

Day 32

Exploitation Analyst

Restrict users from using the old password:

Why is old password reuse a bad practice?

Reusing old passwords defeats the purpose of changing them. Here's why:

1. **Predictability:** Users often rotate through a small set of familiar passwords. Attackers who compromise one can guess future ones.
2. **Brute-force risk:** If an old password leaks, the user might reuse it again — making brute-force or credential stuffing attacks more effective.
3. **Security hygiene:** Preventing reuse encourages genuinely new passwords, reducing long-term exposure

Where are passwords stored in Linux?

Passwords are not stored in plaintext.

In Linux, passwords are securely stored as hashes in `/etc/shadow`, which is only readable by root, unlike `/etc/passwd`, which is world-readable and meant for general user info. `/etc/security/opasswd` stores hashes of old passwords to prevent reuse, enhancing password security and hygiene.

```
(root@kali)~# nano /etc/shadow
```

```
GNU nano 8.4 /etc/shadow
root:$5$3T5F0K0JMU7_s1h21q.0c.R1$g7WVSLUp1jG8J8Mo/InJ.ZXNDx5HkLL6ft9g/vrXap3:19946:0:99999:7:::
daemon:*:19870:0:99999:7:::
bin:*:19870:0:99999:7:::
sys:*:19870:0:99999:7:::
sync:*:19870:0:99999:7:::
games:*:19870:0:99999:7:::
man:*:19870:0:99999:7:::
lp:*:19870:0:99999:7:::
mail:*:19870:0:99999:7:::
news:*:19870:0:99999:7:::
uucp:*:19870:0:99999:7:::
proxy:*:19870:0:99999:7:::
www-data:*:19870:0:99999:7:::
backup:*:19870:0:99999:7:::
list:*:19870:0:99999:7:::
irc:*:19870:0:99999:7:::
_apt:*:19870:0:99999:7:::
nobody:*:19870:0:99999:7:::
systemd-network:*:19870:0:99999:7:::
systemd-timesync:*:19870:0:99999:7:::
messagebus:*:19870:0:99999:7:::
tss:*:19870:0:99999:7:::
strongswan:*:19870:0:99999:7:::
tcpdump:*:19870:0:99999:7:::
sshd:*:19870:0:99999:7:::
usbmux:*:19870:0:99999:7:::
```

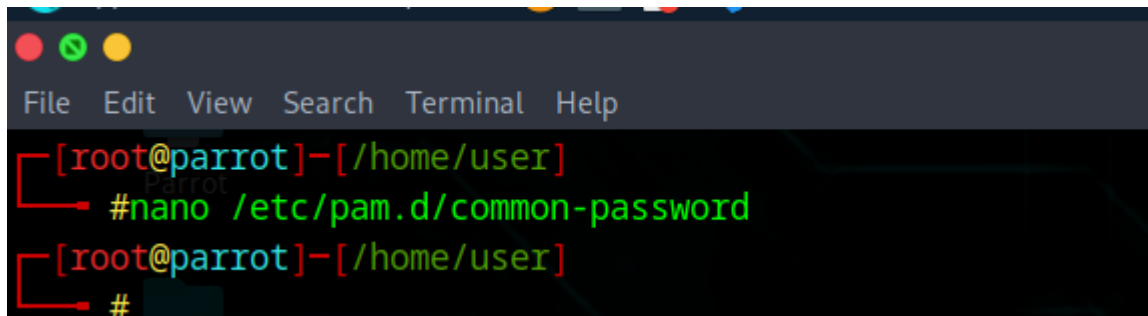
```
(root@kali)~# nano /etc/passwd
```

```
GNU nano 8.4 /etc/passwd
root:x:0:0:root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
```

Steps to Restrict users from using the old password:

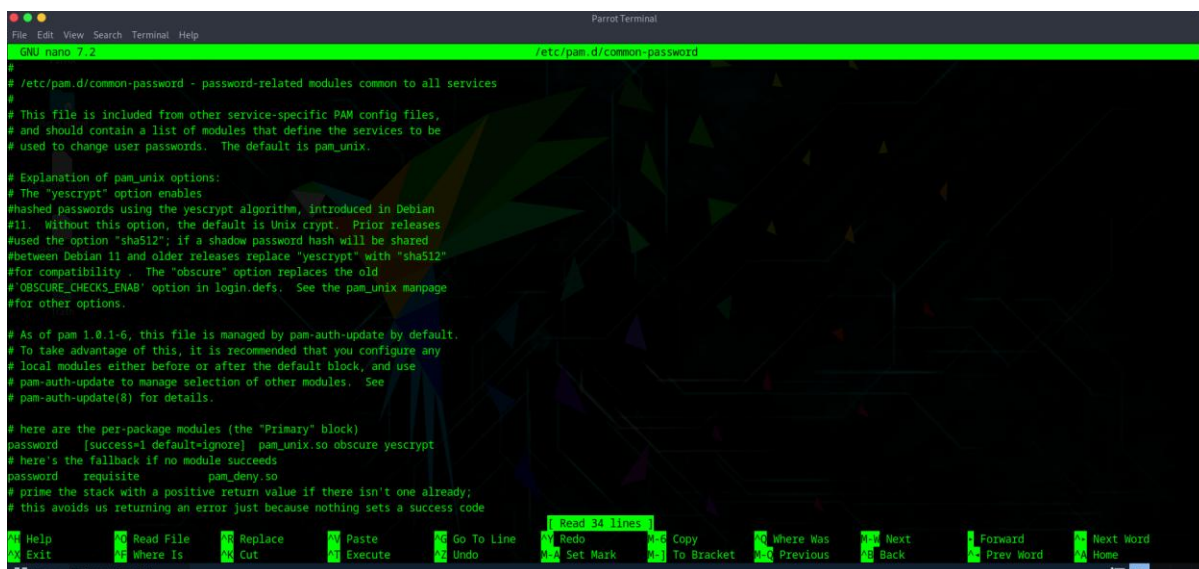
Steps:

Open the common-password file in pam.d:



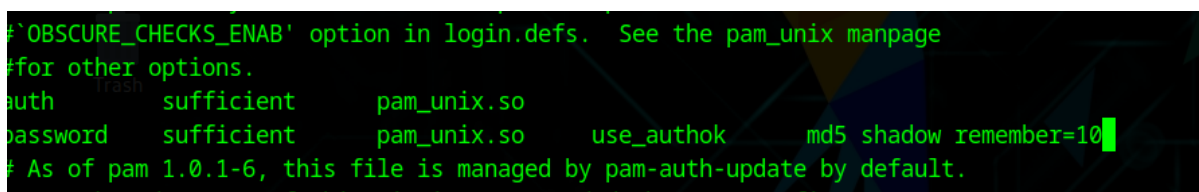
```
[root@parrot]-[/home/user]
# nano /etc/pam.d/common-password
[root@parrot]-[/home/user]
#
```

Following screen will appear:



```
GNU nano 7.2 /etc/pam.d/common-password
# /etc/pam.d/common-password - password-related modules common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords. The default is pam_unix.
#
# Explanation of pam_unix options:
# The "yescrypt" option enables
# hashed passwords using the yescrypt algorithm, introduced in Debian
# 9.11. Without this option, the default is Unix crypt. Prior releases
# used the option "sha512"; if a shadow password hash will be shared
# between Debian 11 and older releases replace "yescrypt" with "sha512"
# for compatibility. The "obscure" option replaces the old
# 'OBSCURE_CHECKS_ENAB' option in login.defs. See the pam_unix manpage
# for other options.
#
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules. See
# pam-auth-update(8) for details.
#
# here are the per-package modules (the "Primary" block)
password [success=1 default=ignore] pam_unix.so obscure yescrypt
# here's the fallback if no module succeeds
password requisite pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
#
```

Scroll down and add the following line:



```
# 'OBSCURE_CHECKS_ENAB' option in login.defs. See the pam_unix manpage
# for other options.
auth sufficient pam_unix.so
password sufficient pam_unix.so use_authok md5 shadow remember=10
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
```

This configuration ensures secure password handling in Linux. The `OBSCURE_CHECKS_ENAB` setting enforces strong password rules, while `shadow` stores passwords securely. The `remember=10` option prevents users from reusing their last 10 passwords. Using `sufficient` helps improve efficiency by skipping further checks once authentication succeeds.