

TD3

PAM

Organisation du travail

Pour cette manipulation, vous travaillez seul. PAM gère toutes les authentifications de votre système ... **soyez prudent**. Voici quelques précautions élémentaires à prendre avant de se mettre au travail :

- ↪ copier ¹ le répertoire pam.d vers pam.d.original
- ↪ vérifier que l'on sait rebooter en *single user*
- ↪ réfléchir avant d'agir

Vous rédigerez au fur et à mesure de votre avancement un document reprenant les diverses manipulations que vous faites ainsi que vos conclusions.

En cas de problèmes

Voici l'extrait de réponse de Peter WOOD sur <http://kernel.org>

```
> What the hell do I do now?
OK, don't panic. The first thing you
have to realize is that this happens to
50% of users who ever do anything with
PAM. It happened here, not once, not
twice, but three times, all different,
and in the end, the solution was the same
every time.

First, I hope you installed LILO with
a delay. If you can, reboot, hit shift or
tab or something and type :
LILO boot : linux single
(Replace 'linux' with 'nameofyournormal-
linuximage'). This will let you in without
logging in. Ever wondered how easy it is
to break into a linux machine from the
console? Now you know.
```

If you can't do that, then get yourself
a bootkernel floppy and a root disk a-la
slackware's rescue.gz. (Red Hat's ins-
tallation disks can be used in this mode
too.)

In either case, the point is to get
back your root prompt.

Second, I'm going to assume that

you haven't completely nuked your pam
installation - just your configuration
files. Here's how you make your configs
nice again :

```
cd /etc
mv pam.conf pam.conf.orig
mv pam.d pam.d.orig
mkdir pam.d
cd pam.d
```

and then use vi to create a file called
"other" in this directory. It should
contain the following four lines :

```
auth      required      pam_unix.so
account  required      pam_unix.so
password required      pam_unix.so
session  required      pam_unix.so
```

Now you have the simplest possible
PAM configuration that will work the
way you're used to. Everything should
magically start to work again. Try it
out by hitting ALT-F2 and logging in on
another virtual console. If it doesn't
work, you have bigger problems, or you've
mistyped something. One of the wonders of

¹J'ai bien dit copier, pas déplacer .. sinon vous êtes déjà foutu !

this system (seriously, perhaps) is that if you mistype anything in the conf files, you usually get no error reporting of any kind on the console - just some entries in the log file. So look there! (Try 'tail

/var/log/messages'.)

From here you can go back and get a real configuration going, hopefully after you've tested it first on a machine you don't care about screwing up. :/>

Lectures

- ↪ *The System Administrators' Guide*, [\[lien\]](#)
- ↪ *The Application developers' Manual*, [\[lien\]](#)

1 Manipulations

1.1 Manipulation 'administrateur'

Commande su

Modifier le fichier `/etc/pam.d/su` afin que l'utilisateur `root` doive entrer son mot de passe lors de l'utilisation de la commande.

Remodifier le fichier pour qu'aucun utilisateur ne doive entrer de mot de passe lors de l'utilisation de `su`.

Tester également

- ↪ les restrictions d'accès "temporelle", pas de `su` entre 10h15' et 10h30',
- ↪ les restrictions de lieu (pas de `su` d'un terminal, uniquement d'une console),
- ↪ ... (l'un ou l'autre paramètre au choix)

1.2 Manipulation 'developpeur'

1.2.1 Exemple

Tester le programme proposé dans *The Application developpers' Manual*, le comprendre, l'utiliser et, éventuellement, l'améliorer.

```
$ cat check_user.c
/*
 * This program was contributed by Shane Watts
 * [modifications by AGM]
 * You need to add the following (or equivalent) to the /etc/pam.conf file.
 * # check authorization
 * check_user  auth      required  /usr/lib/security/pam_unix_auth.so
 * check_user  account   required  /usr/lib/security/pam_unix_acct.so
 */

#include <security/pam_appl.h>
#include <security/pam_misc.h>
#include <stdio.h>
```

```

static struct pam_conv conv = {
    misc_conv,
    NULL
};

int main(int argc, char *argv[])
{
    pam_handle_t *pamh=NULL;
    int retval;
    const char *user="nobody";

    if (argc == 2) {
        user = argv[1];
    }

    if (argc > 2) {
        fprintf(stderr, "Usage: _check_user_ [username]\n");
        exit(1);
    }

    retval = pam_start("check_user", user, &conv, &pamh);

    if (retval == PAM_SUCCESS)
        retval = pam_authenticate(pamh, 0); /* is user really user? */

    if (retval == PAM_SUCCESS)
        retval = pam_acct_mgmt(pamh, 0);
    /* permitted access? */

    /* This is where we have been authorized or not. */

    if (retval == PAM_SUCCESS) {
        fprintf(stdout, "Authenticated\n");
    } else {
        fprintf(stdout, "Not_Authenticated\n");
    }

    if (pam_end(pamh,retval) != PAM_SUCCESS) {
        /* close Linux-PAM */
        pamh = NULL;
        fprintf(stderr, "check_user:_failed_to_release_authenticator\n");
        exit(1);
    }

    return ( retval == PAM_SUCCESS ? 0:1 );
    /* indicate success */
}

```

Pour rappel, vous devez lier les bibliothèques PAM lors de la compilation ... ce qui vous donne une commande du style

```
gcc check_user.c -o check_user -lpam -lpam_misc -ldl
```

1.2.2 Programme ls avec authentification

Écrire un *wrapper* pour la commande ls qui demande une authentification.

Pour rappel, le langage C permet de lancer une commande système en substituant le processus en cours. Voir pour ce faire la commande `execvp`.