Praca z systemami plików Linuksa

Witold Paluszyński
Katedra Cybernetyki i Robotyki
Politechnika Wrocławska
http://www.kcir.pwr.edu.pl/~witold/

2013

Podstawowe atrybuty plików

```
chmod 640 /var/log/maillog
chmod u=rw,g=r,o= /var/log/maillog
chmod -R o-r /home/*
chmod u+s /path/to/prog

find / -perm -u+s -print

chown user:group /path/to/file
chgrp group /path/to/file

chmod 640 `find ./ -type f -print`
chmod 751 `find ./ -type d -print`
```

inode: struktura stat

```
stat nazwa_pliku      # wyśw. strukturę inode pliku/kat.

find . -inum 84213815 # znajdź plik o danym numerze inode
```

Linux: rozszerzone atrybuty plików

Rozszerzone atrybuty plików pozwalają przechowywać w bloku kontrolnym pliku dodatkowe dane o postaci: nazwa,wartość.

W większości systemów potrzebne jest doinstalowanie odpowiedniego wsparcia (narzędzi) oraz zamontowanie systemu plików ze specjaną opcją `user_xattr`

Przykładem zastosowania rozszerzonych atrybutów są listy praw dostępu ACL.

Narzędzia do sprawdzania i ustawiania rozszerzonych atrybutów: `getfattr` i `setfattr`

Zdefiniowane kategorie rozszerzonych atrybutów: trusted, security, system, user

```
touch probny
setfattr -n user.comment -v "this is a comment" probny
getfattr probny
```

`http://www.linux-mag.com/id/8741/`

System plików: listy praw dostępu ACL

Zarządzanie uprawnieniami za pomocą 9 bitów praw dostępu, nawet w połączeniu z mechanizmem grup nie daje możliwości potrzebnych w wielu sytuacjach. Dlatego w systemach Unix w pewnym momencie zaimplementowano listy praw dostępu (ACL — *Access Control List*) dla plików, znane z innych systemów operacyjnych. Poza podstawowymi dziewięcioma bitami określającymi prawa dostępu, pliki i katalogi mogą mieć listę ACL, która rozszerza lub ogranicza te prawa dostępu. Listy ACL rozszerzają możliwości nadawania, lub odbierania, konkretnych uprawnień konkretnym użytkownikom lub grupom.

Dany plik lub katalog może posiadać listę ACL lub nie. Można to poznać po znaku plus po podstawowym ciagu praw dostępu na listingu `ls -l`:

```
sequoia-582> ls -l Unix_shell.pyt
-rw-r--r--+  1 witold   staff      39244 lut  6  2008 Unix_shell.pyt
```

Listy praw dostępu ACL: prawa dla użytkowników

Lista ACL może zawierać szereg wpisów dla dowolnie wybranych użytkowników i dowolnie wybranych grup, wprowadzających nowe uprawnienia dla danego użytkownika lub grupy.

```
sequoia-583> getfacl Unix_shell.pyt
# file: Unix_shell.pyt
# owner: witold
# group: staff
user::rw-
user:ekr:r--            #effective:r--
group::r--              #effective:r--
mask:r--
other:r--
```

Podobnie jak w przypadku podstawowych praw dostępu, gdy istnieje wpis dla danego użytkownika, to jest on stosowany niezależnie od wpisów dla grup, lub innych. Dzięki temu, można np. odebrać konkretne uprawnienia konkretnemu użytkownikowi z jakiejś grupy, nie usuwając uprawnień dla całej grupy.

Listy praw dostępu ACL: uprawnienia domyślne

Możliwe jest również tworzenie wpisów domyślnych (*default*), które dla katalogów określa domyślne prawa dla tworzonych w nim plików.

Wszystkie pliki i podkatalogi utworzone następnie w katalogu z ustawioną listą ze wpisem domyślnym będą miały uprawnienia z tego wpisu. Będzie to zrealizowane za pomocą zwykłych bitów praw dostępu, a jeśli to niemożliwe, to przez utworzenie listy ACL dla każdego z nich.

Listy praw dostępu ACL: maska

Maska filtruje wpisy dla nazwanych użytkowników i wszystkich grup (właściciela i nazwanych). Nie filtruje ona wpisów dla właściciela ani innych użytkowników:

```
sequoia-584> setfacl -m user:ekr:rw- Unix_shell.pyt
sequoia-585> getfacl Unix_shell.pyt
# file: Unix_shell.pyt
# owner: witold
# group: staff
user::rw-
user:ekr:rw-            #effective:r--
group::r--              #effective:r--
mask:r--
other:r--
sequoia-586> setfacl -m mask:rwx Unix_shell.pyt
sequoia-587> getfacl Unix_shell.pyt
# file: Unix_shell.pyt
# owner: witold
# group: staff
user::rw-
user:ekr:rw-            #effective:rw-
group::r--              #effective:r--
mask:rwx
other:r--
```

Zajętość miejsca na dysku

```
du -sh *                # rozmiar plików/katalogów
du -csh                 # calkowity rozmiar bieząc.katalogu
du -ks * | sort -nr     # sortuje wszystko wg rozmiaru w kb
ls -lSr                 # lista plików, największe na końcu

df                      # rozmiar i wolne miejsce syst.plik.
```

