Użytkownicy, grupy i prawa dostępu

Paweł Józiak

Wydział Matematyki i Informatyki Politechniki Warszawskiej

15 XI 2021

Plan zajęć

Plan zajęć

Baza użytkowników lokalnych

- W większości systemów baza tożsamości użytkowników jest rozłożona na poniższe trzy pliki:
  - `/etc/passwd` – zawiera podstawowe informacje o użytkownikach,
  - `/etc/shadow` – zawiera zaszyfrowane hasła oraz metainformacje o ograniczeniach użytkowników (w niektórych systemach BSD odpowiednikiem tego pliku jest `/etc/master.passwd`),
  - `/etc/group` – specyfikuje grupy w systemie.
- Każdy z nich jest plikiem tekstowym ASCII, w którym każda linia oznacza nowy rekord, a pola są rozdzielone dwukropkiem.

Informacje o użytkownikach – */etc/passwd*

Format wpisu do */etc/passwd*

```
username:password:uid:gid:gecos:directory:shell
```

- Kolejne pola zawierają następujące informacje:
  - `username` – nazwa użytkownika,
  - `password` – symbol zastępczy hasła, utrzymywany dla wstecznej kompatybilności (formatu pliku `/etc/passwd`), zwykle ‘`x`’,
  - `uid` – unikalny identyfikator numeryczny użytkownika,
  - `gid` – unikalny identyfikator numeryczny podstawowej grupy użytkownika,
  - `gecos` – pełna nazwa użytkownika,
  - `directory` – ścieżka do katalogu domowego użytkownika,
  - `shell` – ścieżka do Powłoki używanej przez użytkownika – program wywoływany po poprawnym zalogowaniu do konta użytkownika (dla Linuxów to zwykle `/bin/bash`).

Przykładowy wpis do */etc/passwd*

```
pawel:x:1000:1000:Pawel Joziak:/home/pawel:/bin/bash
```

Nazwa użytkownika a UID

- Wpisy do `/etc/passwd`, `/etc/shadow` oraz `/etc/group` są powiązane ze sobą za pomocą nazw (*username*).
- Podczas logowania, użytkownicy są proszeni o podanie nazwy *username*.
- Dla każdego innego celu, np. przechowywania informacji o właścicielu pliku, procesu czy zarządzania zasobami używa się UID. Niemniej, wiekszość programów parsujących te dane zagląda do adekwatnych baz tożsamości i wyświetla nazwy, nie UID użytkowników.
- Warto zauważyć, że UID przechowywane w pamięci komputera nie wymagają istnienia odpowiadających wpisów w bazie tożsamości. Wówczas jedynie wartość numeryczna jest wyświetlana.
- Dwóch użytkowników z jednakowym UID jest nierozróżnialnych dla systemu.

Typy kont użytkowników

- Można wyróżnić trzy typy kont użytkowników:
  - `root` (UID=0, GID=0) – superużytkownik z nieograniczonymi uprawnieniami,
  - zwykłych użytkowników,
  - *konta systemowe*.
- Każdy proces odpalony jest w imieniu jakiegoś użytkownika, i z jego uprawnieniami. Jest zatem wybitnie niestosownym uruchamiać serwisy jak: serwer internetowy, serwer SQL, serwer poczty e-mail, serwer drukarki itd, które wymagają jedynie ograniczonych uprawnien, z uprawnieniami roota. Dla takich serwisów tworzone są osobne konta systemowe. Tworzone są dla nich sztuczne loginy powłók oraz niewłaściwe hasła, by nie było możliwości zalogowania się do tych użytkowników.
- Bezpieczną praktyką jest pozostawienie numerów UID aż do 1000 na poczet kont systemowych, podobnie jak numerów GID aż do 100 dla predefiniowanych grup dodawanych automatycznie przy instalacji oprogramowania.

Powłoki – */etc/shells*

- `/etc/shells` zawiera informacje o wszystkich poprawnych Powłokach.
- Użytkownicy, którzy mają swoje domyślne powłoki (ustawione w `/etc/passwd`) nieuwzględnione w tym pliku, nie mogą się zalogować (wystartować sesji TTY).
- Zwykle, Powłoki są automatycznie dodawane do tego pliku przy instalacji.
- Dla powłók, które mają symlinki bądź twarde linki, wszystkie ścieżki powinny tu być podane.

