Wykorzystanie systemu PAM w GNU/Linuksie

Grzegorz Jacek Nalepa

20.3.2000, Kraków, *Revision* : 1.4

Streszczenie

Artykuł opisuje system *PAM*, czyli *Pluggable Authentication Modules*. PAM jest elastycznym systemem autentykacji użytkowników umożliwiającym tworzenie aplikacji całkowicie niezależnych od metody autentykacji. Przedstawiona jest architektura systemu i jego konfigurowanie. PAM używa zestawu „modułów autentykacyjnych", które umożliwiają implementację dowolnych metod autentykacji i używanie ich z dowolnymi aplikacjami. Artykuł opisuje najpopularniejsze moduły, sposób ich konfigurowania i wykorzystywania. System PAM jest bardzo ważnym rozszerzeniem mechanizmów bezpieczeństwa w systemie GNU/Linux. Jego uniwersalność pozwala na opracowanie specjalnych metod autentykacji dla istniejących usług w sieciach komputerowych, a otwarta architektura umożliwia jego dowolną rozbudowę i wsparcie dla nowych technologi autentykacji.

Spis treści

[1]Tekst ukazał się w: *Magazynie NetForum*, nr 5/2000, wydawanym przez Lupus.

[2]Kontakt z autorem: `mail:gjn@agh.edu.pl`

[3]Tytuł angielski: *Using PAM in GNU/Linux system*

1. Wstęp

Jedną z najważniejszych cech współczesnych systemów operacyjnych jest bezpieczeństwo. Systemy Unix były zawsze znane ze swojego zaawansowanego modelu bezpieczeństwa. Jednym z podstawowych elementów tego modelu jest identyfikacja użytkowników i weryfikacja ich tożsamości (autentykacja). Tradycyjne mechanizmy identyfikacji użytkowników, jakkolwiek bardzo elastyczne, po upływie 20 lat wykazują pewne ograniczenia. System GNU/Linux ma praktycznie identyczne mechanizmy identyfikacji użytkowników jak klasyczne systemy Unix. W związku z tym dziedziczy zarówno zalety jak i ograniczenia tych mechanizmów. Aby uświadomić sobie na czym one polegają, warto krótko omówić klasyczny mechanizm identyfikacji użytkowników w systemie GNU/Linux.

2. Tradycyjny model autentykacji w Uniksie

Aby użytkownik uzyskał dostęp do systemu, musi zostać zidentyfikowany, a jego tożsamość zweryfikowana. W chwili gdy system ustalił tożsamość użytkownika daje mu dostęp do odpowiednich zasobów, najczęściej otwiera dla niego sesję interaktywną, lub daje dostęp do określonej usługi. Identyfikacja użytkownika opiera się na podaniu przez niego nazwy konta użytkownika, a tożsamość jest potwierdzana na podstawie podanego hasła. System ma bazę danych o wszystkich kontach użytkowników i ich hasłach w pliku `/etc/passwd`. Plik zawiera nazwę i identyfikator konta, parametry sesji, dane osobowe użytkownika, oraz zaszyfrowane hasło. Wprowadzona przez użytkownika podczas logowania nazwa konta jest porównywana z nazwami kont w pliku `passwd`. Wprowadzone hasło jest szyfrowane, a wynik porównywany z zaszyfrowanym hasłem z tego pliku. Dla poprawienia bezpieczeństwa system został z czasem rozszerzony o dodatkowy plik, `/etc/shadow`, w którym są przechowywane zaszyfrowane hasła (zamiast w `/etc/passwd`). Podczas gdy plik `passwd` może czytać każdy, plik `shadow` może czytać tylko użytkownik root.

