PAM

Pluggable Authentication Modules

Curs 2017-2018

Descripció dels aprenentatges

---

1. El model de funcionament de PAM.
    a. Identificar les aplicacions que utilitzen PAM.
    b. Identificar els fitxers de configuració de PAM de les aplicacions i entendre el funcionament.
    c. Elements de la configuració: type, control, module i options. Identificar la funcionalitat de cada type.
    d. Entendre el funcionament dels controls bàsics: requisite, requiret, sufficient, optional, include i stack. identificar els altres controls avançats.
    e. Consultar la ubicació i el man dels modules pam_<nom>.so utilitzats en les configuracions PAM.
2. Exemples de configuracions PAM bàsiques
    a. Usant l'aplicació chfn i la seva configuració PAM identificar clarament els types auth i account.
    b. Utilitzar el mòdul pam_echo.so per fer el seguiment de la configuració PAM.
    c. Utilitzar journalctl per observar l'avaluació dels mòduls PAM usats en les configuracions.
    d. També amb chfn practicar les combinacions dels controls bàsics i dels mòduls: pam:permit.so, pam_deny.so, pam_unix.so, pam_suceed_if i pam_rootok.so
3. Arguments dels mòduls (més usuals)
    a. Identificar els arguments que es poden usar en els mòduls i provar-los.
    b. Opcions típiques: nullok, try_first_pass, use_first_pass, etc.
    c. Establir condicions amb pam_succeed_if segons sigui el uid, login, etc.
4. Exemples de configuracions del type: password
    a. Redefinir la política usada per modificar la contreassenya. Generar un nou PAM passwd.
    b. Practicar les opcions disponibles de pam_pwquality.so i de pam_cracklib.so.
5. Exemples de configuracions dels types: account i session:
    a. Establir limitacions d'accés segons el dia i hora de la setmana amb el mòdul pam_time.so.
    b. Crear el directori dels usuaris automàticament amb pam_mkhomedir.so.
    c. Muntar / Desmuntar automàticament recursos de disc (locals, tmpfs, nfs, samba, etc) en iniciar / tancar sessió, usant pam_mount.so
    d. Configuracions de pam_mount globals per tots els usuaris i/o configuracions particulars per a un usuari concret.
6. Exemples de funcionament dels controls include i substack.
7. Configuració global: system-auth
    a. Examinar el contingut de la configuració actual de system-auth.
    b. Manipulació del link de system-auth, system-auth-ac i creació de versions de configuració pròpies.

    c. Consulta de la configuració actual de authconfig.
    d. Generar una nova configuració d'autenticació amb authconfig-tui.
8. Pràctica: PAM + LDAP, configuració conjunta.
    a. Implementar un sistema d'autenticació PAM+LDAP usant el paquet: nss-pam-ldapd.
    b. Examinar el software instal·lat: executables, documentació i serveis. Serveis nscd i nslcd.
    c. Configuració de l'accés al servidor LDAP propi usant el dimoni nslcd. Configurar el base search i els criteris de recerca d'usuaris i grups en el DIT LDAP..
    d. Configurar la resolució de noms via nsswitch.conf indicant la precedència de resolucions per als serveis de: passwd, groups i hosts.
    e. Configurar un servei basant-se en la resolució ldap, per exemple chfn.
    f. Configurar el sistema complert per usar autenticació unix + ldap.
9. Monitoritzar el funcionament de la configuració PAM + LDAP
    a. Utilització de la ordre getent per examinar la resolució de passwd, groups i hosts segons la configuració actual.
    b. Usar wireshark per monitoritzar el seguiment de les comunicacions realitzades per procedir a l'autenticació de l'usuari.
    c. Usar journalctl per monitoritzar el seguiment de les comunicacions realitzades per procedir a l'autenticació de l'usuari.
    d. Ampliar la base de dades LDAP amb usuaris i grups i declarar els memberUid dels grups.
10. Pràctiques:
    a. Crear una aplicació python que utilitzi el mòdul python-PAM per autenticar a l'usuari que la vol utilitzar.
    b. Crear en python un mòdul PAM tipus pam_quiz.so que realitza preguntes a l'atzar com a mecanisme d'autenticació.
    c. Implementar: LDAP + NFS + PAM. Autenticar usuaris locals i de xarxa (LDAP) amb el muntatge automàtic dels seus homes d'un servidor NFS.
    d. Implementar: LDAP + NFS + SAMBA. Autenticar usuaris locals i de xarxa (LDAP) amb el muntatge automàtic dels seus homes d'un servidor SAMBA.

Documentació

Documentació de PAM a consultar:

- objectius-pam
- activitats_asix_m06_uf1_nf1_2015-2016 (fitxers exemples: dades config)
- HowTo-ASIX_PAM_2016-2017

Més documentació:

- Documentació: Matering Pluggable Authentication Modules
- Fitxers d'exemples de configuració PAM

Pàgines de manual:

- pam(8)
- pam.conf(5)
- pam_<module-name>(8)
    - pam_permit.so, pam_deny.so, pam_echo, pam_unix.so, pam_pwquality.so, pam_succeed_if.so, pam_mkhomedir.so, pam_mount.so, pam_ldap...
- pam_mount.conf(5)

Containers:

- edtasixm06/hostpam:base
- edtasixm06/hostpam:nfs
- edtasixm06/hostpam:smb
- edtasixm06/ldapserver:dataDB
- edtasixm06/nfsserver:pam
- edtasixm06/samba:pam

PAM Pluggable Authentication Modules

1. Exemples de configuració de PAM

Exemple del fitxer de configuració PAM del servei *chfn*: /etc/pam.d/chfn

```
#%PAM-1.0
auth          sufficient   pam_rootok.so
auth          include      system-auth
account       include      system-auth
password      include      system-auth
session       include      system-auth
```

Exemple del fitxer de configuració PAM del servei login: /etc/pam.d/login

```
#%PAM-1.0
auth             substack         system-auth
auth             include  postlogin
account           required        pam_nologin.so
account           include         system-auth
password         include          system-auth
# pam_selinux.so close should be the first session rule
session           required        pam_selinux.so close
session           required        pam_loginuid.so
session           optional        pam_console.so
# pam_selinux.so open should only be followed by sessions to be executed in the user
context
session           required        pam_selinux.so open
session           required        pam_namespace.so
session           optional        pam_keyinit.so force revoke
session           include         system-auth
session           include         postlogin
-session         optional         pam_ck_connector.so
```

Exemple del fitxer de configuració PAM del servei *passwd*: /etc/pam.d/passwd

```
#%PAM-1.0
auth           include    system-auth
account        include    system-auth
password    substack   system-auth
-password  optional    pam_gnome_keyring.so use_authtok
password    substack   postlogin
```

Exemple del fitxer de configuració PAM del fitxer *system-auth* que es genera automàticament amb l'eina authconfig i que serveix de include a molts dels altres fitxers de configuració de servei: /etc/pam.d/auth-config

```
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required    pam_env.so
auth        sufficient  pam_fprintd.so
auth        sufficient  pam_unix.so nullok try_first_pass
auth        requisite   pam_succeed_if.so uid >= 1000 quiet_success
auth        required    pam_deny.so

account     required    pam_unix.so
account     sufficient  pam_localuser.so
account     sufficient  pam_succeed_if.so uid < 1000 quiet
account     required    pam_permit.so

password    requisite   pam_pwquality.so try_first_pass local_users_only retry=3
authtok_type=
password    sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password    required    pam_deny.so

session     optional    pam_keyinit.so revoke
session     required    pam_limits.so
-session    optional    pam_systemd.so
session     [success=1 default=ignore] pam_succeed_if.so service in crond quiet use_uid
session     required    pam_unix.so
```

2. zFuncionament de PAM

Documentació extreta de la pàgina de manual "*man pam*".

> Linux-PAM is a system of libraries that handle the authentication tasks of applications (services) on the system. The library provides a stable general interface (Application Programming Interface - API) that privilege granting programs (such as login(1) and su(1)) defer to perform standard authentication tasks.
>
> When a PAM aware privilege granting application is started, it activates its attachment to the PAM-API. This activation performs a number of tasks, the most important being the reading of the configuration file(s): /etc/pam.conf. Alternatively, this may be the contents of the /etc/pam.d/ directory. The presence of this directory will cause Linux-PAM to ignore /etc/pam.conf.

Cada línia de configuració (anomenada regla) en el fitxer PAM d'un servei te la forma:

```
type control module-path module-arguments
```

On cada *type* correspon a un dels elements seguents:

> **authentication** - authenticate a user and set up user credentials. Typically this is via some challenge-response request that the user must satisfy: if you are who you claim to be please enter your password. Not all authentications are of this type, there exist

hardware based authentication schemes (such as the use of smart-cards and biometric devices), with suitable modules, these may be substituted seamlessly for more standard approaches to authentication - such is the flexibility of Linux-PAM.

account - provide account verification types of service: has the user's password expired?; is this user permitted access to the requested service?

password - this group's responsibility is the task of updating authentication mechanisms. Typically, such services are strongly coupled to those of the auth group. Some authentication mechanisms lend themselves well to being updated with such a function. Standard UN*X password-based access is the obvious example: please enter a replacement password.

session - this group of tasks cover things that should be done prior to a service being given and after it is withdrawn. Such tasks include the maintenance of audit trails and the mounting of the user's home directory. The session management group is important as it provides both an opening and closing hook for modules to affect the services available to a user.

Per a cada control d'un servei es poden especificar diverses regles PAM. A cada regla es poden aplicar els següents *control*:

Documentació extreta del "*man pam.conf*"

For the simple (historical) syntax valid control values are:

required
failure of such a PAM will ultimately lead to the PAM-API returning failure but only after the remaining stacked modules (for this service and type) have been invoked.

requisite
like required, however, in the case that such a module returns a failure, control is directly returned to the application or to the superior PAM stack. The return value is that associated with the first required or requisite module to fail. Note, this flag can be used to protect against the possibility of a user getting the opportunity to enter a password over an unsafe medium. It is conceivable that such behavior might inform an attacker of valid accounts on a system. This possibility should be weighed against the not insignificant concerns of exposing a sensitive password in a hostile environment.

sufficient
if such a module succeeds and no prior required module has failed the PAM framework returns success to the application or to the superior PAM stack immediately without calling any further modules in the stack. A failure of a sufficient module is ignored and processing of the PAM module stack continues unaffected.

optional
the success or failure of this module is only important if it is the only module in the stack associated with this service+type.

include

include all lines of given type from the configuration file specified as an argument to this control.

substack
include all lines of given type from the configuration file specified as an argument to this control. This differs from include in that evaluation of the done and die actions in a substack does not cause skipping the rest of the complete module stack, but only of the substack. Jumps in a substack also can not make evaluation jump out of it, and the whole substack is counted as one module when the jump is done in a parent stack. The reset action will reset the state of a module stack to the state it was in as of beginning of the substack evaluation.