Konta użytkowników – przykłady

Przykład */etc/passwd*

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
adm:x:3:4:adm:/var/adm:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/bin/false
news:x:9:13:news:/usr/lib/news:/bin/false
uucp:x:10:14:uucp:/var/spool/uucppublic:/bin/false
operator:x:11:0:operator:/root:/bin/bash
postmaster:x:14:12:postmaster:/var/spool/mail:/bin/false
nobody:x:65534:65534:nobody:/:/bin/false
smithj:x:1001:100:John Smith:/home/smithj:/bin/bash
```

Zaszyfrowane hasła – */etc/passwd*

- Wszyscy użytkownicy mają możliwość czytania plików `/etc/passwd` oraz `/etc/group` .
- Ze względów bezpieczeństwa, zaszyfrowane hasła zostały przesunięte do osobnego pliku, który nie jest czytalny przez zwykłych użytkowników. W większości systemów Uniks plikiem tym jest `/etc/shadow` .
- Pierwsze trzy pola zawierają: nazwę użytkownika, zaszyfrowane hasło oraz datę ostatniej zmiany hasła (0 oznacza hasło przeterminowane, konieczność zmiany przy najbliższym zalogowaniu). Kolejne pola zawierają informacje o terminowości hasła i są zwykle puste.
- Wszystkie daty są reprezentowane jako liczba dni od początku epoki Uniksa, tj. od 1 stycznia 1970.

Hasła w systemach Unix

- Istnieje szereg różnych algorytmów pozwalających na zaszyfrowanie hasła (zob. `man 3 crypt`)
- Szyfrowanie jest jednokierunkowe (zaszyfrowane hasła nie mogą być odszyfrowane).
- Różni użytkownicy o tożsamych hasłach (jako stringi) mają zwykle różne postacie haseł zaszyfrowanych.
- Poprawne zaszyfrowane hasło nie będzie zaczytać się od niektórych znaków (np. `*` albo `!`); ich użycie jest metodą na tymczasowe zablokowanie dotępu do konta.
- Publicznie odczytywalne postacie zaszyfrowanych haseł pozwalałyby na zastosowanie metody *brute force attack*.
- Można włączyć alerty dla zbyt wielu prób nieskutecznego zalogowania na dane konto z ustalonego TTY bądź adresu IP (zob. `https://www.fail2ban.org/`).

Zmiana hasła

- `root` nie zna haseł żadnego użytkownika, lecz może je zmienić.
- Zwykli użytkownicy mogą zmienić swoje własne hasło.
- Komendy związane ze zmianą haseł to m.in:
  - `passwd [username]` – interaktywna zmiana hasła dla `username`. Domyślnie: dla bieżącego użytkownika.
  - `pwgen` – generator losowych, bezpiecznych haseł o wysokiej entropii.
  - `chpasswd` – narzędzie podobne do `passwd`, ale wygodniejsze w skryptowaniu (m.in. obsługa podejścia batchowego)

Grupy – */etc/group*

- Każdy użytkownik jest przypisany do jednej grupy podstawowej (GID w `/etc/passwd`).
- Użytkownicy mogą należeć do innych grup: uprawnienia i wyłączenia na daną grupę dotyczą wszystkich jej użytkowników.
- Definicje grup oraz informacje o ich członkach są przechowywane w `/etc/group` .

Format wpisu do */etc/group*