System disk quota

koncepcja: limit twardy, limit miękki, okres karencji

```
edquota -u USER
edquota -g GROUP
repquota  -a
quotacheck
```

Dyski i partycje dyskowe

```
hdparm -I /dev/sda      # informacje o dysku IDE/ATA
fdisk /dev/sda          # przeglądanie/edycja tablicy partycji
smartctl -a /dev/sda    # wyświetla informacje SMART dysku
```

Własności systemów plików

- ext2
- ext3
- jfs
- xfs
- raiserfs
- ext4

journaling

Operacje na systemach plików

Operacje na systemach plików:

- tworzenie (*mkfs*)
- montowanie (*mount*)
- odmontowanie (*umount*)
- sprawdzanie spójności i naprawa (*fsck*)
- defragmentacja (*e4defrag*)

Montowanie systemów plików

```
mount /cdrom
mount                   # lista zamontowanych systemów plików
```

Identyfikatory UUID

W pliku `/etc/fstab` (Solaris: `/etc/vfstab`) znajduje się lista znanych systemów plików, które mogą być automatycznie montowane w czasie startu systemu. Te systemy plików tradycyjnie są identyfikowane nazwą partycji dyskowej, typu `/dev/sda7` lub `/dev/dsk/c0t3d0s7`.

Ten schemat może nie działać poprawnie jeśli dany dysk zostanie inaczej podłączony do systemu, i w czasie startu zostanie inaczej przenumerowany. W związku z tym nowe Linuksy stosują uogólniony schemat identyfikacji pozwalający określać w pliku `/etc/fstab` systemy plików przez:

- przez nazwę partycji dyskowej, jak wcześniej,
- przez etykietę systemu plików (którą można nadać poleceniem `e2label`),
- przez identyfikator UUID (*Universally Unique Identifier*).

Identyfikatory UUID są unikalnie generowanymi identyfikatorami wpisywanymi do systemu plików i pozwalającymi je rozpoznawać bez kolizji nazw. Polecenie `blkid` wyświetla UUID systemów plików zamontowanych w systemie.

Specjalne systemy plików

Posiadając obraz systemu plików w postaci pliku, możemy chcieć przeglądać go, i/lub modyfikować, jako drzewo katalogów. Wymaga to zamontowania systemu w określonej lokalizacji. Aby zamontować partycję dyskową program mount wywołuje driver odpowiedniego urządzenia sprzętowego.

Linux posiada szereg driverów pseudourządzeń, pozwalających operować na specjalnych systemach plików:

- loop — pozwala tworzyć systemy plików w plikach i na urządzeniach
- tmpfs — pozwala tworzyć systemy plików w pamięci RAM

```
# tworzy ramdisk o rozmiarze 64m i montuje w systemie
mount -t tmpfs -osize=64m tmpfs /memdisk
```

- sshfs — pozwala tworzyć systemy plików przez połączenia ssh

Pseudourządzenie loop

```
# montuj obraz plyty CD wykorzystujac wolne urzadzenie loop
mount -t iso9660 -o loop Win_XP_Prof_SP3_x86_PL.iso /mnt

# utworz plik o wielkosci 100MB na obraz syst.plikow
dd if=/dev/zero of=/tmp/fs.img bs=1024k count=100
# sprawdz wykorzystywane urzadzenia loop
losetup -a
# utworz urzadzenie loop3 i powiaz z utworzonym obrazem
losetup /dev/loop3 /tmp/fs.img
# utworz nowy pusty system plikow ext3
mkfs.ext3 /dev/loop3
# montuj utworzony system plikow
mkdir -p /tmp/fs
mount -t ext3 /dev/loop3 /tmp/fs

# procedura odwrotna
umount /tmp/fs
losetup -d /dev/loop3
rm /tmp/fs.img
```

Systemy plików w przestrzeni użytkownika

fusermount

Kto otworzył dany plik?

Zdarza się, że w czasie pracy z systemem chcemy stwierdzić, który proces korzysta z określonego pliku. Być może system nie może odmontować systemu plików bo jakiś proces z niego korzysta, albo chcemy bezpiecznie przenieść plik w inne miejsce, skompresować. Lub ogólnie chcemy określić proces, który otworzył konkretny plik lub pliki w danym obszarze. W uniksowych systemach plików egzystują, poza zwykłymi plikami i katalogami: potoki, gniazdka, pliki specjalne urządzeń, i inne rodzaje plików.

Zauważmy, że jest to problem odwrotny do pytania, jakie pliki otworzył dany proces. Ten drugi problem jest łatwiejszy, ponieważ system posiada listę plików wykorzystywanych przez dany proces, i można je łatwo odnaleźć np. w systemie /proc. Na odwrót, odnalezienie procesu, który korzysta z danego pliku zawsze wymaga przeszukiwania, ponieważ takich tabel system nie utrzymuje. Jednak są programy, które to ułatwiają i warto się z nimi zapoznać.