A part de la sintaxis de control clàssica hi ha la mes avançada (extret del *man pam.conf*):

For the more complicated syntax valid control values have the following form:
[value1=action1 value2=action2 ...]

ignore
when used with a stack of modules, the module's return status will not contribute to the return code the application obtains.

bad
this action indicates that the return code should be thought of as indicative of the module failing. If this module is the first in the stack to fail, its status value will be used for that of the whole stack.

die
equivalent to bad with the side effect of terminating the module stack and PAM immediately returning to the application.

ok
this tells PAM that the administrator thinks this return code should contribute directly to the return code of the full stack of modules. In other words, if the former state of the stack would lead to a return of PAM_SUCCESS, the module's return code will override this value. Note, if the former state of the stack holds some value that is indicative of a modules failure, this 'ok' value will not be used to override that value.

done
equivalent to ok with the side effect of terminating the module stack and PAM immediately returning to the application.

N (an unsigned integer)
equivalent to ok with the side effect of jumping over the next N modules in the stack. Note that N equal to 0 is not allowed (and it would be identical to ok in such case).

reset
clear all memory of the state of the module stack and start again with the next stacked module.

La següent taula mostra les equivalències entre els controls clàssics i els de parell de valors (extret del *man pam.conf*):

Each of the four keywords: required; requisite; sufficient; and optional, have an equivalent expression in terms of the [...] syntax. They are as follows:

required
[success=ok new_authtok_reqd=ok ignore=ignore **default=bad**]

requisite
[success=ok new_authtok_reqd=ok ignore=ignore **default=die**]

sufficient
[**success=done** new_authtok_reqd=done default=ignore]

optional
[**success=ok** new_authtok_reqd=ok default=ignore]

Rols del PAM

Podeu consultar la documentació de PAM a la web "The Linux-PAM Guides", on s'observen els tres rols existents:

- The System Administrator's Guide.
  Administrar PAM en un sistema GNU/Linux.

- The Module Writer's Guide.
  Escriure mòduls pam, programar nous mòduls pam_<nom>.so que facin noves funcionalitats. Per exemple autenticar els usuaris segons la mida del seu nas.

- The Application Developpers Guide.
  Desenvolupar aplicacions que siguin PAM Aware, que utilitzin PAM: Per exemple fer programes en C, Java o Python que facin autenticació d'usuaris basant-se en PAM.

Arquitectura i fitxers

Mòduls
Les aplicacions PamAware estan linkades amb PAM. Els mòduls PAM són llibreries dinàmiques "***Shared Objects***" com per exemple ***pam_mount.so***, que usuelment es troben a "***/usr/lib64/security***".

```
$ locate pam_mount.so
/usr/lib64/security/pam_mount.so
```

Llistat dels mòduls PAM instal·lats en un sistema d'exemple:

```
$ ls /usr/lib64/security/
pam_access.so          pam_krb5.so            pam_selinux.so
pam_cap.so             pam_lastlog.so         pam_sepermit.so
pam_chroot.so          pam_limits.so          pam_shells.so
pam_console.so         pam_listfile.so        pam_sss.so
pam_cracklib.so        pam_localuser.so       pam_stress.so
pam_debug.so           pam_loginuid.so        pam_succeed_if.so
pam_deny.so            pam_mail.so            pam_systemd.so
pam_echo.so            pam_mkhomedir.so       pam_tally2.so
pam_env.so             pam_motd.so            pam_time.so
pam_exec.so            pam_mount.so           pam_timestamp.so
pam_faildelay.so       pam_namespace.so       pam_tty_audit.so
pam_faillock.so        pam_nologin.so         pam_umask.so
pam_filter             pam_passwdqc.so        pam_unix_acct.so
pam_filter.so          pam_permit.so          pam_unix_auth.so
pam_fprintd.so         pam_pkcs11.so          pam_unix_passwd.so
pam_ftp.so             pam_postgresok.so      pam_unix_session.so
pam_gnome_keyring.so   pam_pwhistory.so       pam_unix.so
pam_group.so           pam_pwquality.so       pam_userdb.so
pam_issue.so           pam_rhosts.so          pam_warn.so
pam_keyinit.so         pam_rootok.so          pam_wheel.so
pam_krb5               pam_securetty.so       pam_xauth.so
pam_krb5afs.so         pam_selinux_permit.so
```

Configuració

La configuració de pam pot residir en dues ubicacions seguint la típica estratègia de un fitxer de configuració global o un directori de configuracions particulars:

- /etc/pam.conf → fitxer de configuració global
- /etc/pam.d → directori de configuracions de cada servei

```
ls /etc/pam.d
atd             chsh            fingerprint-auth    gdm-password    kscreensaver    nntp            polkit-1        ppp             sieve
                sshd                su-l                vlock
authconfig      config-util     fingerprint-auth-ac gdm-pin         ksu             other           pop             remote
smartcard-auth  sssd-shadowutils system-auth        vmtoolsd
authconfig-gtk  crond           gdm-autologin       gdm-smartcard   lmtp            passwd          postgresql
runuser         smartcard-auth-ac su                system-auth-ac  xserver
authconfig-tui  csync           gdm-fingerprint     imap            login           password-auth   postlogin       runuser-l       smtp
                sudo                system-config-authentication
chfn            cups            gdm-launch-environment  kcheckpass  mupdate         password-auth-ac postlogin-ac   setup
smtp.sendmail   sudo-i          systemd-user
```

Per a cada servei PAM Aware hi ha un fitxer amb regles específiques, o be s'apliquen les regles generals que hi ha a others.

Aplicació PAM Aware

Com saber si una aplicació és PAM Aware? El mecanisme més fàcil és usar l'ordre ldd per examinar si l'executable de l'aplicació està lligat a les llibreries PAM.

```
# ldd /usr/bin/chfn
    linux-vdso.so.1 (0x00007ffd11d58000)
    libuser.so.1 => /lib64/libuser.so.1 (0x00007f965d127000)
    libpam.so.0 => /lib64/libpam.so.0 (0x00007f965cf18000)
    libpam_misc.so.0 => /lib64/libpam_misc.so.0 (0x00007f965cd13000)
    libselinux.so.1 => /lib64/libselinux.so.1 (0x00007f965caec000)
    libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f965c8d0000)
    libc.so.6 => /lib64/libc.so.6 (0x00007f965c50d000)
    libgmodule-2.0.so.0 => /lib64/libgmodule-2.0.so.0 (0x00007f965c309000)
    libgobject-2.0.so.0 => /lib64/libgobject-2.0.so.0 (0x00007f965c0b7000)
    libglib-2.0.so.0 => /lib64/libglib-2.0.so.0 (0x00007f965bda8000)
    libcrypt.so.1 => /lib64/libcrypt.so.1 (0x00007f965bb72000)
    libaudit.so.1 => /lib64/libaudit.so.1 (0x00007f965b94a000)
    libdl.so.2 => /lib64/libdl.so.2 (0x00007f965b745000)
    libpcre.so.1 => /lib64/libpcre.so.1 (0x00007f965b4d2000)
    /lib64/ld-linux-x86-64.so.2 (0x0000564952599000)
    libffi.so.6 => /lib64/libffi.so.6 (0x00007f965b2c9000)
    libfreebl3.so => /lib64/libfreebl3.so (0x00007f965b0c6000)
    libcap-ng.so.0 => /lib64/libcap-ng.so.0 (0x00007f965aec0000)
```

```
# ldd /usr/bin/login | grep "pam"
    libpam.so.0 => /lib64/libpam.so.0 (0x00007f4e99f8f000)
    libpam_misc.so.0 => /lib64/libpam_misc.so.0 (0x00007f4e99d8b000)
```

Configuració other

Una configuració a tenir en compte és la anomenada *other*, que no correspon a cap servei concret anomenat així, sinó que s'aplica quan un servei és PAM Aware i no té una configuració específica. Usualment és un fitxer de tancar la porta que ho denega tot.

LListat de la configuració del 'servei' other:

```
# cat /etc/pam.d/other
#%PAM-1.0
auth     required       pam_deny.so
account  required       pam_deny.so
password required       pam_deny.so
session  required       pam_deny.so
```

Configuracions del sistema

Hi ha fitxers amb configuracions globals que s'utilitzen com a includes d'altres configuracions. Entre elles les més conegudes són:

- system-auth
- password-auth

Podeu observar que en realitat aquests fitxers són links simbòlics que apunten als fitxers reals:

- system-auth-ac

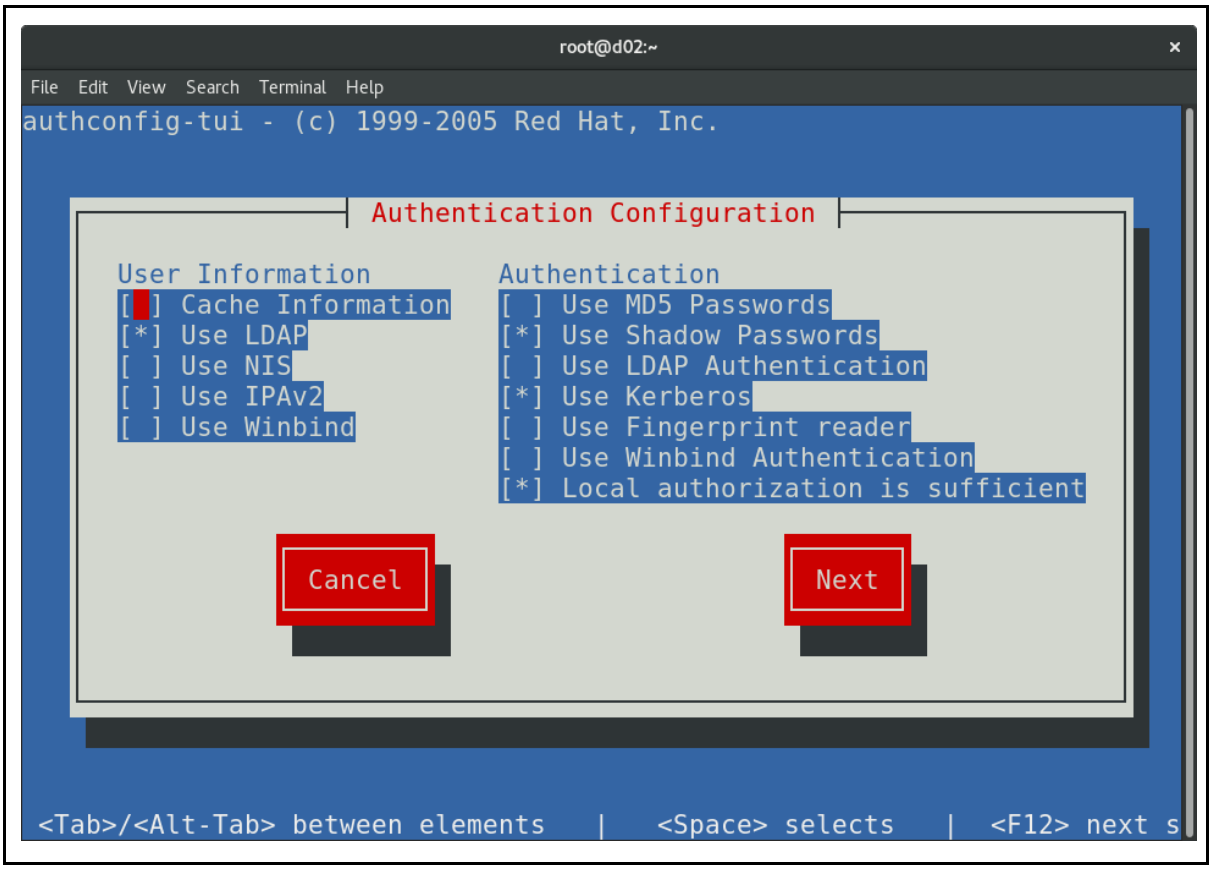
- password-auth-ac

La mecànica és crear els fitxers ‘reals’ i poder-ne tenir versions diferents, per exemple system-auth-edt. Generar llavors que el symbolic link apunti al fitxer de configuració particular que calgui. És una manera de poder tenir o provar combinacions diferents sense perdre els fitxers en fer les modificacions.

Generació automàtica de la configuració

Qui ha generat els fitxers de PAM system-auth-ac? i password-asuth-ac? S’han generat automàticament en fer la instal·lació del sistema. L’eina ***authconfig*** genera i retoca els fitxers necessàris per configurar l’autenticació.

En el nostre cas particular en instal3lar a l’aula es va usar l’applet semigràfic authconfig-tui per indicar el tipus d’autenticació, que s’utilitza LDAP i Kerberos.



Aquesta eina en finalitzar reesciu tots els fitxers realacionats amb l’autenticació. Podem observar quin és el llistat actual d’opcions de configuració amb:

```
# authconfig --test
caching is disabled
nss_files is always enabled
nss_compat is disabled
nss_db is disabled
nss_hesiod is disabled
 hesiod LHS = ""
 hesiod RHS = ""
nss_ldap is enabled
```

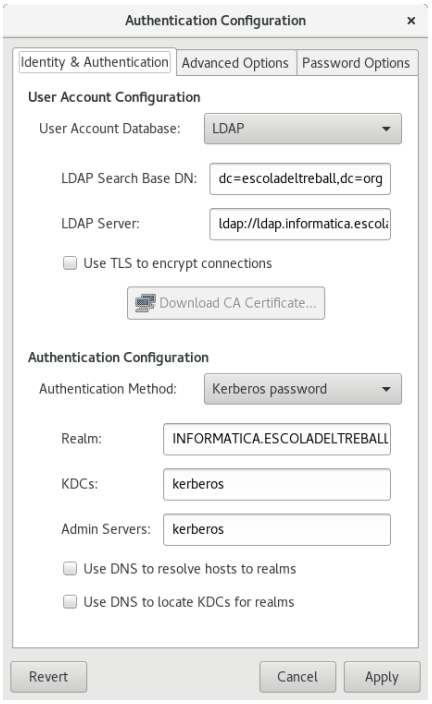
```
 LDAP+TLS is disabled
 LDAP server = "ldap://ldap.informatica.escoladeltreball.org/"
 LDAP base DN = "dc=escoladeltreball,dc=org"
nss_nis is disabled
 NIS server = ""
 NIS domain = ""
nss_nisplus is disabled
nss_winbind is disabled
 SMB workgroup = "SAMBA"
 SMB servers = ""
 SMB security = "user"
 SMB realm = ""
 Winbind template shell = "/bin/false"
 SMB idmap range = "16777216-33554431"
nss_sss is enabled by default
nss_wins is disabled
nss_mdns4_minimal is enabled
myhostname is enabled
DNS preference over NSS or WINS is disabled
pam_unix is always enabled
 shadow passwords are enabled
 password hashing algorithm is sha512
pam_krb5 is enabled
 krb5 realm = "INFORMATICA.ESCOLADELTREBALL.ORG"
 krb5 realm via dns is disabled
 krb5 kdc = "kerberos"
 krb5 kdc via dns is disabled
 krb5 admin server = "kerberos"
pam_ldap is disabled
 LDAP+TLS is disabled
 LDAP server = "ldap://ldap.informatica.escoladeltreball.org/"
 LDAP base DN = "dc=escoladeltreball,dc=org"
 LDAP schema = "rfc2307"
pam_pkcs11 is disabled
 use only smartcard for login is disabled
 smartcard module = "coolkey"
 smartcard removal action = "Ignore"
pam_fprintd is disabled
pam_ecryptfs is disabled
pam_winbind is disabled
 SMB workgroup = "SAMBA"
 SMB servers = ""
 SMB security = "user"
 SMB realm = ""
pam_sss is disabled by default
 credential caching in SSSD is enabled
 SSSD use instead of legacy services if possible is enabled
IPAv2 is disabled
IPAv2 domain was not joined
 IPAv2 server = ""
 IPAv2 realm = ""
 IPAv2 domain = ""
pam_pwquality is enabled (try_first_pass local_users_only retry=3 authtok_type=)
pam_passwdqc is disabled ()
pam_access is disabled ()
pam_mkhomedir or pam_oddjob_mkhomedir is enabled (skel=/etc/skel umask=0066)
Always authorize local users is enabled ()
Authenticate system accounts against network services is disabled
```

De fet l'ordre anterior ho llistat tot, les opcions de configuració específicament establertes es desen al fitxer "***/etc/sysconfig/authconfig***":

```
# cat /etc/sysconfig/authconfig
PASSWDALGORITHM=sha512
USENIS=no
FORCELEGACY=no
USEHESIOD=no
USEMKHOMEDIR=yes
```

```
IPADOMAINJOINED=no
FORCESMARTCARD=no
USESMARTCARD=no
WINBINDKRB5=no
USELDAPAUTH=no
USEDB=no
USEKERBEROS=yes
USESSSD=yes
USEWINBIND=no
USESSSDAUTH=no
USELOCAUTHORIZE=yes
IPAV2NONTP=no
USEPASSWDQC=no
USEPWQUALITY=yes
USESHADOW=yes
USEPAMACCESS=no
USEWINBINDAUTH=no
USEIPAV2=no
USESYSNETAUTH=no
USEFPRINTD=no
USEECRYPTFS=no
USELDAP=yes
CACHECREDENTIALS=yes
```

L'eina semigràfica es deprecated i es recomana la utilització de la eina gràfica ***system-config-authentication*** que executa l'applet ***authconfig-gtk***.



De la pàgina del man de authconfig podem observar els fitxers relacionats:

```
FILES
        /etc/sysconfig/authconfig
        Used to track whether or not particular authentication mechanisms are enabled.  Currently includes variables
named USESHADOW, USEMD5, USEKERBEROS, USELDAPAUTH, USESMBAUTH, USEWINBIND, USEWIN-
        BINDAUTH, USEHESIOD, USENIS, USELDAP, and others.
        /etc/passwd
```

```
        /etc/shadow
        Used for shadow password support.
        /etc/yp.conf
        Configuration file for NIS support.
        /etc/sysconfig/network
        Another configuration file for NIS support.
        /etc/ldap.conf
        /etc/nss_ldap.conf
        /etc/pam_ldap.conf
        /etc/nslcd.conf
        /etc/openldap/ldap.conf
        Used to configure nss_ldap, pam_ldap, nslcd, and the OpenLDAP library. Only the files already existing on the
system are modified.
        /etc/krb5.conf
        Used to configure Kerberos 5.
        /etc/hesiod.conf
        Used to configure Hesiod.
        /etc/samba/smb.conf
        Used to configure winbind authentication.
        /etc/nsswitch.conf
        Used to configure user information services.
        /etc/login.defs
        Used to configure parameters of user accounts (minimum UID of a regular user, password hashing algorithm).
        /etc/pam.d/system-auth
        Common PAM configuration for system services which include it using the include directive. It is created as
symlink and not relinked if it points to another file.
        /etc/pam.d/system-auth-ac
        Contains  the actual PAM configuration for system services and is the default target of the /etc/pam.d/system-auth
symlink. If a local configuration of PAM is created (and symlinked from sys-
        tem-auth file) this file can be included there.
```

Es pot observar la configuració actual "***last***" en el directori ***/var/lib/authconfig***:

```
# ll  /var/lib/authconfig/last/
total 32
-rw-r--r--. 1 root root  890 Jul  5  2017 fingerprint-auth-ac
-rw-rw-r--. 1 root root  607 Jul  5  2017 krb5.conf
-rw-r--r--. 1 root root  432 Jul  5  2017 openldap.conf
-rw-rw-r--. 1 root root 1341 Jul  5  2017 password-auth-ac
-rw-r--r--. 1 root root  326 Jul  5  2017 postlogin-ac
-rw-r--r--. 1 root root 1047 Jul  5  2017 smartcard-auth-ac
-rw-------. 1 root root  563 Jul  5  2017 sssd.conf
-rw-rw-r--. 1 root root 1341 Jul  5  2017 system-auth-ac
```

Es poden desar backups de la configuració amb noms particulars per poder-los restaurar:

```
# authconfig --savebackup edt

#  ll /var/lib/authconfig/
drwxr-xr-x. 2 root root 4096 Jan 23 10:17 backup-edt
drwxr-xr-x. 2 root root 4096 Jul  5  2017 last
```

3. Afegir comentaris amb el mòdul pam_echo.so

Amb el mòdul **pam_echo.so** es poden afegir comentaris a fitxers clau per tal d'observar quin dels serveis es crida segons l'acció que es realitza. Usarem l'ordre *chfn* per mostrar que fa ús dels serveis *auth* i *account*. L'ordre *login* per mostrar que utilitza els serveis *auth*, *account* i *session*. I finalment l'ordre *passwd* per observar que fa ús del servei *password*.

