

PAM

Pluggable Authentication Modules

Markus Korzendorfer Hagen Paul Pfeifer

Software Anwendungs Architektur
Computer-Networking — Fachbereich Informatik
Fachhochschule Furtwangen

<http://pam.0xdef.net>
pam@0xdef.net

15. Dezember 2004

Agenda

- 1 Einführung
 - Grundsätzlich
 - PAM Architektur
- 2 Anwendung
 - Konfiguration
 - PAM Module
 - PAM Applicationen
 - Anwendungsbeispiele
- 3 Implementierung (PAM-Unplugged)
 - Anwendungsentwickler
 - Modulentwickler
 - pam_blue

Was ist PAM

- Pluggable Authentication Modules
- Flexibler Mechanismus für Authentification, Session Management, ...

PAM Ziele (nach RFC 86.0)

- 1 Systemadministrator soll Authentisierungsmechanismus bestimmen
- 2 Interaktion mit Anwendung (Darstellung telnet \Leftrightarrow xdm)
- 3 Konfigurierbar pro Anwendung
- 4 verschiedene Authentisierungsmechanismem stackbar
- 5 mehrere Passwörter möglich
- 6 Passwort-, Benutzerkonten-, und Sitzungsmanagment Model
- 7 Abwärtskompatibel
- 8 Benutzertransparenz

Historisch

- Vergleich Passwort mit /etc/passwd
- Benutzer **ist** Benutzer wenn Passwort korrekt

Application

UNIX **LDAP**

Probleme

- Bei neuen Authentication Schema folgt ein rewrite der Application
- Keine strikte Trennung Authentifikation von Applicationscode

PAM

- PAM trennt Applicationscode von Authenticationscode durch Schnittstelle
- Administrator bestimmt Authentifikationsmechanismus
- Sammlung von Modulen
- flexibel durch „Modulstacking“
- spezifiziert in OSF-RFC 86.0
- unterstützung durch AIX, FreeBSD, HP/UX, GNU/Linux und Solaris

PAM Schichten Modell

ssh

login

apache

gdm

...

pluggable authentication modules - interface

ldap

unix

kerberos

secureid

...

Konfiguration - Übersicht

- /etc/pam.d - Konfigurationsdateien der Applicationen
- /etc/pam.conf - Konfigurationsdateien der Applicationen (historisch)
- /etc/security - Konfigurationsdateien der Module
- /lib/security - Modulverzeichnis (Bibliotheksmodule)

Authentifizierungsmechanismus (visuell)

Typische PAM Konfiguration

Modultyp	Kontrollflag	Modulpfad	Argumente
auth	required	/lib/security/pam_unix.so	debug
auth	sufficient	/lib/security/pam_ldap.so	use_first_pass
session	required	/lib/security/pam_unix.so	debug
password	required	/lib/security/pam_cracklib.so	minlen=10
password	required	/lib/security/pam_unix.so	md5 shadow

Modultyp

Spezifiziert welche Managementfunktion erfüllt werden soll

- ① auth
 - Benutzeridentifizierung und -authentifizierung (z.B. Passwortabfrage oder Smartcards)
- ② account
 - Verwaltung des Accounts („Gibt es dieses Benutzer im System und darf er sich anmelden?“)
- ③ password
 - Steuerung der Passwortänderung („Dieses Passwort ist zu kurz!“)
- ④ session
 - Verwaltung der Sitzung (Limits, Berechtigungen, ... während des Zugriffes)

Modulsteuerung (Kontrollflag)

Spezifiziert das Verhalten in Anhängigkeit des Rückgabewertes

- ① required
 - Modul muss zwingend durchlaufen werden
- ② requist
 - bei Fehler wird sofort zum Anwendungsprogramm zurückgekehrt
- ③ sufficient
 - bei Erfolg des Modul ist dies für eine positive Gesamtmeldung ausreichend
- ④ optional
 - bei Erfolg oder Misserfolg werden trotzdem alle nachfolgende Module abgearbeitet

Modulpfad und Argumente

- Modulpfad
 - ist ein Verweis auf das zu benutzende Modul
- Argumente
 - debug
 - liefert Diagnosemeldungen an Logging Daemon
 - use_first_pass
 - versucht Passwort von vorhergehenden Modul zu übernehmen
 - try_first_pass
 - fordert im Fehlerfall den User auf, sein Passwort erneut einzugeben

PAM Module (kleiner Auszug)

- 1 pam_unix
 - bildet historischen Authentifizierungsmechanismus nach (/etc/passwd)
- 2 cracklib
 - prüft Passwort auf Schwachstellen
- 3 ldap
 - vergleicht Passwort gegen LDAP Verzeichnissdienst
- 4 time
 - setzt zeitgesteuerte Zugangskontrollen
- 5 limits
 - teilt Systemressourcen pro Benutzer zu (CPU, Speicher, ...)

PAM enabled applications™

- apache
- login
- samba
- ftp
- imapd
- ssh
- ...

Anwendungsbeispiel Nr. 1

Beispiel vHost (remote login via ssh (/etc/pam.d/sshd))

auth	required	/lib/security/pam_securetty.so	
auth	sufficient	/lib/security/pam_ldap.so	debug
auth	required	/lib/security/pam_unix.so	try_first_pass
account	sufficient	/lib/security/pam_ldap.so	
account	required	/lib/security/pam_unix.so	
password	sufficient	/lib/security/pam_ldap.so	use_auth_tok
password	required	/lib/security/pam_unix.so	use use_first_pass md5 shadow
session	required	/lib/security/pam_unix.so	

Anwendungsbeispiel Nr. 2

Beispiel für sicherheitskritischen Bereich (login
(/etc/pam.d/login))

auth	required	/lib/security/pam_securetty.so	
auth	required	/lib/security/pam_secureid.so	
auth	required	/lib/security/pam_unix.so	
account	required	/lib/security/pam_unix.so	
password	required	/lib/security/pam_cracklib.so	minlen=12 retry=3
password	required	/lib/security/pam_unix.so	use use_first_pass md5 shadow
session	required	/lib/security/pam_time.so	

Anwendungsentwickler

Modulentwickler

- Module sind dynamische Programmbibliotheken
- liefern definierte Schnittstelle für PAM

pam_blue

Quellen

- ▶ Manual Pages
man {ptrace,gdb,gcc,readelf,nm}
- ▶ <http://www.kernel.org/pub/linux/libs/pam>
Documentation, Module,
- 📖 Richard M. Stallman and Roland H. Pesch
Using GDB: A Guide to the GNU Source-Level Debugger
- 📖 John Gillmore
GDB Internals