2.1. Zalety i wady

Zaletą tego systemu jest jego prostota i przejrzystość. Do niedawna w bardzo wielu przypadkach jego funkcjonalność była wystarczająca. Ograniczenia systemu zaczęły się ujawniać wraz ze zwiększeniem się ilości usług oferowanych przez systemy operacyjne, szczególnie usług sieciowych. Przykładowo, takie usługi sieciowe jak Rsh, Ssh czy Ftp mają własne mechanizmy identyfikacji użytkowników, które rozszerzają, lub nawet zastępują standardowy mechanizm. Te programy wymagają dodatkowych mechanizmów dostępu i dodatkowych informacji, których nie daje metoda oparta na pliku `/etc/passwd`. Jakiekolwiek rozszerzenia podstawowego systemu musiałyby za sobą pociągnąć modyfikacje wszystkich programów, które z niego korzystają, co byłoby niezwykle trudne, a przeważnie po prostu niemożliwe. Dlatego Rsh używa pliku `.rhosts`, Ssh metody wymiany kluczy publicznych RSA (między innymi), a Ftp pliku `/etc/ftpusers`. W związku z tym mogą pojawić się niespójności. Przykładowo, zablokowanie dostępu dla użytkownika w pliku `/etc/passwd` nie zablokuje może nie zablokować dostępu przez Rsh, czy SSh.

Ponieważ architektura Uniksa jest elastyczna, wielu producentów komercyjnych Uniksów starało się udoskonalić system. W systemie GNU/Linux wprowadzono dość rewolucyjną zmianę, system PAM, opracowany przez firmę SunSoft. Jest on w stanie w pełni zastąpić tradycyjne metody autentykacji użytkowników.

3. Czym jest PAM

Skrót PAM pochodzi od *Pluggable Authentication Modules*. PAM jest elastycznym systemem autentykacji użytkowników umożliwiając tworzenie aplikacji całkowicie niezależnych od metody autentykacji. Równocześnie nie narzuca żadnych metod oraz nie ma ograniczeń na ich tworzenie. PAM używa zestawu „modułów autentykacyjnych", które umożliwiają implementację dowolnych metod autentykacji i używanie ich z dowolnymi aplikacjami.

System został rozwinięty w firmie SunSoft i przystosowany do pracy w systemie Solaris. Jego standard jest opisany przez V. Samara i R. Schemersa w dokumencie *OSF RFC 86.0, "Unified Login with Pluggable Authentication Modules (PAM)"*.

Za implementację systemu dla platformy GNU/Linux jest odpowiedzialny Andrew G. Morgan (`<morgan@ftp.kernel.org>`). Ta implementacja nosi nazwę Linux-PAM i jej aktualną wersją (w marcu 2000) jest wersja Linux-PAM-0.71. Ma ona szereg rozszerzeń względem oryginalnej implementacji SunSoft, takie jak modularna konfiguracja i możliwość hierarchicznej autentykacji.

Integracja PAM w dystrybucji systemu GNU/Linux jest złożona i wymaga rekompilacji wielu programów. W związku z tym nie wszystkie dystrybucje używają PAM. PAM jest wykorzystywany w Caldera Open Linux, Debian/GNU 2.2, RedHat Linux od wersji 3.0.4, SuSE Linux 6.2 oraz w systemie FreeBSD 3.2 (dane przytoczone za Andrew G. Morganem).

4. Architektura PAM

PAM uniezależnia rozwijanie aplikacji od metod autentykacji. Jest to możliwe dzięki zastosowaniu osobnej biblioteki, z której korzystają aplikacje wymagające metod autentykacji użytkowników. W celu zapewnienia tych metod biblioteka wykorzystuje zewnętrzne uniwersalne moduły. W dystrybucjach GNU/Linux biblioteka `libpam` znajduje się w katalogu `/lib`, a moduły, noszące nazwy zaczynające się od `pam_` znajdują się w katalogu `/lib/security`.

