

LinuxForum 2007

Pluggable Authentication Modules

Authentication in the UNIX world

Kenneth Geisshirt
<http://kenneth.geisshirt.dk/>

Agenda

- The auth problem
- Introduction to PAM
- Examples
- Testing PAM configuration
- Where to find more information

The auth problem

- Classic UNIX user databases are `/etc/passwd` and `/etc/group`
 - Services like IMAP, POP3, and FTP might have a set of users
 - Adding web applications increase the number of auth databases
- ⇒ PAM is the solution

Introduction to PAM

- PAM is a framework for authentication
- Implemented by many operating systems
 - Solaris (2.6 and later)
 - GNU/Linux (most distributions)
 - FreeBSD and NetBSD
 - Mac OS X
 - AIX (5.1 as add-on, native in 5.2 and later)
 - HP-UX v11 (add-on)

Introduction to PAM

- **Benefits for system administrations**
 - reuse user databases
 - change mechanisms without recompilation or reboot
- **Benefits for application developers**
 - use stable API for authentication
 - leave configuration to system administrators

Directories and files

- `/etc/pam.conf` – original main configuration file
- `/etc/pam.d/` - newer type of configuration
- `/lib/security/` – PAM modules (so-files)
- `/etc/security/` - extra configuration of modules

Key concepts

- **service name**
 - login, gdm, ...
- **management group**
 - auth, session, password, account
- **control flags**
 - requisite, required, sufficient, optional
- **modules**
 - pam_unix, pam_mount, pam_winbind, ...

Services

- Applications register as services
- Often hard coded in the source code
- Configuration file is
`/etc/pam.d/service`

```
int main(int argc, char *argv[]) {  
  
    pam_handle_t *pamh = NULL;  /** PAM data structure **/  
  
    char *user = getlogin();  
  
    /** Creating and initializing a PAM session **/  
  
    retval = pam_start("vault", user, &conv, &pamh);  
  
}
```

Stacking

- Modules can be stacked
- They are “called” by order in configuration file
- Control flags are used for flow control

```
auth          required    pam_unix.so nullok_secure
```

```
auth          optional    pam_mount.so use_first_pass
```

Management groups

- auth is for validating users and assignment of group membership
- session creates and destroys working environment
- account controls access to services
- password controls how passwords are changed

Control flags

- Control flags is a kind of flow construction
 - requisite; failure \Rightarrow terminated immediately and return not-OK
 - required; failure \Rightarrow continue but will return not-OK
 - sufficient; success \Rightarrow terminate immediately and return OK
 - optional; has not influence on flow and return code

Common options

- `debug`; write log messages using `syslog` (often `/var/log/auth`)
- `try_first_pass`; reuse the first password, prompt for new if incorrect
- `use_first_pass`; reuse the first password, does not prompt

Example 1

- Encrypted home directory
 - laptops often hold sensitive data
 - requires an empty partition
- `# apt-get install libpam-mount cryptsetup openssl`
- `# cryptsetup -c aes -h ripemd160 -s 256 -y create kneth /dev/sda2`
- Use your password as encryption key

Example 1

- Add the following line to

```
/etc/security/pam_mount.conf:
```

```
volume kneth crypt - /dev/hda6 /home/kneth  
cipher=aes - -
```

- The `/etc/pam.d/{login,gdm}` is:

```
auth          required    pam_unix.so nullok_secure  
auth          optional    pam_mount.so use_first_pass  
session       required    pam_unix.so  
session       optional    pam_mount.so use_first_pass
```

Example 2

- Remote authentication using Microsoft Active Directory
- Samba's winbind is one solution

Example 2

- **Modify** `/etc/samba/smb.conf`
- **Modify** `/etc/krb5.conf`
- **Modify** `/etc/nsswitch.conf`
- **Join the directory/domain:**

```
kinit Administrator
```

Example 2

- **Finally, PAM:**

```
auth sufficient pam_winbind.so
```

```
auth required pam_unix2.so use_first_pass
```

```
session sufficient pam_mkhome.so skel=/etc/skel/umask=0022
```

```
session required pam_unix2.so
```

Testing

- Always use a test computer
- or use a virtual computer
- Leave a back door open
- Log in and never log out

Test examples

- Construct test examples as for ordinary software:
 - Input and expected output
 - Run all test after any change
 - Test all combinations in stacks

Example

```
auth          required    pam_unix.so nullok_secure
```

```
auth          optional    pam_mount.so use_first_pass
```

- Valid account, correct password, encrypted home directory
- Valid account, correct password, noencrypted home directory
- Valid account, incorrect password, encrypted home directory
- Valid account, incorrect password, noencrypted home directory
- Invalid account

Automatic tests

```
#!/usr/bin/expect -f

send_user "Valid user, valid
password\n"
spawn pamtester httpd kneth
authenticate
expect "assword: "
send "Only2day\r"
expect
set timeout 60
```

More information

- *Pluggable Authentication Modules: The Definitive Guide to PAM for Linux SysAdmins and C Developers.* Kenneth Geisshirt. Packt Publishing, January 2007.
- Linux-PAM:
- Solaris PAM:
<http://www.sun.com/software/solaris/pam>
- Pamtester:
<http://pamtester.sourceforge.net/>