Exemple del fitxer de configuració PAM del servei *chfn*: /etc/pam.d/chfn

```
#%PAM-1.0
auth        optional    pam_echo.so [type:auth rhost: %H lhost: %h service: %s terminal:
%t ruser:%U luser: %u]
auth        sufficient  pam_rootok.so
auth        include     system-auth
account     optional    pam_echo.so [type:account rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
account     include     system-auth
password    optional    pam_echo.so [type:password rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
password    include     system-auth
session     optional    pam_echo.so [type:session rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
session     include     system-auth
```

Exemple del fitxer de configuració PAM del servei login: /etc/pam.d/login

```
#%PAM-1.0
auth        optional    pam_echo.so [type:auth rhost: %H lhost: %h service: %s terminal:
%t ruser:%U luser: %u]
auth            substack        system-auth
auth            include  postlogin
account     optional    pam_echo.so [type:account rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
account         required        pam_nologin.so
account         include         system-auth
password    optional    pam_echo.so [type:password rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
password      include         system-auth
# pam_selinux.so close should be the first session rule
session     optional    pam_echo.so [type:session rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
session         required        pam_selinux.so close
session         required        pam_loginuid.so
session         optional        pam_console.so
# pam_selinux.so open should only be followed by sessions to be executed in the user
context
session         required        pam_selinux.so open
session         required        pam_namespace.so
session         optional        pam_keyinit.so force revoke
```

```
session         include         system-auth
session         include         postlogin
-session        optional        pam_ck_connector.so
```

Exemple del fitxer de configuració PAM del servei *passwd*: /etc/pam.d/passwd

```
#%PAM-1.0
auth          optional    pam_echo.so [type:auth rhost: %H lhost: %h service: %s terminal:
%t ruser:%U luser: %u]
auth          include     system-auth
account       optional    pam_echo.so [type:account rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
account       include     system-auth
password      optional    pam_echo.so [type:password rhost: %H lhost: %h service: %s
terminal: %t ruser:%U user: %u]
password    substack    system-auth
-password   optional    pam_gnome_keyring.so use_authtok
password    substack    postlogin
```

4. Exemples PAM: usant el servei chfn

Exemples bàsics

Els següents són exemples per entendre el funcionament del PAM. Es tracta de fer modificacions al fitxer del servei ***chfn*** (fitxer */etc/pam.d/chfn*) i observar com afecta al servei.
****Atenció**** per comprovar el funcionament de les modificacions del servei chfn cal executar l'ordre chfn des d'un usuari no root. Sembla que la pròpia ordre chfn identifica si l'usuari és el UID 0 i actua diferent que per a un usuari no privilegiat.
****Atenció**** Cada vegada que s'executa l'ordre chfn per modificar les dades de l'usuari es obligatori canviar almenys una dada, sinó chfn descarta l'actualització i no fa ús del PAM (si no hi ha res a modificar és que no cal fer res!).

Es recomana disposar de tres usuaris no administradors (per exemple pere, marta i julia), identificar el seu UID i fer un dels usuaris passwordless.

Exemple 01 L'usuari pot modificar sempre el seu chfn sense password

Usualment un usuari no privilegit quan modifica el seu finger amb chfn és requerit (al final i si ha fet modificacions) d'identificar-se amb el seu password. Si l'usuari és root la pròpia ordre s'estalvia aquest pas i permet el canvi automàticament.

La següent configuració actualitza el chfn automàticament sense demanar password.

```
# PAM exemple-01
auth          optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth          sufficient  pam_permit.so
```

```
account    optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account    sufficient  pam_permit.so
```

Exemple 02 L'usuari no pot modificar mai el chfn.

Aquest exemple no permet modificar el chfn, tampoc es demana en cap moment el password a l'usuari per identificar-se.

```
# PAM exemple-01
auth       optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth       sufficient  pam_deny.so
account    optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account    sufficient  pam_permit.so
```

Exemple 03 Es permet si és un usuari unix valid

En aquest exemple depèn de si es tracta d'un usuari unix vàlid o no. Es demana el pasword (el mòdul pam_unix.so el demana) i si és correcte es realitza l'actualització de les dades de chfn.
El modul pam_unix.so (sense arguments) demana sempre el password.

```
# PAM exemple-01
auth       optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth       sufficient  pam_unix.so
account    optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account    sufficient  pam_permit.so
```

Exemple 04 Es permet sempre el canvi, demanant sempre el password

En aquest exemple pam_permit.so permet sempre el chfn, i es tracta d'un mòdul required.
El modul pam_unix.so demana sempre el password (sense arguments que ho contradiguin) i és de tipus sufficient.

```
# PAM exemple-01
auth       optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth       required    pam_permit.so
auth       sufficient  pam_unix.so
account    optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account    sufficient  pam_permit.so
```

Si l'usuari és un usuari unix vàlid es compleix el sufficient i el required i es fa el chfn. Si l'usuari no s'identifica apropiadament al pam_unix.so el suffient falla. Quan sufficient falla la regla es descarta, no compta ni com a success ni com a fail. per tant com que el required del pam_permit.so és success, es realitza sempre el chfn (encara que l'usuari no s'hagi validat apropiadament!).

Exemple 05 L'ordre de les regles és significatiu

Observeu que el mateix exemple escrit amb els 'control' a l'inrevés perd tot el sentit:

```
# PAM exemple-01
auth       optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth       sufficient  pam_permit.so
```

```
auth        required    pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient   pam_permit.so
```

La regla sufficient amb pam_permit.so és success i el comportament de suffient indica que es pot finalitzar l'execució de les regles del stack auth. De manera que finalitza PAM permetent fer el chfn i sense demanar el password ja que la línia pam_unix.so no s'arriba a processar.

Exemple 06 Complir múltiples condicions (I)

Exemple amb tres condicions:

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required     pam_permit.so
auth        required    pam_unix.so
auth        required     pam_succeed_if.so debug uid > 1000
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient   pam_permit.so
```

Per poder modificar el chfn es requereix que sigui success les tres regles, si una d'elles falla no es podrà canviar. Però s'avaluen totes tres i sempre es demana el password a l'usuari. Evidentment la primera regla és 'de florero' ja que no afecta per res. La segona demana al password i valida que sigui un usuari unix. Tant si és valid com si no també es verifica la tercera regla que requereix que sigui un usuari amb un uid superior a 1000.

Es poden consultar els logs generats per aquestes accions amb la comanda:

```
# journalctl -b -r
# journalctl -f
```

Exemple 07 Complir múltiples condicions (II)

Verificar el compte unix i permetre només fer el chfn a l'usuari pere,

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_unix.so
auth        required    pam_succeed_if.so debug user = pere
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient pam_permit.so
```

Exemple 08 Usar pam_deny.so

En aquest exemple el mòdul pam_deny.so no genera cap efecte i l'usuari podrà fer chfn o no en funció de si s'identifica com un usuari unix valid o no.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        sufficient  pam_deny.so
auth        required     pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
```

```
account    sufficient    pam_permit.so
```

L'efecte del control sufficient quan és fail no és tingut en compte, i es passa a evaluar la següent regla.

Exemple 09 Usar pam_deny.so com a requisite

Evidenment l'avaluació de pam_deny.so com a requisite fallarà sempre generant un fail, i per tant el resultat global final serà que no es permet fer el chfn. Però en ser requisite es continuen avaluant la resta de regles, de manera que es demana el password a l'usuari.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_deny.so
auth        required     pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient    pam_permit.so
```

Independentment de que es validi apropiadament com un usuari unix no es permetrà l'actualització de les dades de finger perquè no s'ha complert el require.
Observar que si la línia del pam_unix.so es modifica a sufficient o optional, l'efecte és el mateix.

Exemple 10 L'efecte de requisite usant pam_deny.so

Usar com a requisite (no required) un modul que s'avalua a fail provoca la finalització de l'stack de regeles. Així usar pam_deny.so com a requisite provoca que es finalitzi sense poder modificar el finger i sense que s'executi la regla del pam_unix.so.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        requisite   pam_deny.so
auth        required     pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient    pam_permit.so
```

L'usuai pot endevinar que ha fallat una regla abans del pam_unix.so perquè no se li ha arribat a demanar el password.
Aquest és un dels motius d'usar required, per avaluar totes les regles i no donar pistes de quina és la que falla.

Exemple 11 L'efecte del control optional

Els mòduls amb el control optional no afecten al resultat final a no ser que siguin l'únic mòdul a avaluar.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        optional    pam_deny.so
auth        required     pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient    pam_permit.so
```

Exemple 12 Sufficient no és suficient si hi ha required previs que han fallat

Que l'autenticació pam_unix (de tipus suficient) sigui success no generara una avaluació positiva perquè hi ha un required previ que ha fallat.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_deny.so
auth        sufficient  pam_uznix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

Exemple 13 Evaluar diversos suffficient

En l'exemple següent els usuaris unix identificats poden canviar les dades de finger.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        sufficient  pam_deny.so
auth        sufficient  pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

Cal recordar que el control sufficient si falla es descarta, no es te en compte.