```
nazwa_grupy:haslo:gid:uzytkownicy
```

- gdzie pola tłumaczą się następująco:
  - `nazwa_grupy` – ludzka nazwa grupy, np. *uzytkownicy*,
  - `haslo` – symbol zastępczy hasła (‘`x`’), utrzymywany dla wstecznej kompatybilności,
  - `gid` – GID grupy, unikalny identyfikator numeryczny,
  - `uzytkownicy` – lista rozdzielonych przecinkiem nazw użytkowników, którzy są członkami tej grupy.

Grupy w systemach Unix

- Uprawnienia pliku mogą być ustawione dla wszystkich członków grupy.
- Dodatkowe uprawnienia do zarządzania zasobami sprzętowymi mogą być przydzielone członkom poszczególnych grup (np. `audio`, `camera`, `disk`, `optical`, `power`, `storage`, `tty`, `video`, etc)
- UWAGA! Współczesne systemy Uniksowe zorientowane na zastosowania domowe (w szczególności: Linuksy) mogą przyznawać rozszerzone uprawnienia do zasobów sprzętowych dla wszystkich użytkowników zalogowanych w trybie graficznym (za pomocą tzw. *consolekits*, *policykits* etc).
- Niektóre działania mogą być ograniczone jedynie do członków grupy `wheel`.
- Członkowie grupy `wireshark` mogą ustawić karty sieciowe w tzw. trybie ogólnym (analizy surowych klatek).
- Używanie niektórych funkcjonalności i programów może wymagać bycia członkiem w określonych grupach.

Rozszerzona autentyfikacja – PAM

- Autentyfikacja w Uniksie jest przeprowadzana przez wykonanie sekwencji kroków autentyfikacyjnych – tzw. modułów PAM (*Pluggable Authentication Modules*).
- Autentyfikacja względem wpisów do `/etc/passwd` oraz `/etc/shadow` to tylko jeden z możliwych modułów PAM.
- Inne moduły mogą określać inne źródła tożsamości użytkowników (bazy danych, LDAP etc), ponad te domyślne.
- Moduły PAM są skonstruowane wyłącznie w celu przeprowadzenia zadań autentyfikacyjnych.
- Właścicielstwo pliku/procesu jest przechowywane w formie UID oraz GID.

Zarządzanie tożsamościami – NSS

- NSS (*Name Service Switch*) pozwala na wykonywanie kwerend o możliwe zestawy tożsamości do źródeł określonych w pliku konfiguracyjnym – `/etc/nsswitch.conf` :

Przykładowe wpisy do *etc/nsswitch.conf*

```
passwd: files ldap
shadow: files ldap
group: files ldap
```

- Interfejsem wywoływanym z powłoki do NSS jest komenda `getent` :

Kwerendy NSS (dla lokalnych i zdalnych źródeł tożsamości)

```
$ getent passwd
$ getent group
```

- Zdobywanie UID/GID jest wykonywane przez biblioteki NSS.
- Wyświetlanie informacji pomocniczych o tożsamości (np. `id`) również opiera się o NSS.

Wykonywanie komend jako inny użytkownik – *su*

```
$ su [-] [uzytkownik]
```

- `su` startuje Powłokę wskazanego użytkownika,
- brak wskazania użytkownika wartość domyślną – `root` ,
- symbol myślnika oznacza załadowanie nowego środowiska.
- Zwykli użytkownicy muszą podać hasło. `root` może wykonywać tę komendę bez hasła.
- Ze względów bezpieczeństwa większość administratorów blokuje możliwość logowania via `ssh` do użytkownika `root` . Dla zdalnego dostępu konieczne jest zalogowanie się do użytkownika z grupy `wheel` i wykonanie `su -` .

Tworzenie nowych kont użytkowników

- Możliwe bezpośrednio, edytując `/etc/passwd` oraz `/etc/shadow`.
- Istnieją dedykowane narzędzia. Dla Linuxa to `useradd`, `userdel`, `usermod`, `newgrp`, `groupadd`, `groupdel` oraz `groupmod`.

Przykład *useradd*