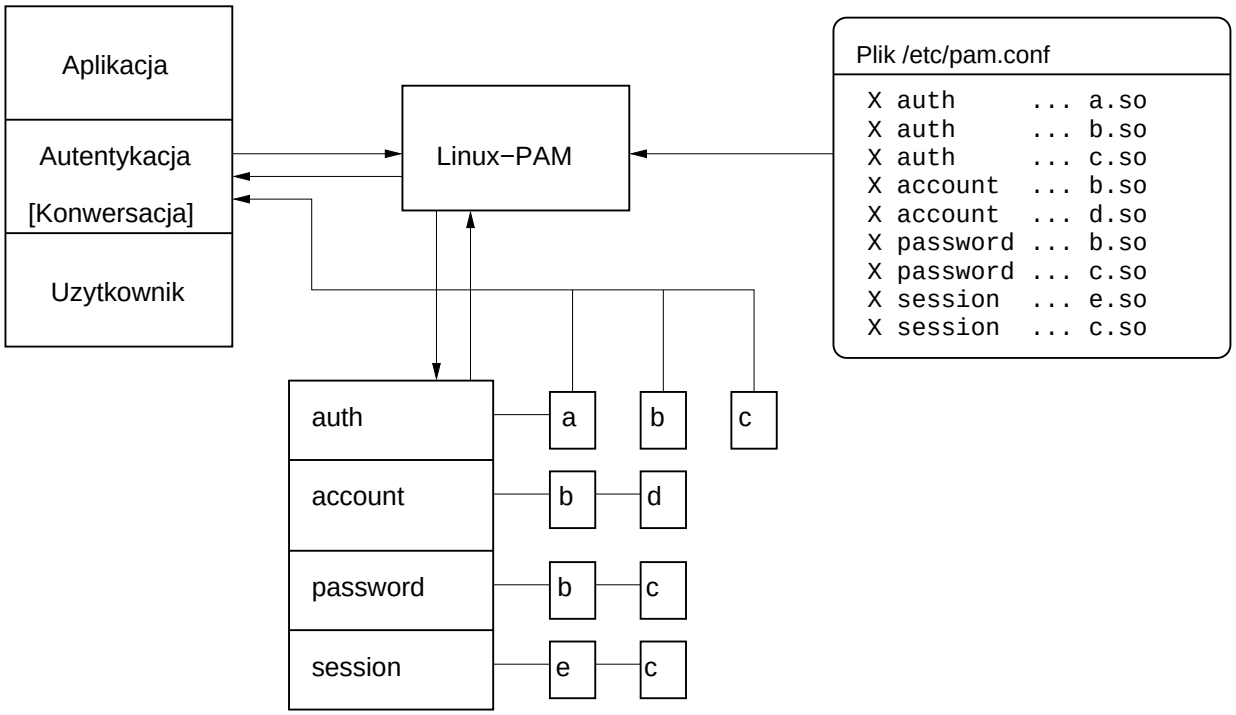
Architektura systemu PAM jest przedstawiona na Rysunku 1. Aplikacja jest wyposażona w interfejs do biblioteki Linux-PAM i jest niezależna od metody autentykacji. Aby dokonać autentykacji użytkownika, aplikacja komunikuje się z Linux-PAM. System PAM określa na podstawie konfiguracji, odpowiedniej dla danej aplikacji sposób autentykacji i określa, które moduły i w jaki sposób będą realizowały autentykację. Aplikacja może opcjonalnie udostępniać PAM własny moduł realizujący konwersację z modułami PAM. Po ustaleniu uprawnień PAM przesyła odpowiednie informacje o uprawnieniach użytkownika do aplikacji.

System PAM ma cztery podstawowe grupy modułów, zarządzające czterema grupami zadań:

- autentykacją (ang. *authentication management*) – moduł `auth`. Ten moduł umożliwia identyfikację użytkownika na podstawie hasła lub innych metod. Oprócz tego ma możliwość przypisania użytkownika do odpowiedniej grupy.

- kontami (ang. *account management*) – moduł `account`. Przez zarządzanie kontami rozumie się zarządzanie niezależne od autentykacji użytkowników. Moduł ustala uprawnienia na podstawie dostępnych zasobów systemu, pory dnia, czy lokalizacji użytkownika.