Exemples avançats

Exemple 14 Un usuari sense password no pot modificar el finger

El modul pam_unix.so per defecte no permet els usuaris passwordless (sense password). Si un usuari és passwordless i intenta modificar el seu finger amb aquesta configuracio del servei chfn no podrà.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

Exemple 15 Permetre validar usuaris passwordless amb nullok

L'opció **nullok** del mòdul pam_unix.so permet validar usuaris sense password.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_unix.so nullok
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

Exemple 16 Combinar un conjunt de regles

Aquest és un exemple on es combinen un conjunt de regles auth per permetre modificar el finger d'un usuari amb chfn.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        sufficient  pam_rootok.so
auth        required    pam_unix.so nullok
auth        required    pam_succeed_if.so uid > 1000
#auth        required    pam_succeed_if.so user != "pere"
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_unix.so try_first_pass
```

Exemple 17 Observar l'efecte de pam-unix.so amb try_first_pass

Com s'ha observat dels exemples precedents el mòdul pam_unix.so demana a l'usuari que s'identifiqui amb el nom d'usuari i una contrassenya. Si es posen dos mòduls pam_unix.so seguts es demanarà dues vegades la identificació de l'usuari (totalment absurd però permet mostrar com es comporta!).

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_unix.so
auth        required    pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

Ara bé, si el segon cop que cridem pam_unix.so (repetim que és del tot innecessari, es només per aprendre què fa try_first_pass) li passem l'opció **try_first_pass** observem que només cal validar-se un cop (tot i que ho fa internament dues vegades).

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_unix.so
auth        required    pam_unix.so try_first_pass
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

Aquest opció signfica que si en alguna regla anterior ja s'ha demanat el password no es torni a demanar. Ara bé, si no s'ha demanat encara, llavors cal demanar-lo.

Exemple 18 Observar l'efecte de pam-unix.so amb l'opció use_first_pass

L'opcio **user_first_pass** indica al pam_unix.so que per validar l'usuari cal usar el password que tingui en memòria usat en alguna regla anterior. Així si en una regla anterior de l'sctack ja s'ha demanat el password s'utilitza aquest ja existent. Ara bé, si no s'ha demanat cap password anteriorment no es demana i per tant no es pot realitzar l'autenticació.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
```

```
auth        required    pam_unix.so
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_unix.so use_first_pass
```

L'exemple anterior permet validar apropiadament l'usuari i li permet modificar el seu finger. En el primer pam_unix.so del auth es requereix identificar l'usuari amb un passwd i en el account fa us d'aquest mateix password.

El següent exemple utilitza en el primer mòdul pam_unix.so l'opció use_first_pass i com que no se n'ha introduït cap prèviament, l'autenticació fallarà.

```
# PAM exemple-01
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        required    pam_unix.so use_first pass
account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     sufficient  pam_permit.so
```

5. Exemples PAM: usant el servei passwd

Exemple 19 Generar un fitxer password complert

El servei passwd que es crida en executar l'ordre password es pot observar en el primer llistat d'aquest document, conte una regla substack que incorpora les regles del fitxer syste-auth.
Un primer exemple és copiar a mà (cut+paste) aquestes regles en el propi fitxer del servei passwd per poder fer-ne modificacions i estudiar el seu comportament. Podem veure el següent llistat de /etc/pam.d/passwd amb les línies del system-auth inserides:

```
#%PAM-1.0
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        include     system-auth

account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     include     system-auth

password    optional    pam_echo.so [type:passwod lhost: %h service: %s terminal: %t luser: %u] user: %u]
password    requisite   pam_pwquality.so try_first_pass local_users_only retry=3
authtok_type=
password    sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password    required    pam_deny.so
-password   optional    pam_gnome_keyring.so use_authtok
password    substack    postlogin
```

El mòdul pwquality.so és l'encarregat de validar que el nou password compleixi uns requeriments concrets (longitud, varietat de dígits/lletres/símbols, diccionari, no similar, etc). Aquest mòdul permet només usuaris locals i permet un màxim de tres intents per posar un nou password (a vegades són pocs!). Per defecte no es permeten passswords en blanc.

Fixeu-vos que utilitza l'opció try_first_pass que indica que ha de demanar el password actual si encara no s'ha fet. El mòdul següent pam_unix.so també utilitza aquesta opció de manera que no el torna a demanar.

Exemple 20 Usar el mòdul pam_pwquality.so

Consultant el man pam_pwquality.so es poden observar les opcions que permet aquest mòdul per exigir el format que ha de tenir el password. Aquest és només un dels mòduls existents de validació de format dels passwords, n'hi ha molts més (cpam_cracklib.so, etc).

L'exemple següent demana sempre el password actual abans de fer la modificació, permet establir passwords en blanc (passwordless) i només permet un sol intent d'establir el password, si no és prou bo… a tornar-hi!

```
#%PAM-1.0
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        include     system-auth

account     optional    pam_echo.so [type:account lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     include     system-auth

password    optional    pam_echo.so [type:password  lhost: %h service: %s terminal: %t luser: %u] user: %u]
password    requisite   pam_pwquality.so nullok  retry=1 authtok_type=
password    sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password    required    pam_deny.so
-password   optional    pam_gnome_keyring.so use_authtok
password    substack    postlogin
```

Exemple 21 Modificar un password sense regles!

A vegades ens emprenyem de tant difícil com és de posar un password i de lo fàcil que ho fa root, oi? I si fem que els usuaris també puguin posar els passwords que vulguin? Aquest és un exemple, fixeu-vos que no hi ha pam_pwquality.so:

```
#%PAM-1.0
auth        optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth        include     system-auth

account     optional    pam_echo.so [type:account  lhost: %h service: %s terminal: %t luser: %u] user: %u]
account     include     system-auth

password    optional    pam_echo.so [type:password  lhost: %h service: %s terminal: %t luser: %u] user: %u]
password    sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password    required    pam_deny.so
-password   optional    pam_gnome_keyring.so use_authtok
password    substack    postlogin
```

Exemple 22 Usar l'opció use_first_pass

El modificador use_first_pass indica al modul que no demani el password actual sinó que utilitzi el que ja s'ha entrat prèviament, si no se n'ha entrat cap doncs no n'hi ha cap!.

Usant el codi de l'exemple 17 observem que si al mòdul pam_pwquality.so se li posa aquest argument llavors el modul no tindria quin password ha d'usar per validar l'usuari, perquè no n'hi ha cap d'anterior.

En canvi si en el modul pam_unix.so es canvia el try_first_pass per un use_first _pass funcionaria si en el modul anterior pam_pwquality.so s'ha demanat el password (deixant-lo com estava en l'exercici 17).

```
#%PAM-1.0
auth          optional    pam_echo.so [type:auth lhost: %h service: %s terminal: %t luser: %u]
auth          include     system-auth

account       optional    pam_echo.so [type:account  lhost: %h service: %s terminal: %t luser: %u] user: %u]
account       include     system-auth

password      optional    pam_echo.so [type:password  lhost: %h service: %s terminal: %t luser: %u] user: %u]
password      requisite   º.so nullok  retry=1 authtok_type=
password      sufficient  pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password      required    pam_deny.so
-password     optional    pam_gnome_keyring.so use_authtok
password      substack    postlogin
```

6. Exemples PAM: usant el servei account

- usuaris amb el compte bloquejat
- usuaris amb el compte expirat
- usuaris amb el compte en warning

- usar pam_time.so per establir horaris de connexió.

```
account  optional     pam_echo.so [type:account rhost: %H lhost: %h service: %s terminal: %t ruser:%U user: %u]
account  optional     pam_echo.so [entra al time]
account  sufficient   pam_time.so debug
account  optional     pam_echo.so [surt del time]
account  required     pam_deny.so
```

Cal configurar el fitxer /etc/security/time.conf

```
/etc/security/time.conf
        chfn;*;marta;Al0000-2400
```

/etc/security/time.conf
chfn;*;marta;Mo0800-1400

- usar pam_time.so per establir horaris de connexió. Per exemple fer:

l'usuai pere nomes es pot connectar els dies entre setmana (workingdays)
l'usuaria marta es pot connectar tots els dies per la tarda

/etc/security/time.conf

7. Exemples PAM: usant el servei session

- pam_mkhomedir.so

```
session     optional     pam_echo.so [ ara fara el homedir]
session     optional     pam_mkhomedir.so skel=/etc/skel
```

- pam_mount.so

```
session     optional     pam_echo.so [ ara munta directoris ]
session     optional     pam_mount.so
```

Consultar el fitxer /etc/security/pam_mount.conf.xml
typus extrets del pam_mount.conf:

- **sshfs:**
  <volume fstype="fuse" path="sshfs#%(USER)@fileserver:" mountpoint="~" />
- **nfs:**
  <volume fstype="nfs" server="fileserver" path="/home/%(USER)" mountpoint="~" />
- **tmpfs:**
  <volume user="test" fstype="tmpfs" mountpoint="/home/test" options="size=10M,uid=%(USER),mode=0700" />
- **smbfs:**
  <volume user="user" fstype="smbfs" server="krueger" path="public" mountpoint="/home/user/krueger" />
- **Luks:**
  <volume path="/home/%(USER).img" mountpoint="~" cipher="aes-cbc-essiv:sha256" />

- pam_unix enregistra els logs
- pam_syslog
- pam_debug

8. Altres exemples PAM

Salts Condicionals

Els següents exemples mostren com establir salts el les regles PAM

authconfig:

- authconfig --test, authconfig --savebackup
- directori de configuracions desades: /var/lib/authconfig/….last…

Exemple regles PAM:

```
#auth          [default=1 success=ignore] pam_succeed_if.so debug uid = 1001
#auth          sufficient       pam_permit.so
#auth          sufficient       pam_deny.so
```

Descripció:
Si posem [default=1 success=ignore] significa que per defecte es salta la línia seguent i va a la que hi ha després de la seguent. default=1 vol dir saltan una. Ara be, si es produeix un success ignora aquesta regla i per tant va a la següent.
Així doncs si l'usuari és el 1001 ignora el salt i fa el permit. Si l'usuari no és el 1001 salta el permit i fa el deny.