UWAGA: poszukiwanie procesów, które mają otwarty dany plik jest limitowane zwykłymi prawami dostępu. Pełną informację można uzyskać tylko z poziomu użytkownika root.

Program fuser

Program `fuser` pochodzi z Uniksa, ale istnieje również jego wersja GNU. Wyświetla numery i informacje o procesach, albo wysyła sygnał do procesów, które używają określonego pliku jako:

- zwykłego pliku, potoku, gniazdka, itp., otwartego przez proces
- katalogu bieżącego
- programu binarnego, który wykonuje dany proces.

```
fuser .
fuser /usr/bin/gnome-session
fuser /tmp/orbit-witold/*
fuser /
```

Wersja GNU programu `fuser` posiada również zdolność odnajdowania procesów wykorzystujących porty sieciowe zadane jako: `loc-port,rem-host,rem-port`

```
fuser -n tcp 22             # procesy zdalnych polaczen ssh
fuser -n tcp ,,22           # proc.ktore otworzyly pol. ssh
fuser -n tcp ,156.17.9.3    # proc.otwarte z okresl. adresu
```

Program lsof

Lsof jest programem o podobnych funkcjach jak fuser. Ma jednak nieporównanie więcej możliwości. Domyślnie lsof wyświetla nie tylko numer znalezionego procesu, ale wiersz informacji o nim w stylu programu ps. W braku opcji wywołania wyświetla informacje o wszystkich plikach otwartych przez wszystkie procesy w systemie.

Przykłady:

```
# procesy danego uzytkownika i ich otwarte pliki
lsof -u witold

# procesy otwierajace pliki w katalogach uzytkownikow
lsof +D /home

# procesy korzystajace z bibliotek w katalogu /usr/lib
lsof /usr/lib/lib*

# aby pominac wielokrotne przypadki jednej biblioteki
lsof /usr/lib/lib* | sort -k 9,9 -k 1,1 | uniq -f 8
```

```
# otwarte lokalne porty sieciowe w zakresie 1-1024
lsof -i TCP:1-1024

# polaczenie do zdalnego portu sieciowego
lsof -i TCP@156.17.9.3:22

# polaczenie z komputera o okreslonym adresie
lsof -i TCP@156.17.9.3

# aktywne porty sieciowe, polaczone i nasluchujace
```

Kopiowanie plików, katalogów, systemów plików, dysków

Problemy związane z kopiowaniem plików: zachowanie czasów dostępu, praw dostępu, list ACL, rozszerzonych atrybutów, linków (twardych), kopiowanie plików rzadkich.

Kopiowanie katalogów, narzędzia: cp, cpio, tar, rsync

Kopiowanie systemów plików: logiczne lub fizyczne

```
dd if=/dev/sda1 bs=1k conv=sync,noerror |\
 gzip -c |\
 ssh -c blowfish user@hostname \
     "gunzip -c | dd of=/dev/sda5 bs=1k"
```

Kopiowanie dysków: sprawdzenie block size, kopiowanie tablicy partycji, kopiowanie MBR.

Kasowanie zawartości dysku: zamazywanie.

Odzyskiwanie skasowanych plików

W systemach uniksowych odzyskiwanie skasowanych plików jest trudne, bo bloki dyskowe zwolnione ze skasowanych plików są od razu przyłączane do listy wolnych bloków i przydzielane nowo tworzonym lub zapisywanym plikom.

Na obciążonym systemie, gdzie wiele procesów na bieżąco tworzy i zapisuje pliki, odzyskanie skasowanego pliku może być niemożliwe. Szczególnie trudne jest odzyskiwanie plików skasowanych na systemowym systemie plików. Dlatego dobrym rozwiązaniem jest oddzielenie systemu plików użytkowników (`/home`) od systemowego (/), albo oddzielenie systemu szybkozmiennego (`/var`).

Zasady:

1. najlepiej od razu odmontować partycję, gdzie został skasowany plik, a jeśli jest to partycja systemowa to zamknąć system, najlepiej w sposób „nagły" aby maksymalnie zapobiec pisaniu przez procesy informacji do plików (np. ALT+PrtSc/SysRq+u i potem ALT+PrtSc/SysRq+o)
2. jeśli system został zastopowany to zabootować np. „live CD",
3. jeśli to możliwe i praktyczne, zrobić kopię zapasową całego systemu plików,
4. rozpocząć odzyskiwanie, przy czym jeśli następuje ono z aktywnego systemu plików to zapis odzyskiwanych plików musi być na innym systemie plików.

Narzędzia do odzyskiwania skasowanych plików:

- debugfs
- foremost — odzyskuje pliki określonego typu na podstawie charakterystycznych sekcji, np.:
  `foremost -s 512 -t jpeg -i /dev/sda5 -o /tmp/recovered/`
- ext3grep — narzędzie do analizy systemów plików ext3 i odzyskiwania skasowanych plików, np.:
  `ext3grep --restore-all /dev/sda5`
- extundelete
- scalpel
- photorec

Przydatne linki

Colin Barschel
`http://cb.vu/unixtoolbox.xhtml#filesystem`
Polska wersja (niekoniecznie najnowsza)
`http://cb.vu/unixtoolbox_pl.xhtml#filesystem`