```
# useradd -u 1001 -g 123 -c "Jan Nowak" -ms /bin/sh nowakj
# passwd nowakj
```

- użyte opcje to:
  - `-u` – UID (domyślnie: pierwsza większa od 999, nieprzypisana do UID),
  - `-g` – GID (zachowanie domyślne różni się między dystrybucjami),
  - `-c` – GECOS (prawdziwa nazwa),
  - `-m` – utwórz katalog domowy o nazwie jak użytkownik w domyślnej lokalizacji (typowo: `/home`) oraz spopuluj go `/etc/skel/*`,
  - `-s` – domyślna Powłoka dla użytkownika.
- `/etc/default/useradd` zawiera wartości domyślne dla `useradd`.
- Pierwsze hasło konta jest domyślnie niewłaściwe.
- W `/etc/skel/` można znaleźć prototypy dla m.in. `.bashrc`, `.bash_profile`, `.bash_logout` itd.

Plan zajęć

Właściciel pliku i grupa pliku

- Każdy plik i katalog przynależy do jednego użytkownika (`user`) jednej grupy (`group`) – sprawdź wynik `ls -l` . Domyślnie są to UID oraz GID użytkownika, który stworzył plik.
- Właściciel pliku może:
  - zarządzać uprawnieniami do pliku,
  - może zmienić grupę pliku (por. nast. slajd),
  - może mieć zdefiniowany limit wielkości własnych plików (`quota`).
- Członkowie grupy inni niż właściciel pliku mogą dostać określone prawa dostępu do pliku.

Właściciel pliku i grupa pliku – *chown* i *chgrp*

- Jedynie `root` może zmienić właściciela pliku.
- Właściciele pliku mogą zmienić grupę pliku na dowolną, do której należą.
- Komendy `chown` i `chgrp` pozwalają na zmianę właściciela/grupy pliku/katalogu. Z opcją `-R` zmiana jest rekursywna, na zawartość katalogu.

Syntaks *chown* i *chgrp*

```
# chown [-R] uzytkownik[:grupa] plik
$ chgrp [-R] grupa plik
```

Prawa dostępu

- Unix definiuje 3 poziomy praw dostępu (oznaczane `rwx`):
  - `r` (*Read* – odczyt) pozwala na odczyt treści pliku oraz jego metadanych;
  - `w` (*Write* – zapis) pozwala na zmianę zawartości pliku;
  - `x` (*eXecute* – wykonanie) pozwala na wykonanie pliku jako programu.
- Każde prawo dostępu definiuje się na 3 poziomach: właściciela (`u` – *User*), grupy (`g` – *Group*) i reszty świata (`o` – *Others*):
  - `u` etykietuje prawa dostępu dotyczące właściela. Są wyświetlane jako pierwsze, na pozycjach 1-3;
  - `g` etykietuje prawa dostępu członków grupy, do której należy plik, innych niż właściciel. Są wyświetlane jako drugie, na pozycjach 4-6;
  - `o` etykietuje prawa dostępu pozostałych użytkowników. Są wyświetlane jako trzecie, na pozycjach 7-9.
- Prawa dostępu `ugo` reprezentuje się jako ciąg postaci `rwxrwxrwx` (każdy ze znaków może być zastąpiony `-`, który reprezentuje brak takiego prawa, por. `ls -l` na pliku).

Prawa dostępu – przykłady

- `rwxr-xr-x` (rekomendowany dla programów) właściciel pliku może zrobić z plikiem wszystko, a inni użytkownicy mogą go odczytać oraz wykonać;
- `rw-r--r--` (rekomendowany dla publicznych danych) wszyscy użytkownicy mogą go odczytać, tylko właściciel może go zmodyfikować;
- `rw-------` (rekomendowany dla prywatnych danych) właściciel może go odczytać i modyfikować, pozostali użytkownicy mają brak dostępu;
- `rw-rw-r--` właściciel oraz członkowie grupy mogą odczytywać i modyfikować plik, pozostali użytkownicy mogą go tylko odczytać;
- `rwx---r-x` wszyscy poza członkami grupy mogą plik odczytać oraz wykonać, właściciel może go ponadto modyfikować.

- Uprawnienia działają na tożsamej zasadzie na poziomie katalogów, ale znaczenie poszczególnych symboli jest nieco inne:
  - `r` pozwala na listowanie zawartości katalogu;
  - `w` pozwala na modyfikację zawartości katalogu, tj. zmianę nazw istniejącej zawartości oraz zapisywanie (wkopiowywanie, przesuwanie) plików do katalogu;
  - `x` pozwala na wykonanie na katalogu `find`, `cd` i in.
- Pełen zakres praw odczytu wymaga ustawień `r-x`
- Ustawienie prawa na katalogu na poziomie `--x` nie pozwala na wykonanie `ls` na katalogu. Wciąż jest możliwy jednak dostęp do plików/podkatalogów w tym katalogu, pod warunkiem, że ich nazwy są znane, a ich uprawnienia wystarczające.

Zmiana praw dostępu – *chmod*

- Komenda `chmod` pozwala na zmianę praw dostępu pliku/katalogu. Syntaks jest zbliżony do syntaksu `chown` / `chgrp` :

Syntaks *chmod*