- sesją (ang. *session management*) – moduł `session`. Zarządzanie sesją obejmuje działania, jakie system podejmuje przed otwarciem czy zamknięciem sesji użytkownika. Działania te mogą obejmować rejestrowanie informacji o użytkowniku, modyfikacje środowiska lub montowanie odpowiednich systemów plików.



Rysunek 1: Architektura PAM

- hasłami (ang. *password management*) – moduł `password`. Ten moduł jest odpowiedzialny za uaktualnianie baz danych o użytkownikach, zawierających ich hasła, dane osobowe, czy listy uprawnień.

Dzięki systemowi PAM to administrator ma możliwość pełnego wyboru mechanizmów autentykacji dla całego systemu i każdej aplikacji z osobna. Przez odpowiednią konfigurację można całkowicie wyłączyć autentykację (przy pomocy modułu `pam_permit`). Z drugiej strony można wymusić wieloetapową autentykację na podstawie hasła jednorazowego (`pam_pwgen`), systemu LDAP (`pam_ldap`), linii papilarnych (`pam_biomouseplus`), kart dostępu (`SecureID`, `pam_cryptocard`), czy skaningu siatkówki.

5. Konfigurowanie systemu

Konfiguracja PAM znajduje się w pliku `/etc/pam.conf` lub w plikach w katalogu `/etc/pam.d`. Ta druga wersja jest nowsza i wygodniejsza. Składnia plików konfiguracyjnych jest praktycznie w obydwóch przypadkach identyczna.

5.1. Plik `pam.conf`

Plik konfiguracyjny `/etc/pam.conf` składa się z lini o następującej składni:

```
service-name   module-type   control-flag   module-path   arguments
```

W przypadku plików w katalogu `/etc/pam.d` różnica jest tylko taka, że każdy plik odpowiada jednej usłudze. Nazwa pliku jest taka sama jak pole `service-name`, a linie w pliku zawierają pozostałe cztery pola. Każda linia odpowiada pojedynczemu modułowi PAM dla konkretnej usługi. Stosowanie plików konfiguracyjnych w katalogu `/etc/pam.d` jest rozwiązaniem bardziej uniwersalnym i zalecanym.

Znaczenie pól jest następujące:

1. `service-name` – oznacza typ usługi wymagającej metody autoryzacji, na przykład `login`, `su`, `ftpd`,
2. `module-type` – jeden z opisanych powyżej czterech typów modułów: `auth`, `account`, `session`, `password`,
3. `control-flag` – to pole definiuje reakcję systemu PAM na rezultat pracy modułu.

   Pole ma następującą składnię:

   ```
   [wartość1=działanie1 wartość2=działanie2 ...]
   ```

   Pole `wartość` oznacza jeden z około 30 różnych wartości zwracanych przez moduł, dotyczących przebiegu autentykacji, na przykład: `user_unknown`, `acct_expired`, `try_again`. Pole `działanie` może przyjmować jedną z 6 wartości: `ignore`, `ok`, `done`, `bad`, `die`, `reset`. Oznaczają one działanie, jakie biblioteka ma podjąć po otrzymaniu kodu o podanej `wartości`. Na przykład `ok` oznacza, że kod jest do zaakceptowania i autentykacja ma się zakończyć sukcesem, a pole `bad` oznacza sytuację odwrotną.

   Istnieje jeszcze uproszczona, starsza składnia definiowania `control-flag`. Polega ona na podaniu jednego z 4 słów kluczowych: `required`, `requisite`, `sufficient` i `optional`. Wskazują one na rolę, jaką ma odgrywać moduł. Na przykład słowo `required` oznacza, że pomyślna autentykacja przy pomocy tego modułu jest konieczna dla całego procesu autentykacji.
4. `module-path` – pełna ścieżka dostępu do danego modułu, najczęściej `/lib/security/pam_nazwamodułu.so`.
5. `arguments` – opcjonalne argumenty dla modułu, przekazywane mu przez linię poleceń, wpływające na jego pracę. Argumenty są różne dla konkretnych modułów. Oprócz tego jest grupa argumentów standardowych. Wszystkie moduły powinny przyjmować ogólne argumenty takie jak: `debug`, `no_warn`, `use_first_pass`, `try_first_pass`, `expose_account`.