Exemple:

```
auth    [success=1 default=ignore] pam_succeed_if.so debug uid = 1001
auth    sufficient       pam_permit.so
auth    sufficient       pam_deny.so
```

Descripció

Ara diu per defecte fes la línia següent (default=ignore), ara be, si es produeix un success salta una línia. Si l'usuari és el 1001 salta una línia i fa deny. Si l'usuari és un altre fa la línia següent qu eés un permit.

```
auth    [success=2 default=ignore] pam_succeed_if.so debug uid = 1001
auth    sufficient       pam_permit.so
auth    sufficient       pam_deny.so
auth    optional  pam_echo.so [ xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx ]
auth    sufficient pam_deny.so
```

Descripció
Indica que si l'usuari és el 1001 ha de saltar dues línies i fa el pam echo de les XXX i el denega. Si no és el 1001 fa el permit

exemple del manual de pam_mount.conf

```
          auth [success=2 default=ignore] pam_unix2.so
         auth [success=1 default=ignore] pam_ldap.so use_first_pass
         auth requisite pam_deny.so
```

```
auth optional pam_mount.so
```

Include Vs Substack

Crear un exemple amb la configuració pam de chfn que permeti examinar la diferència de funcionament de include i substack.

Recordeu que include provoca que els controls **die** i **done** finalitzin l'avaluació de tot l'stack (substack i stack principal). En canvi include continua l'avaluació de l'stack principal.
Nota combineu pam_echo.so al substack i a mòdul principal i observeu les accions segons s'utilitzi sufficient o requisite.

Implantació del servei nss-pam-ldap

Implementar nss-pam-ldapd per autenticar usuaris contra el servidor LDAP local.
Provar-ho també contra el servidor LDAP d'un company de classe.

Procediment bàsic:

- Instal·lar nss-pam-ldapd
- configurar l'acéss nslcd a la base de dades ldap local.
- configurar la resolució de noms amb nsswitch.conf.
- Engegar els serveis slapd, nscd i nslcd.
- Comprovar amb l'ordre getent passwd, getent group i getent hosts que la resolució de noms utilitza els files locals i el servidor ldap local.

Procediment autenticació PAM::

- Amb getent verfificar que es llisten i es consulten les dades d'usuaris i grups locals i de ldap.
- Practicar fent modificacions al PAM passwd per permetre modificar els password als usuaris locals i als usuaris ldap. Utilitzar pam_ldap.so.
- Modificar system-auth per tal d'establir l'autenticació permetent usuaris locals i usuaris LDAP.
- Tots els usuari locals i ldap han de poder fer login i modificar-se el password.
- Establir precedència entre local / ldap. Provar un cas de prova (Pere duplicat).

Procediment final:

- Ampliar la configuració de system-auth permetent que tinguem tres tipus d'usuaris, locals unix, ldap local i ldap+kerberos de l'escola. Cal utilitzar la configuració ldap i també la configuració sssd.

Altres:

- Posar filtres als usuaris
- [Ampliar la BD ldap amb més departaments.]
- [Usar una BD ldap per a hosts? Per a networks?]

```
# yum -y install  nss-pam-ldapd
Downloading Packages:
(1/2): nss-pam-ldapd-0.8.14-6.fc24.x86_64.rpm
(2/2): nscd-2.23.1-10.fc24.x86_64.rpm
...
```

Authconfig

Veure l'ordre authconfig:

- save/restore
- last config

- exemple d'instal·lació de les aules

- exemple usant pam_ldap
- exemple usant pam_kerberos
- exemple_ conjunt ladap+kerberos

Creació d'una aplicació PamAware

Crear una aplicació que autentiqui l'usuari utilitzant PAM:

- Fer un programa en Python que mostri els números del 0 al 10 però que per utilitzar-lo calgui l'autenticació de l'usuari.

Consultar la documentació de:

- Python-PAM " https://pypi.python.org/pypi/python-pam/"
- Descarregar i incorporar el mòdul per usar-lo en python.

Pràctica: Implementar en Python un programa que mostra els núemros naturals del 1 al 10 i que per poder-lo utilitzar cal validar l'usuari via PAM.

Instal·lar el software de python-pam:

```
# tar xvzf python-pam-1.8.2.tar.gz

# tree python-pam-1.8.2
python-pam-1.8.2
├── MANIFEST.in
├── pam.py
├── PKG-INFO
├── python_pam.egg-info
│   ├── dependency_links.txt
│   ├── PKG-INFO
│   ├── SOURCES.txt
│   └── top_level.txt
├── README.md
├── README.rst -> README.md
├── setup.cfg
└── setup.py
```

Escriure el codi del progarma pamaware.py (el nostre programa del 1 al 10):

```
#!/usr/bin/python
import pam
p=pam.pam()
userName=raw_input("Nom usuari: ")
userPasswd=raw_input("Passwd: ")
p.authenticate(userName, userPasswd)
print('{} {}'.format(p.code,p.reason))
if p.code == 0:
        for i in range(1,11):
        print i,
else:
        print "Error autenticacio"
```

Provar-ne l'execució:

```
# python pamaware.py
Nom usuari: marti
Passwd: marti
10 User not known to the underlying authentication module
Error autenticacio

# python pamaware.py
Nom usuari: pere
Passwd: kaka
7 Authentication failure
Error autenticacio

# python pamaware.py
Nom usuari: pere
Passwd: pere
0 Success
1 2 3 4 5 6 7 8 9 10
```

Creació d'un modul PAM

Consultar documentacio PAM de:

- Crear mòduls PAM implementant la API PAM en C o en Pythom.
- Desenvolupar aplicacions que utilitzin PAM com a esquema de seguretat

1. Usant PAM-Python crear un mòdul pam_pyquiz.so que permeti autenticar l'usuari si respon correctament una pregunta a l'atzar.

2. Fer un programa Python de mostrar la taula de multiplicar de N que només es pot usar si ets un usuari autenticat usant el mòdul de python pam_pyquiz.so.

Documentació:

- https://pypi.python.org/pypi/python-pam/1.8.1
- http://stackoverflow.com/questions/5286321/pam-authentication-in-python-without-root-privileges

PAM Python

- https://atlee.ca/software/pam/index.html

Python PAM

- http://pam-python.sourceforge.net/

Pràctica:
Crear un modul pam anomenat ***pam_mates.py*** que per validar un usuari li farà una pregunta de matemàtiques, si la respon be pot accedir al servei. Caldrà usar el mòdul pam_python.so que permet executar mòduls escrits en python.
S'aplicarà el mòdul al servei chfn de manera que l'usuari es podrà canviar el finger si sap matemàtiques.

Configuració PAM de chfn:

```
#%PAM-1
auth            optional        pam_echo.so [ auth ----------------- ]
auth            sufficient      pam_python.so /tmp/pam_mates.py

account         optional        pam_echo.so [ account -------------- ]
account         sufficient      pam_python.so /tmp/pam_permit.py

password        include         pam_deny.so
session         include         pam_deny.so
```

Programa pam_mates.py:

```
# pam_mates.py
# Validar l'usuari realitzant una pregunta de matemàtiques

def pam_sm_authenticate(pamh, flags, argv):
  print "Quant fan 3*2?"
  resposta=raw_input()
  if int(resposta) == 6:
          return pamh.PAM_SUCCESS
  else:
          return pamh.PAM_AUTHTOK_ERR

def pam_sm_setcred(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_acct_mgmt(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_open_session(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_close_session(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_chauthtok(pamh, flags, argv):
  return pamh.PAM_SUCCESS
```

El programa anterior s'ha fet a partir de copiar l'exemple *pam_permyt.py* de la documentació de pam-python. Es recomana mirar els tres exemples que mostra la documentació, que imiten en python els mateixos mòduls .so:

- pam_deny.py
- pam_permit.py
- pam_nologin.py

Exemple pam_permit.py

```
#
# Duplicates pam_permit.c
#
DEFAULT_USER    = "nobody"

def pam_sm_authenticate(pamh, flags, argv):
  try:
          user = pamh.get_user(None)
  except pamh.exception, e:
          return e.pam_result
  if user == None:
          pamh.user = DEFAULT_USER
  return pamh.PAM_SUCCESS

def pam_sm_setcred(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_acct_mgmt(pamh, flags, argv):
  return pamh.PAM_SUCCESS
```

```
def pam_sm_open_session(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_close_session(pamh, flags, argv):
  return pamh.PAM_SUCCESS

def pam_sm_chauthtok(pamh, flags, argv):
  return pamh.PAM_SUCCESS
```

Exemple pam_deny.py:

```
#
# Duplicates pam_deny.c
#
def pam_sm_authenticate(pamh, flags, argv):
  return pamh.PAM_AUTH_ERR

def pam_sm_setcred(pamh, flags, argv):
  return pamh.PAM_CRED_UNAVAIL

def pam_sm_acct_mgmt(pamh, flags, argv):
  return pamh.PAM_ACCT_EXPIRED

def pam_sm_chauthtok(pamh, flags, argv):
  return pamh.PAM_AUTHTOK_ERR

def pam_sm_open_session(pamh, flags, argv):
  return pamh.PAM_SYSTEM_ERR

def pam_sm_close_session(pamh, flags, argv):
  return pamh.PAM_SYSTEM_ERR
```