```
$ chmod [-R] uprawnienia plik
```

- Uprawnienia mogą być zdefiniowane:
  - w trybie symbolicznym, albo
  - w trybie numerycznym.

chmod (tryb symboliczny)

- Symboliczny ciąg uprawnień to lista rodzielonych przecinkiem trójek:
  - *klasy*
    - ★ `u` – właściciel,
    - ★ `g` – grupa,
    - ★ `o` – reszta,
    - ★ `a` albo brak referencji – wszyscy użytkownicy (`ugo`);
  - *operatora*
    - ★ `=` – nastaw,
    - ★ `+` – dodaj,
    - ★ `-` – usuń,
  - i *uprawnień* określanych notacją `rwx` .
- Ustawienia klas nienazwanych pozostają niezmienione.
- Dodawanie uprawnień, które istnieją, jest ignorowane. Podobnie z usuwaniem uprawnień, których nie ma.

chmod (tryb symboliczny) – przykłady

Przykłady *chmod*

```
$ chmod u=rw,g=r,o=r plik      # ustawia rw-r--r--
$ chmod u=rw,go=r plik         # to samo, co wyżej
$ chmod go-rwx plik_niejawny   # ustawia ???------
$ chmod +x skrypt              # ustawia ??x??x??x
```

- Pamiętaj, że znaki białe są niedozwolone w ciągach symbolicznych:

Przykłady *chmod*

```
$ chmod u=rw,g=r,o=r plik     # ustawia rw-r--r--
$ chmod u=rw, g=r, o=r plik   # błąd!
```

chmod (tryb numeryczny)

- Liczby `4`, `2` oraz `1` reprezentują uprawnienia `r`, `w` oraz `x`, odpowiednio.
- Uprawnienia każdej klasy (`u`, `g` oraz `o`) są wyrażalne przez pojedynczą liczbę będącą sumą uprawnień dla danej klasy, tj. `rwx` jest reprezentowane przez `7`, `rw` – `6`, `rx` – `5`, `wx` – `3`.
- W odróżnieniu od trybu symbolicznego `chmod`, w trybie numerycznym można nastawić (zmodyfikować) jedynie wszystkie uprawnienia jednocześnie (żadne nie pozostaje niezmienione).

Przykłady *chmod*

```
$ chmod 755 plik  # ustawia rwxr-xr-x
$ chmod 700 plik  # ustawia rwx------
$ chmod 644 plik  # ustawia rw-r--r--
```

Uprawnienia specjalne

- Poza prawami `rwx` dla klas `ugo` , istnieją trzy dodatkowe flagi/uprawnienia: *setuid*, *setgid* i *sticky bit*.
- Procesy – podobnie do plików – mają swoich właścicieli oraz grupę. Domyślnie jest to UID oraz GID użytkownika, który uruchomił proces. *Setuid* (`s`) oraz *setgid* (`s`) rozszerzają/modyfikują standardowe uprawnienie wykonywalności programów – który startuje z *efektywnym UID/GID*, zamiast równym UID/GID użytkownika, który go wykonał, to z UID/GID właściciela/grupy pliku.
- Przykład: `/usr/bin/passwd` używany do zmiany hasła ma setuid i odpalany przez zwykłych użytkowników ma efektywny UID roota.
- Sticky bit (`t`) katalogu zabezpiecza przed usunięciem/zmianą nazwy pliku w nim (zmiany dopuszczalne jedynie dla właściciela).
- Przykład: `/tmp` ma ustawiony sticky bit.
- Te flagi specjalne są reprezentowane jak uprawnienia:
  - setuid – `rwsr-xr-x` ,
  - setgid – `rwxr-sr-x` ,
  - sticky bit – `rwxrwxrwt` .

Uprawnienia specjalne – cd.

- W trybie symbolicznym flagi te ustawia się jak zwykłe prawa.

Ustawienie specjalnych uprawnień symbolicznie

