas.

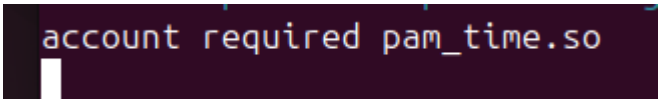
Ejemplo para el sistema

Paso 1: Editar la configuración de PAM

Abro el archivo de configuración para las sesiones del sistema:

sudo nano /etc/pam.d/common-auth

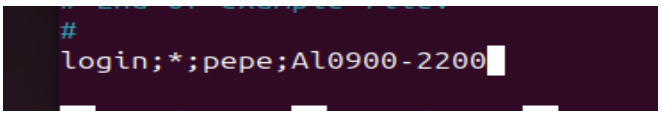
Añado la siguiente línea al final del archivo para activar las restricciones horarias:



Paso 2: Configurar las reglas de tiempo

Edito el archivo /etc/security/time.conf:

Definir las reglas para las franjas horarias.

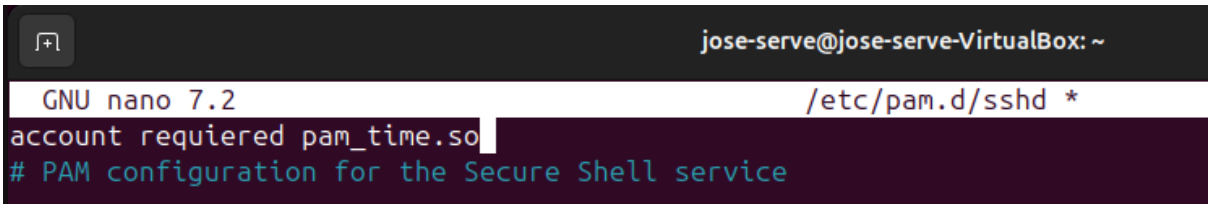


Para accesos locales, usa login.
Especifica el terminal. Usa * para aplicar la regla a todos.
Al → Lunes a Domingo de 9 a 10.

Ejemplo para conexión ssh.

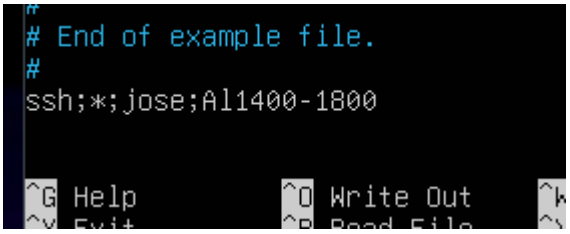
Paso 1: Configurar PAM para SSH

Añado la siguiente línea al inicio del archivo (si no está ya presente):
sudo nano /etc/pam.d/sshd



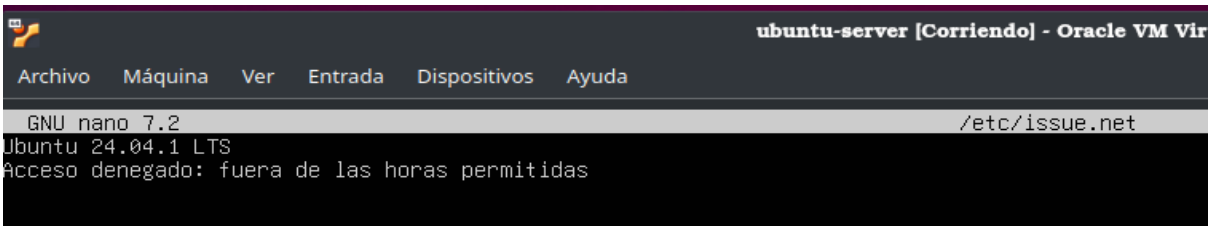
Esto indica que PAM debe aplicar las reglas definidas en /etc/security/time.conf para gestionar el acceso.

Paso 2: Configurar las franjas horarias en /etc/security/time.conf
El archivo /etc/security/time.conf permite definir las reglas de acceso basadas en horarios, días y servicios.



