Pluggable Authentication Modules (PAM)

Theodore Ts'o
tytso@mit.edu
http://web.mit.edu/tytso/www/home.html

The Problem

- Many different authentication systems
  - Kerberos
  - S/Key
  - Shadow passwords
- System administrators want to customize /bin/login
  - Authorization controls
  - Set up home directories

The Problem II

- Need to customize "system entry services"
  - /bin/login
  - ftpd
  - /bin/passwd
- Often hard to customize OS binaries

Solution

- Pluggable Authentication Modules
- Developed by Sun
- Adopted by the OSF
  - DCE RFC 86.0
- Adopted by the Common Desktop Environment (CDE)

Design Goals of PAM

- Allow the system administrator to choose the default authentication mechanism
- Configure mechanisms on a per-application basis
- Support multiple mechanisms for each application
- Support display requirements (X vs. tty) of applications

Design Goals of PAM II

- Avoid requiring the user to retype passwords
- However, allow multiple passwords for higher security
- System-entry services should not have to change when mechanisms change

Design Goals of PAM III

- Provide a "pluggable" model for authentication, account, password, and session management.
- Should support the requirements of current system entry services

Issues not addressed by PAM

- Focus mainly on mechanisms where user use passwords to authenticate themselves to the system
  - How user authentication can be communicated to other parties is not addressed by PAM
    - Kerberos
    - GSSAPI
- Source of information of getpwnam(), getgroups(), etc. not addressed by PAM

PAM architecture

- PAM applications
  - Access PAM functionality using the PAM API
- The PAM library
  - Provides the PAM API
  - Interfaces to PAM mechanisms
- PAM Mechanisms
  - Do the actual authentication work
  - Implemented as dynamically loaded libraries

The /etc/pam.conf file

- Read by the PAM library
- Specifies which PAM mechanisms should be used for each PAM application
- Syntax

```
application module_type control_flag module_path       options
login       auth        required     pam_unix_auth.so  nowarn
login       session     required     pam_unix_session.so
ftp         auth        required     pam_skey_auth.so  debug
```


PAM services

- Authentication (auth)
  - Is the password OK? Get tickets, etc.
- Account Management (account)
  - Is the user allowed to log in?
- Session Management (session)
  - Home directory setup
- Password Management (password)
  - Allow password changes

Stacking multiple PAM mechanisms

- Run modules in order given by config file
- Control flags
  - required
    - If the mechanism fails, return an error after executing all the mechanism on the stack
  - sufficient
    - If the mechanism succeeds, return success
  - optional
    - Ignore mechanism failures

PAM Implementation efforts

- SunSoft implementation
  - Licensed to vendors as part of CDE
- The Linux PAM effort

Linux PAM

- Instigated by me in January, 1996
- Nearly all of the work done by others
  - Andrew Morgan
  - Michael Johnson
- Mailing list: pam-list@redhat.com
- Linux PAM is shipping as part of the RedHat 4.0 distribution

It's not just for Linux anymore!

- Fundamentally, Linux-PAM is just a free PAM implementation
- Uses nothing inherently Linux specific
- Code under either GPL or BSD-style copyright (your choice!)
- Other OS development efforts are encouraged to pick it up.

For more information

- See the Linux PAM web page
  - http://parc.power.net/morgan/Linux-PAM/index.html
  - Materials available there
    - Linux PAM distribution and documentation
    - OSF/DCE RFC 86.0
    - PAM man pages generously provided by SunSoft

Conclusion

- An idea whose time had come long ago
- Kudos to the programmers at Sun who created the PAM specification
- The power of the free software development community

January 9, 1996 Pluggable Authentication Modules 17