```
$ chmod +x,u+s program
$ chmod +x,g+s program
$ chmod o+t katalog
```

- W trybie numerycznym używa się dodatkowej liczby (w prefiksie):

Ustawienie specjalnych uprawnień numerycznie

```
$ chmod 4755 program
$ chmod 2755 program
$ chmod 1777 katalog
```

- Flagi *setuid*/*setgid*
  - mają sens jedynie dla plików binarnych; są ignorowane dla skryptów,
  - są automatycznie wycofywane przy zmianie zawartości pliku,
  - potencjalnie wprowadzają krytyczną podatność systemu – należy dawać je tylko małym i bezpiecznym programom.

Ramowe uprawnienia do plików

- Każdy program ma możliwość nadania dowolnych ustawień nowo utworzonym plikom/katalogom. Na przykład kompilator nadaje uprawnienie `x`, zaś edytor tekstu – nie.
- Prawa zdefiniowane w `umask` (ramowe uprawnienia) są wówczas wycofywane z postulowanych uprawnień. Zwykle wartość ramowych uprawnień jest postaci `0022`, które wycofują uprawnienie `w` wszystkim poza właścicielowi, albo `0002`.
- Użytkownicy mogą sprawdzić/zmienić swój poziom ramowych uprawnień do pliku za pomocą komendy `umask`.

  ✍ *`touch` tworzy nowe pliki z uprawnieniami na poziomie `0666`*

Umask i prawa dostępu

```
$ umask 0000; touch a
$ umask 0777; touch b
$ ls -l
 -rw-rw-rw- 1 pawel pawel 0 lis 14 07:04 a
 ---------- 1 pawel pawel 0 lis 14 07:04 b
```

Plan zajęć

Listy kontroli dostępu

- Istotnym ograniczeniem standardowego modelu przyznawania uprawnień jest to, że prawa dostępu do pliku są przyznawane tylko do jednej, zdefiniowanej z góry (przez użytkownika `root`), grupy.
- Współczesne systemy Uniksowe dopuszczają implementację *List kontroli dostępu* (`ACL` – *Access Control List*). Pozwala na zindywidualizowane nadawanie dostępów użytkownikom i grupom.
- Symbol plusa po standardowej reprezentacji uprawnień do pliku (np. `rwxr-xr-x+`) reprezentuje fakt, że plikowi określono dodatkowe prawa dostępu za pomocą ACL.
- ACL używa się za pomocą dwóch komend: `getfacl` oraz `setfacl`.
- Niektórye systemy plików (na przykład: NFS4) nie wspierają ACL. Inne wymagają jawnego zadeklarowania, by włączyć tę funkcjonalność.
- Ponad *zwykłe* ustawienia ACL, dopuszcza się też ustawienie *domyślnego* ACL dla katalogu. Oznacza ono ACL, które będzie aplikowane do przyszłej zawartości katalogu.
- Ustawienie *zwykłego*/*domyślnego* ACL dla katalogu to dwa oddzielne, niezależne zadania.

Używanie ACL – przykłady

✍ *Ustawienie (`-m` : modyfikacja) ACL na katalogu i jego zawartości (`-R` : rekursywnie)*

Modyfikacja wpisów ACL

```
$ setfacl -Rm u:nowaka:rwx,u:kowalskib:rx katalog
```

✍ *Ustawienie (`-m` : modyfikacja) domyślnego (`-d` : default) ACL na katalogu i jego zawartości (`-R` : rekursywnie)*

Modyfikacja wpisów ACL

```
$ setfacl -Rdm u:nowaka:rwx,u:kowalskib:rx katalog
```

✍ *Usunięcie (`-x`) wpisów ACL dla zadanych użytkowników/grup z pliki*

Modyfikacja wpisów ACL

```
$ setfacl -x u:nowaka,g:studenci plik
```