Exemle pam_nologin.py

```
#
# Emulate what pam_nologin.c does.
#
import pwd

#
# Parse our command line.
#
def parse_args(pamh, argv):
  #
  # Parse the arguments.
  #
  nologin_file = "/etc/nologin"
  retval_when_nofile = pamh.PAM_IGNORE
  for arg in argv[1:]:
        if arg.starts_with("file="):
        nologin_file = arg[5:]
        elif arg == "successok":
        retval_when_nofile = pamh.PAM_SUCCESS
  return nologin_file, retval_when_nofile

#
# Check the /etc/nologin file.
#
def check_nologin(pamh, nologin_file, retval_when_nofile):
  #
  # Get the user name.
  #
  try:
        username = pamh.get_user()
  except pamh.exception:
        username = None
  if username == None:
```

```
                return pamh.PAM_USER_UNKNOWN
    #
    # Can we open the file?
    #
    try:
                handle = file(nologin_file, "r")
    except EnvironmentError:
                return retval_when_nofile
    #
    # Print the message.
    #
    try:
                try:
                msg = handle.read()
                except EnvironmentError:
                return pamh.PAM_SYSTEM_ERR
    finally:
                handle.close()
    #
    # Read the user's password entry so we can check if he is root.
    # Root can login regardless.
    #
    try:
                pwent = pwd.getpwnam(username)
    except KeyError:
                retval = pamh.PAM_USER_UNKNOWN
                msg_style = pamh.PAM_ERROR_MSG
    else:
                if pwent[2] == 0:                       # Is this root?
                retval = pamh.PAM_SUCCESS
                msg_style = pamh.PAM_TEXT_INFO
                else:
                retval = pamh.PAM_AUTH_ERR
                msg_style = pamh.PAM_ERROR_MSG
    #
    # Display the message
    #
    try:
                pamh.conversation(pamh.Message(msg_style, msg))
    except pamh.exception:
                return pamh.PAM_SYSTEM_ERR
     return retval

#
# Entry points we handle.
#
def pam_sm_authenticate(pamh, flags, argv):
  nologin_file, retval_when_nofile = parse_args(pamh, argv)
  return check_nologin(pamh, nologin_file, retval_when_nofile)

def pam_sm_setcred(pamh, flags, argv):
  nologin_file, retval_when_nofile = parse_args(pamh, argv)
  return retval_when_nofile

def pam_sm_acct_mgmt(pamh, flags, argv):
  nologin_file, retval_when_nofile = parse_args(pamh, argv)
  return check_nologin(pamh, nologin_file, retval_when_nofile)
```

Compilar pam_python.so

Per obtenir l'executable Shared Object pam_python.si cal compilar el codi font en llenguatge que que proporciona el paquet tar i generar a cada màquina un element shared object pam_python.so. Per fer-ho a cada màquina caldrà instal·lar a més a més del paquet

pam-python aquelles dependències que necessiti. L'objectiu és acabar compilant i generant el mòdul que permet la interpretació de mòduls en python.

Instal·lar i generar l'executable de pam_python.so:

```
tar xvzf pam-python-1.0.6.tar.gz
dnf -y install sphinx python3-sphinx phyton2.sphinx
dnf -y install pam-devel
dnf -y install redhat-rpm-config
dnf -y install python-devel
## /usr/include/features.h:166:0:  --> he canviat 700 per 600
make
```

Si amb els passos anteriors hem obtingut el mòdul pam_python.so copiar-lo al directori on hi ha els mòduls de pam:

```
cp pam_python.so /usr/lib64/security/.
```

També cal que el mòduls python que fem i els d'exemple estiguin en una ubiacció coneguda. En el fitxer chfn hem indicat que estan a /tmp, de manera que els copiem allà:

```
cp pam_*.py /tmp
```

Execució

```
$ chfn anna
Changing finger information for anna.
Name []: 2
Office []: 2
Office Phone []: 2
Home Phone []: 2
 auth ----------------
Quant fan 3*2?
6
 account --------------
Finger information changed.
```

```
$ chfn anna
Changing finger information for anna.
Name [2]:
Office [2]:
Office Phone [2]:
Home Phone [2]: 2
 auth ----------------
Quant fan 3*2?
4
 account --------------
```

chfn: Permission denied
chfn: changing user attribute ***failed***: Permission denied

Pràctica global

Conceptes previs a practicar :

- Practicar el mòdul pam_unix.so.
- Practicar el mòdul pam_nologin.so.
- Practicar el mòdul pam_time.so.
- Practicar el mòdul pam_mkhomedir.so.
- Practicar el mòdul pam_mount.so:
    - Usar recursos tipus tmpfs.
    - Usar recursos de tipus nfs.
    - Usar shares de samba.
    - Usar recursos sshfs.

Primera pràctica:

- Usar un contenidor Docker amb un servidor LDAP amb els usuaris.
- Usar un contenidor Docker utilitzat com a recurs de disc, amb els homes dels usuaris exportats per ***nfs***.
- Usar un contenidor Docker *hostpam* que realitzi l'autenticació d'usuaris locals i de xarxa (via LDAP) i munti els homes dels usuaris via ***nfs***.

Segona pràctica:

- Usar un contenidor Docker amb un servidor LDAP amb els usuaris.
- Usar un contenidor Docker utilitzat com a recurs de disc, amb els homes dels usuaris exportats per ***samba***.
- Usar un contenidor Docker *hostpam* que realitzi l'autenticació d'usuaris locals i de xarxa (via LDAP) i munti els homes dels usuaris via ***samba***.

Tercera Pràctica:

- Ídem Primera Pràctica però amb els homes dels usuaris encriptats, cada home amb el seu propi password.

Primera Pràctica: NFS + LDAP + PAM

Implementar una topologia amb un contenidor ***ldapserver***, un contenidor ***nfsserver*** i un ***hostpam***. Aquest últim host conté usuaris locals i usuaris LDAP. Els usuaris LDAP munten automàticament el seu home via nfs.

Podeu trobar a docker hub els contenidors:

- edtasixm06/ldapserver:dataDB
- edtasixm06/nfsserver:pam
- edtasixm06/hostpam:nfs

ldapserver

Passos a fer:

- crear un contenidor ldapserver
- posar-lo en funcionament
- assegurar-se que és accessible exteriorment

```
docker run --name ldapserver --h ldapserver -it edtasixm06/ldapserver:dataDB /bin/bash

ldapserver$ /opt/docker/startup
ldapserver$ ldapserach -x
ldapserver$ ldapserach -x -b 'dc=edt,dc=org'

ldapserver -x -h 172.17.0.2 -b 'dc=edt,dc=org'
ldapserver -x -h ldapserver -b 'dc=edt,dc=org'
# verificar al /etc/hosts la ip apropiada del contenidor
```

nfsserver

Passos a seguir:

- crear un contenidor nfsserver
- verificar els recursos de nfs que s'exporten: editar /etc/exportfs per apropiar-ho a la configuració de xarxa.
- configurar l'arrancada dels serveis de xarxa relacionats amb nfs:
- engegar el servidor
    - Si intentem imitar manualment l'engegada del servei com que nfs-utils.service estira de nfs.sockert, cal fer manualment el socket
      # mkdir /run/rpcbind
      # touch /run/rpcbind/rpcbind.lock
- comprovar localment i externament l'accés als recursos exportats per nfs.

```
docker run --name nfsserver -h nfsserver --privileged=true -it isxvladimir/server-nfs
```

```
# Crear l'estructura de directoris dels homes dels usuaris
# atenció al ouwner
nfsserver$ tree /mnt/data

# Configurar l'exportació de tot el directori
nfsserver$ cat /etc/exports
/mnt/data 192.168.1.42(rw,sync,no_subtree_check,no_root_squash,fsid=0)
/mnt/data 172.17.0.*(rw,sync,no_subtree_check,no_root_squash,fsid=0)
/mnt/data 172.17.0.3(rw,sync,no_subtree_check,no_root_squash,fsid=0)
/mnt/data 127.0.0.1(rw,sync,no_subtree_check,no_root_squash,fsid=0)
/mnt/data *(rw,sync,no_subtree_check,no_root_squash,fsid=0)

# Instal·lar nfs-utils i configurar-ne l'arrancada
nfsserver$ dnf -y instal nfs-utils

# Truc per generar el socket
nfsserver$ mkdir /run/rpcbind
nfsserver$ touch /run/rpcbind/rpcbind.lock

# Script d'arrancada
nfsserver$ cat /opt/docker/starttup.sh
 rpcbind
 rpc.statd
 rpc.nfsd
 exportfs -a
 rpc.mountd &disown

nfsserver$ bash startup.sh

# /Verificar que es pot fer el mount local i remotament
nfsserver$ mount -t nfs localhost:/var/tmp/nfs/ /mnt/
nfsserver$  mount -t nfs
localhost:/var/tmp/nfs/ on /mnt type nfs
(rw,relatime,vers=3,rsize=524288,wsize=524288,namlen=255,hard,proto=tcp6,timeo=600,
retrans=2,sec=sys,mountaddr=::1,mountvers=3,mountport=20048,mountproto=udp6,local
_lock=none,addr=::1)
```

hostpam

Passos a fer:

- crear un container host
- instal·lar nfs-utils i pam_mount
- engegar el servei nfs la part client.
- configurar l'autenticació via ldap amb authconfig-tui
- configurar PAM system-auth per fer el mkhomedir
- configurar PAM system-auth per fer pam_mount
- configurar pam_mount.conf.xml per incloure el muntage automatic del home dels alumnes LDAP via NFS

```
 docker run --name hostpam -h hostpam --privileged=True -it fedora:24 /bin/bash

hostpam$ dnf -y install nfs-utils pam_mount authconfig passwd nss-pam-ldap

# Truc per configurar el socket de nfs-utils
hostpam$ mkdir /run/rpcbind
hostpam$ touch /run/rpcbind/rpcbind.lock

# Script arrancada servei nfs (parcial)
hostpam$ cat /opt/docker/startup.sh
  rpcbind
  rpc.statd

# verificar que podem muntar
mount -t nfs  172.17.0.4:/var/tmp/nfs /mnt
mount -t nfs  nfsserver:/var/tmp/nfs /mnt
umount /mnt

# Crear usuaris locals (local01, local02 i local03)
useradd local01
passwd local01

# Configurar amb authconfig la utilització de LDAP
authconfig-tui
        activar LDAP:
        Use LDAP
        Use LDAP authentication
        server ldap://172.17.0.2    #server ldap://ldapserver
        baseDN dc=edt,dc=org

# Script (final) d'arrancada del client nfs, nslcd i nscd
hostpam$ cat /opt/docker/startup.sh
  rpcbind
  rpc.statd
  /usr/sbin/nslcd
  /usr/sbin/nscd

/opt/docker/startup

# Verificar amb getent que identifica els noms
# nota: assegureu-vos que el /etc/hosts identifica els servidors ldapserver i nfsserver
getent passwd
getent passwd anna

# Editar PAM /etc/pam.d/system-auth
  Incloure pam_mkhomedir.so
  Incloure pam_mount.so

# Incloure el muntatge del recurs NFS dels homes dels usuaris
```

```
vim /etc/secutity/pam_mount.conf.xml
   <volume fstype="nfs" server="nfsserver" path="/var/tmp/nfs/%(GROUP)/%(USER)"
mountpoint="~" />

# Verificar que un usuari LDAP accedeix al seu home de xarxa
$ su - anna
Password:
-sh-4.3$ pwd
/tmp/home/anna
-sh-4.3$ mount
…
nfsserver:/var/tmp/nfs/alumnes/anna on /tmp/home/anna type nfs
(rw,relatime,vers=3,rsize=524288,wsize=524288,namlen=255,hard,proto=tcp,timeo=600,r
etrans=2,sec=sys,mountaddr=172.17.0.3,mountvers=3,mountport=20048,mountproto=udp
,local_lock=none,addr=172.17.0.3)
```

Segona Pràctica: SAMBA + LDAP + PAM

Implementar una topologia amb un contenidor ***ldapserver***, un contenidor ***samba*** i un ***hostpam***. Aquest últim host conté usuaris locals i usuaris LDAP. Els usuaris LDAP munten automàticament el seu home via nfs.