Jose tendra conexion solo de 14:00 a 18:00
Opcional:
Personalización del mensaje de rechazo
Si queremos mostrar un mensaje personalizado al denegar el acceso, se puede configurar en el archivo /etc/ssh/sshd_config.
Añadir o editar la línea:

Crea o edita el archivo /etc/issue.net para incluir el mensaje:



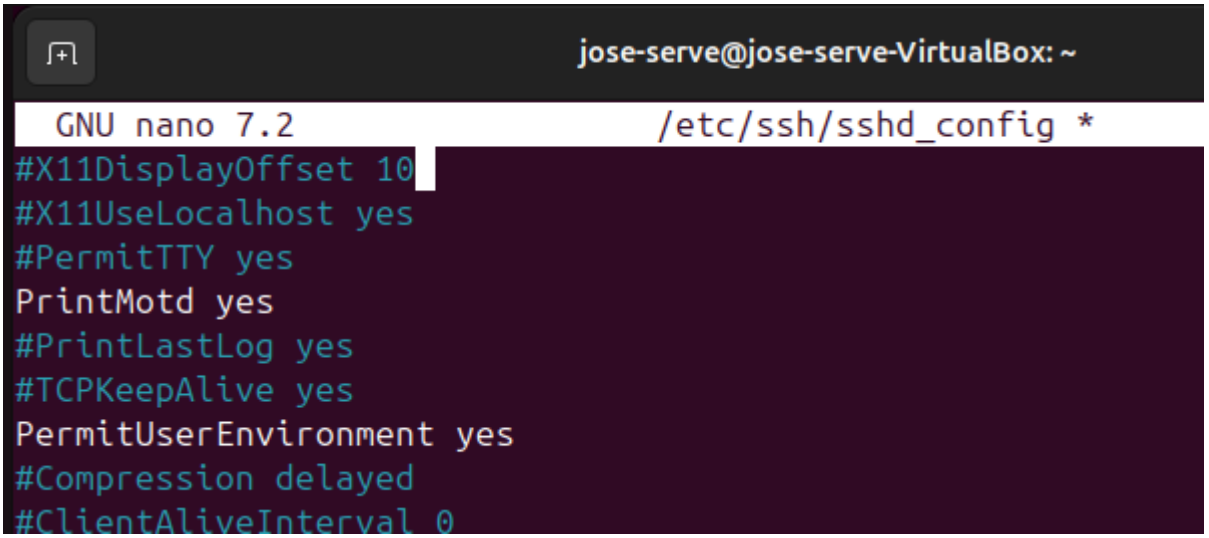
```
jose24@jose24-VirtualBox:~$ ssh jose@192.168.1.102
Ubuntu 24.04.1 LTS
Acceso denegado: fuera de las horas permitidas
jose@192.168.1.102's password:
```

• Define alguna variable de entorno que el usuario se encontrará al realizar la conexión.

Configurar variables de entorno globales (para todos los usuarios)

paso 1: Editar el archivo de configuración de SSH: /etc/ssh/sshd_config:

Busca la línea que contiene: permitiruserEnviroment cambiar a yes



Paso 2: Crear un archivo de entorno global

Edito el archivo /etc/environment para añadir variables globales.

Reiniciar el servicio SSH

Ejercicio 7.

Visita las siguientes páginas:

- **https://github.com/mlabouardy/pam-qrcode**
- **https://github.com/nahil1/pam-bluetooth**

Además de los anteriores, busca módulos PAM que te resulten interesantes. Explica qué te ha llamado la atención de cada uno de ellos.

pam_tally2
Descripción: Este módulo cuenta los intentos de inicio de sesión fallidos y bloquea la cuenta después de un número definido de intentos.
Por qué es interesante:
Proporciona una forma eficaz de prevenir ataques de fuerza bruta.
Permite registrar y auditar intentos fallidos de acceso.

pam_time
Descripción: Restringe el acceso al sistema según franjas horarias definidas.
Por qué es interesante:
Proporciona un control granular para limitar cuándo los usuarios pueden acceder al sistema.
Útil en entornos de trabajo con horarios estrictos o para garantizar que ciertos servicios estén disponibles solo en horas específicas.

am_exec
Descripción: Permite ejecutar comandos o scripts externos como parte del proceso de autenticación.
Por qué es interesante:
Ofrece flexibilidad para personalizar el comportamiento del sistema según las necesidades.
Puede integrarse con herramientas externas para realizar verificaciones adicionales durante el acceso.

pam_limits
Descripción: Aplica límites en el uso de recursos del sistema, como número de procesos, uso de memoria, o tiempo de CPU.
Por qué es interesante:
Permite proteger el sistema contra abusos y errores de los usuarios que puedan consumir excesivamente los recursos.
Es especialmente útil en servidores compartidos o con múltiples usuarios concurrentes.