Stosowanie plików konfiguracyjnych w katalogu `/etc/pam.d` ma różne zalety, takie jak łatwiejsza rekonfiguracja, możliwość używania linków symbolicznych dla tych samych metod autentykacji, oraz ułatwione zarządzanie pakietami (każdy pakiet DEB, lub RPM może dodawać własny plik z metodą autentykacji).

5.2. Narzędzia i moduły

O sile systemu PAM stanowi uniwersalne API umożliwiające rozwijanie różnych modułów autentykacji. W standardowej dystrybucji Linux-PAM jest ponad 20 różnych modułów. Oprócz tego dostępnych jest ponad 40 modułów dodatkowych dostępnych jako osobne pakiety. Listę tych wszystkich modułów, wraz z opisem można znaleźć na stronach projektu Linux-PAM. W Tabeli 1 zebrano informacje o kilku wybranych modułach.

Moduł `pam_unix` realizuje wszystkie funkcje związane z tradycyjnym modelem autentykacji w Uniksie, opartym na pliku `/etc/passwd`. Dzięki niemu można skonfigurować system używający PAM tak, aby naśladował działanie systemu haseł Uniksa.

Moduły dodatkowe umożliwiają wprowadzenie rozproszonego systemu autentykacji wykorzystującego sieć komputerową. Dostępne są moduły umożliwiające połączenia z siecią opartą na serwisach LDAP, NDS czy Samba.

Nazwa	Funkcja
Moduły standardowe	
`pam_cracklib`	eliminuje słabe hasła
`pam_deny`	realizuje odmowę dostępu
`pam_ftp`	obsługuje anonimowe ftp
`pam_group`	przydziela do grup użytkowników
`pam_limits`	ogranicza dostęp na podstawie dostępnych zasobów
`pam_listfile`	reguluje dostęp na podstawie list użytkowników
`pam_tally`	realizuje odmowę dostępu w zależności od liczby nieudanych logowań
`pam_unix`	realizuje standardowe uniksowe zarządzanie autentykacją
`pam_warn`	moduł logujący informacje do Syslog
Moduły dodatkowe	
`mod_auth_pam`	moduł autentykujący dla serwera Apache, są również inne spełniające podobne funkcje
`pam_smb`	grupa modułów współpracujących z serwerem Samba
`pam_mysql`	autentykacja na podstawie bazy danych MySQL
`pam_ldap`	współpraca z serwisem LDAP
`pam_proftpd`	autentykacja dla serwera ProFtpd
`pam_nw_auth`	współpraca z siecią Netware, również NDS

Tablica 1: Wybrane moduły PAM

6. Przykłady zastosowań

Liczba dostępnych modułów, a co za tym idzie liczba możliwych sposobów ich konfiguracji jest tak duża, że trudno w sposób wyczerpujący omówić ich praktyczne konfigurowanie. Można co najwyżej podać kilka konkretnych przykładów konfiguracji.

Po pierwsze, konfiguracja PAM powinna zawierać plik `/etc/pam.d/other` określający domyślne zachowanie systemu dla usług nie mających osobnych metod autentykacji.

Plik `/etc/pam.d/other` może wyglądać następująco:

```
auth     required       /lib/security/pam_warn.so
auth     required       /lib/security/pam_unix.so
account  required       /lib/security/pam_warn.so
account  required       /lib/security/pam_unix.so
password required       /lib/security/pam_warn.so
password required       /lib/security/pam_unix.so
session  required       /lib/security/pam_warn.so
session  required       /lib/security/pam_unix.so
```

Przy takiej konfiguracji usługi nie mające własnych metod autentykacji będą obsługiwane w standardowy sposób, na podstawie pliku `/etc/passwd`. Dodatkowo, moduł `pam_warn` będzie rejestrował dostęp do usługi przez Syslog.

