

Extending and Abusing Pluggable Authentication Modules (PAM)

Bar Camp San Diego 3
D.J. Capelis
{student,researcher,staff} UCSD

Quick Question

What's your level of experience with PAM?

What's PAM?

I know what PAM is... but I don't think I use it

I know I use PAM

I edited my PAM configuration yesterday

I've written a PAM module or application

I'm Andrew Morgan*

(* Andrew Morgan is the developer of Linux-PAM)

Outline

What is PAM?

How do I use PAM?

You already are

Configuring PAM

Writing new authentication modules for PAM

Abuse #1: pam_webform

Abuse #2: pam_sshkey

PAM in the future

Moving beyond passwords

I hear graphical interfaces are hot

Web 2 point goo + PAM

(We'll see how far we get in the time we have)

Authentication

Fundamental Premise:

Teach the computer to tell users apart

Usual Mechanisms:

Passwords

Security Keys

etc...

Basic PAM Info

Almost every Unix machines uses it

Open Standard (Standardized by Sun in 1995)

Easily allows applications to authenticate against other systems

Allows easy configuration of different types of authentication

All In The Name

Modules which Authenticate and are Pluggable!
(More commonly: Pluggable Authentication Modules)

It's all in the name.

Who Uses PAM?

Platform

Linux

FreeBSD

NetBSD

AIX (> 5.2)

HP-UX

Solaris

OS X

PAM implementation:

Linux-PAM

OpenPAM

OpenPAM

???

???

SolarisPAM

Linux-PAM (Embraced and
ext'd.)

You Do

A Few Modules

Module

pam_unix

pam_ldap

pam_smb

pam_krb

pam_permit

pam_deny

pam_motd

pam_mkhomedir

pam_rootok

pam_wheel

Description

"Logging In"

LDAP Integration

Windows Networks

Kerberos

Permit All

Deny All

Display /etc/motd

Make homedirs

No passwds for root

Restrict to users in wheel

Configuring PAM

```
$ cat /etc/pam.d/su
auth          sufficient    pam_rootok.so
#Uncommenting this makes users in group wheel not need a
password
#auth        sufficient    pam_wheel.so use_uid trust
auth         required      pam_wheel.so use_uid
auth         required      pam_env.so
auth         sufficient    pam_unix.so try_first_pass likeauth
nullok
auth         required      pam_deny.so
```

Huh?

Realms:

Auth

Account

Password

Session

Controls:

Required

Requisite

Sufficient

Optional

(The next module will always be checked for required and optional controls)

Who is this user?

Can this user?

Changing passwords

Set the user up

Needed for user to auth

Immediate stop if not

Stop if success

Doesn't matter

More Configurations

Quick... someone tell me what this one does:

```
$ cat /etc/pam.d/useradd  
auth          sufficient    pam_rootok.so
```

(This is a very common use of pam.)

One More

This one is kind of neat:

```
$ cat /etc/pam.d/eject
auth      sufficient  pam_rootok.so
auth      required    pam_console.so
```

What would we do if we wanted to allow everyone access?

Using PAM

Six very simple steps:

From the application perspective:

Load libpam

Tell it you want to authenticate a user

It loads modules

Magic happens

It tells you whether or not the user authenticated and who they are

You do your thing or exit()

Moving On...

Okay... so now we're past the introduction about what PAM is and what it does. Please take some time to let things sink in and fully understand the system.

Moving On...

Okay, let's go.

Moving On...

Let's talk abuse.

Abuse?

By abuse... I just really wanted to share ideas for cool modules I'd like to write but haven't had the time.