Podeu trobar al repositori Docker Hub els contenidors:

- edtasixm06/ldapserver:dataDB
- edtasixm06/samba:pam
- edtasixm06/hostpam:samba

ldapserver

La creació i configuració del ldapserver és la mateixa que la descrita en l'apartat anterior. S'utilitzarà el container l*dapserver:dataDB*.

sambaserver

Passos a fer:

- crear un contenidor samba amb shares, que publiqui el home dels suaris.
- crear usuaris (crear els usuaris de la base de dades LDAp dc=edt,dc=org).
- activar el servei SAMBA.
- verificar que localment i remotament es poden muntar els shares dels homes dels usuaris.

```
docker run -m samba -h samba -it edtasixm06/samba:base /bin/bash

# Activar el servidor samba
/opt/docker/startup.sh

#Verificar exportació de shares
smbclient -L //172.17.0.3
smbclient //samba/manpages
```

hostpam

Passos a fer:

- Crear el container per a fer la funció de hostpam.
- Instal·lar el software de client samba.
- Definir apropiadament al /etc/hosts el nom del *sambaserver* i del *ldapserver*.
- Amb authconfig configurar la utilització de LDAP.
- Editar system-auth per incorporar pam_mkhomesir.so i pam_mount.so.

- Ediatr pam_mount.conf.xml per definir el muntatge dels shares dels usuaris de LDAP.

```
docker run --name hostpam -h hostpam --privileged=True -it fedora:24 /bin/bash

# Instal·lar el software samba client
dnf -y install cifs-utils samba samba-client authconfig pam_mount nss-pam-ldapd

# Verificar que els shares del samba server es poden muntar
smbclient //172.17.0.3
smbclient -L //172.17.0.3
mount -vt cifs -o user=anna //172.17.0.3/anna /mnt
Password for anna@//172.17.0.3/anna:  *******
mount.cifs kernel mount options: ip=172.17.0.3,unc=\\172.17.0.3\anna,user=anna,pass=********
# mount -t cifs
devpts on /dev/console type devpts (rw,nosuid,noexec,relatime,seclabel,gid=5,mode=620,ptmxmode=000)
//172.17.0.3/anna on /mnt type cifs
(rw,relatime,vers=1.0,cache=strict,username=anna,domain=SMB,uid=0,noforceuid,gid=0,noforcegid,addr=172.17.0.3,unix,p
osixpaths,serverino,mapposix,acl,rsize=1048576,wsize=65536,echo_interval=60,actimeo=1,user=anna)
# umount /mnt

# Configurar la utilització de ldap, mkhomedir i pam_mount
authconfig-tui

# Verificar que s'ha engegat nslcd i nscd amb startup.sh
# Verificar l'accés als usuaris de lDAP amb getent
# Crear els usuaris locals

# Configurar el mòdul pam_mount per usar samba
/etc/security/pam_mount.conf.xml
<volume user="anna" fstype="cifs" server="172.17.0.5" path="manpages" mountpoint="~"
/>
 <volume user="*" fstype="cifs" server="172.17.0.5" path="%(USER)" mountpoint="~" />

Verificar l'accés al home de l'usuari:
su - anna
Password: anna
Creating directory '/tmp/home/anna'.
reenter password for pam_mount: smbanna
mount -t cifs
//172.17.0.3/anna on /tmp/home/anna type cifs
(rw,relatime,vers=1.0,cache=strict,username=anna,domain=SMB,uid=5002,forceuid,gid=600,forcegid,addr=172.17.0.3,unix,
posixpaths,serverino,mapposix,acl,rsize=1048576,wsize=65536,echo_interval=60,actimeo=1)
```

Tercera Pràctica: LUKS + LDAP + PAM

Implementar una topologia amb un contenidor ***ldapserver***, un contenidor amb els homes dels usuaris encriptats amb ***LUKS*** i un ***hostpam***. Aquest últim host conté usuaris locals i usuaris LDAP. Els usuaris LDAP munten automàticament el seu home via nfs usant LUKS.

Annex

Configuracions exemple per a pam_mount.so

Configuració nfs

```
# cat /etc/exports
/usr/share/doc *(rw,sync)
/opt *(rw,sync)
/var/tmp *(rw,sync)

# exportfs -av
exporting *:/var/tmp
exporting *:/opt
exporting *:/usr/share/doc

# exportfs -v
/usr/share/doc    <world>(rw,sync,wdelay,hide,no_subtree_check,sec=sys,secure,root_squash,no_all_squash)
/opt              <world>(rw,sync,wdelay,hide,no_subtree_check,sec=sys,secure,root_squash,no_all_squash)
/var/tmp          <world>(rw,sync,wdelay,hide,no_subtree_check,sec=sys,secure,root_squash,no_all_squash)
```

```
# mount -t nfs a31:/var/tmp /mnt

# mount -t nfs
a31:/var/tmp on /mnt type nfs
(rw,relatime,vers=3,rsize=524288,wsize=524288,namlen=255,hard,proto=tcp,timeo=600,retrans=2,sec=sys,mountaddr=192
.168.4.31,mountvers=3,mountport=20048,mountproto=udp,local_lock=none,addr=192.168.4.31)

# umount /mnt
```

```
 dnf -y install nfs-utils
Last metadata expiration check: 2:51:30 ago on Tue 13 Nov 2018 09:46:29 AM CET.
Package nfs-utils-1:2.2.1-4.rc2.fc27.x86_64 is already installed, skipping.
Dependencies resolved.
Nothing to do.
Complete!
```

```
# systemctl  status nfs
● nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled; vendor preset:
disabled)
  Drop-In: /run/systemd/generator/nfs-server.service.d
           └─order-with-mounts.conf
   Active: active (exited) since Tue 2018-11-13 11:38:23 CET; 58min ago
```

```
 Main PID: 7181 (code=exited, status=0/SUCCESS)
         Tasks: 0 (limit: 4915)
   CGroup: /system.slice/nfs-server.service

Nov 13 11:38:23 a31 systemd[1]: Starting NFS server and services...
Nov 13 11:38:23 a31 systemd[1]: Started NFS server and services.

# systemctl status nfs-server
● nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled; vendor preset:
disabled)
  Drop-In: /run/systemd/generator/nfs-server.service.d
           └─order-with-mounts.conf
   Active: active (exited) since Tue 2018-11-13 11:38:23 CET; 57min ago
 Main PID: 7181 (code=exited, status=0/SUCCESS)
         Tasks: 0 (limit: 4915)
   CGroup: /system.slice/nfs-server.service

Nov 13 11:38:23 a31 systemd[1]: Starting NFS server and services...
Nov 13 11:38:23 a31 systemd[1]: Started NFS server and services.
```

Configuració sshfs

```
# dnf -y install fuse-sshfs
Last metadata expiration check: 2:52:20 ago on Tue 13 Nov 2018 09:46:29 AM CET.
Package fuse-sshfs-2.10-1.fc27.x86_64 is already installed, skipping.
Dependencies resolved.
Nothing to do.
Complete!
```

```
# sshfs root@a31:/tmp /mnt
```

Configuració SAMBA

```
# vim /etc/samba/smb.conf
….
[public]
        comment = Directori public
        path = /var/tmp
        guest ok = yes
        browseable = yes
[documents]
        comment = docs
        path = /usr/share/docs
        guest ok = yes
        browseable = yes
```

```
[manuals]
        comment = manuals
        path = /usr/share/man
        guest ok  = yes
        browseable = yes
```

```
# systemctl start nmb smb
[root@a31 ~]# systemctl status nmb smb
● nmb.service - Samba NMB Daemon
  Loaded: loaded (/usr/lib/systemd/system/nmb.service; disabled; vendor preset: disabled)
  Active: active (running) since Tue 2018-11-13 12:09:18 CET; 33min ago
 Main PID: 11653 (nmbd)
  Status: "nmbd: ready to serve connections..."
        Tasks: 1 (limit: 4915)
  CGroup: /system.slice/nmb.service
        └─11653 /usr/sbin/nmbd --foreground --no-process-group

● smb.service - Samba SMB Daemon
  Loaded: loaded (/usr/lib/systemd/system/smb.service; disabled; vendor preset: disabled)
  Active: active (running) since Tue 2018-11-13 12:09:18 CET; 33min ago
 Main PID: 11656 (smbd)
  Status: "smbd: ready to serve connections..."
        Tasks: 5 (limit: 4915)
  CGroup: /system.slice/smb.service
        ├─11656 /usr/sbin/smbd --foreground --no-process-group
        ├─11658 /usr/sbin/smbd --foreground --no-process-group
        ├─11659 /usr/sbin/smbd --foreground --no-process-group
        ├─11660 /usr/sbin/smbd --foreground --no-process-group
        └─11678 /usr/sbin/smbd --foreground --no-process-group
```

```
# mount -vt cifs -o guest //a31/public /mnt
mount.cifs kernel mount options: ip=192.168.4.31,unc=\\a31\public,user=,pass=********

# mount -t cifs
//a31/public on /mnt type cifs
(rw,relatime,vers=default,sec=none,cache=strict,domain=,uid=0,noforceuid,gid=0,noforcegid,addr=192.168.4.31,file_mode=
0755,dir_mode=0755,soft,nounix,serverino,mapposix,rsize=1048576,wsize=1048576,echo_interval=60,actimeo=1)
```