Jeżeli w powyższym przykładzie każde wystąpienie modułu `pam_unix.so`, zastąpi się przez `pam_deny.so` to dostęp do usług, które nie mają zdefiniowanych osobno metod autentykacji będzie zabroniony.

Przykład ilustruje ważną cechę PAM, a mianowicie hierarchiczne łączenie modułów. Dla każdej usługi (w tym przypadku `other`) i każdego typu modułu (na przykład `auth`) można podać kilka modułów, jeden po drugim (ang. *stacking*). W takim przypadku są one wykonywane kolejno i w zależności od wyniku działania konkretnego modułu możliwe jest przerywanie

sekwencji. Przy pomocy modułu `pam_if` istnieje nawet możliwość warunkowego wykonywania sekwencji modułów.

Przykład rozbudowanej konfiguracji PAM dla programu login może wyglądać tak:

```
auth     requisite      /lib/security/pam_unix.so       nullok
auth     required       /lib/security/pam_securetty.so
auth     required       /lib/security/pam_nologin.so
auth     required       /lib/security/pam_env.so
auth     required       /lib/security/pam_mail.so
account  required       /lib/security/pam_unix.so
password required       /lib/security/pam_unix.so
session  required       /lib/security/pam_unix.so
session  required       /lib/security/pam_limits.so
```

W trakcie autentykacji jest sprawdzany terminal z którego użytkownik się loguje, istnienie blokady logowania, ustawiane jest środowisko pracy i wyświetlana jest informacja o poczcie elektronicznej. Parametry sesji są regulowane w zależności od dostępnych zasobów.

Inny, prosty przykład, wykorzystujący listę użytkowników uprawnionych do logowania jest następujący:

```
auth     required       /lib/security/pam_listfile.so \
 onerr=fail item=user sense=allow file=/etc/pam-login_users
auth     required       /lib/security/pam_unix.so
account  required       /lib/security/pam_unix.so
password required       /lib/security/pam_unix.so
session  required       /lib/security/pam_unix.so
```

Przykład konfiguracji usługi Ftp z uwzględnieniem dostępu anonimowego:

```
ftpd    auth    sufficient  /usr/lib/security/pam_ftp.so
ftpd    auth    required    /usr/lib/security/pam_unix_auth.so use_first_pass
ftpd    auth    required    /usr/lib/security/pam_listfile.so \
                        onerr=succeed item=user sense=deny file=/etc/ftpusers
```

Pierwszy z modułów realizuje dostęp przez anonimowe Ftp, drugi typowy dostęp dla użytkowników, a trzeci sprawdza, czy użytkownik nie ma zabronionego dostępu do usługi w pliku `/etc/ftpusers`.

W przypadku użycia opcji `debug` lub modułu `pam_warn.so` system PAM rejestruje zdarzenia przy pomocy Syslog. Te komunikaty mogą wyglądać następująco:

```
login: pam_unix session started for user root, service login
xdm: pam_unix session finished for user gjn, service xdm
```

7. Podsumowanie

System PAM jest bardzo ważnym rozszerzeniem mechanizmów bezpieczeństwa w systemie GNU/Linux. Jego uniwersalność pozwala na opracowanie specjalnych metod autentykacji dla istniejących usług w sieciach komputerowych, a otwarta architektura umożliwia jego dowolną rozbudowę i wsparcie dla nowych technologi autentykacji.

Wyczerpujące omówienie systemu nie jest możliwe w tak krótkim artykule. Pełne informacje na temat projektu Linux-PAM można znaleźć na stronie
http://www.kernel.org/pub/linux/libs/pam. Znajduje się tam również lista dostępnych modułów wraz z adresami, pod którymi są one dostępne. Dzięki nim wdrażanie systemu Linux-PAM może być prostsze i prowadzić do poprawienia bezpieczeństwa systemu.