Ćwiczenie nr 1

Cel ćwiczenia

Celem ćwiczenia jest poznanie zastosowań podstawowych poleceń systemu operacyjnego Linux.

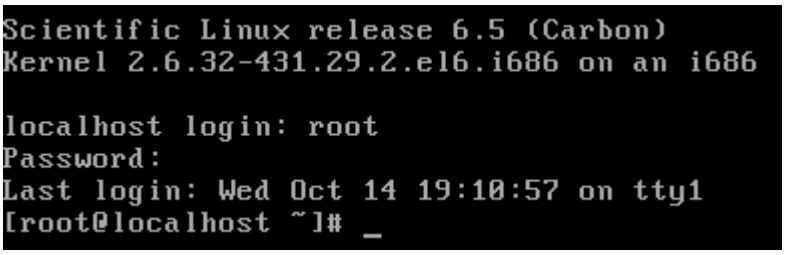
Proszę zapoznać się z praktycznym działaniem każdego z wyszczególnionych tutaj poleceń systemu Linux. Polecana literatura [1,2] pomocne mogą okazać się także publikacje internetowe. [3,4] Użycie każdego polecenia należy udokumentować w sprawozdaniu, które powinno zawierać nazwę wydawanego polecenia i jego krótki opis, rezultaty działania polecenia oraz opis otrzymanych rezultatów. Przy tworzeniu sprawozdania wspomóc się można plikiem logu rejestrującego działanie poszczególnych poleceń. Jednakże, należy pamiętać, aby nie tworzyć zbyt dużych logów bowiem trudno będzie je przeglądać.

Sprawozdanie w postaci pliku PDF proszę przesłać na mój adres e-mailowy: jpszub@matman.uwm.edu.pl podając w tytule e-maila: „ASK Lab1 i swoje nazwisko”.

Dostęp do systemu

W celu uzyskania dostępu do systemu Linux należy otworzyć sesję, aby to zrobić użytkownik musi mieć przydzielone konto, które jest przydzielane przez administratora systemu.

Przyjmujemy, że konto zostało przydzielone. Gdy ukaże się prompt: `login` należy się zalogować podając swój login i hasło.



Domyślnie: `login: root`
`password:” root`

Jeśli nie można się zalogować i nie znamy aktualnego hasła postępujemy w następujący sposób:

1. podczas uruchamiania loadera `grub` (wybieranie systemu przed uruchomieniem) wciskamy "escape",
2. podświetlony jest używany system;
3. wybieramy opcję "e";
4. najeżdżamy na druga linijkę i ponownie wybieramy "e";
5. Na końcu tekstu, który się pojawi dopisujemy `" -s"` (sapacja minus s);
6. wciskamy „Enter”;
7. wybieramy opcję "b";
8. system uruchamia się w trybie `single user` zalogowany jako `root`,
9. aby zmienić hasło wpisujemy polecenie "`passwd`" i podajemy nowe hasło.

```
[root@localhost ~]# passwd
Zmienianie hasła użytkownika root.
Nowe hasło :
Proszę ponownie podać nowe hasło :
Podane hasła nie zgadzają się.
Nowe hasło :
BŁĘDNE HASŁO: za krótkie
BŁĘDNE HASŁO: jest za proste
Proszę ponownie podać nowe hasło :
passwd: zaktualizowanie wszystkich tokenów uwierzytelniania powiodło się.
[root@localhost ~]# _
```

10. wykonujemy polecenie `init 6` (restart systemu)[i]

Aby zakończyć sesję należy wydać polecenie `logout`, które działa jedynie dla „login shells" lub `exit`. Można także zastosować kombinację klawiszy `CTRL+D`.

Polecenie `last` przegląda plik `/var/log/wtmp` (lub plik oznaczony flagą `-f`) i wyświetla listę wszystkich użytkowników zalogowanych do systemu i wylogowanych począwszy od momentu utworzenia tego pliku.

```
reboot   system boot  Wed Oct  8 20:19 - 11:43 (7+15:24)    2.6.32-431.el6.i686
root     tty1         Wed Oct  8 20:01 - crash  (00:17)
reboot   system boot  Wed Oct  8 20:01 - 11:43 (7+15:42)    2.6.32-431.el6.i686
root     tty1         Wed Oct  8 19:56 - 20:00  (00:03)
root     tty1         Wed Oct  8 19:55 - 19:56  (00:00)
reboot   system boot  Wed Oct  8 19:54 - 11:43 (7+15:49)    2.6.32-431.el6.i686

wtmp begins Wed Oct  8 19:54:01 2014
```

Zadanie 1

Wykonać zrzut ekranu dokumentujący historię logowania i wylogowania (niekoniecznie wszystkich) użytkowników systemu.

Polecenia powłoki systemu

Ogólna składnia poleceń systemu jest następująca:

```
polecenie [- opcje] [argumenty] […]
```

Opcje służą do modyfikacji działania poleceń powłoki. W niektórych przypadkach można grupować w jedną zbiorczą opcję, jak np. w przypadku polecenia `who -mH`.

Informacja o bieżącej sesji

Polecenie `who` wyświetla informacje o wszystkich aktualnie zalogowanych użytkownikach.:

[i] Polecenie `init 6` ponownie uruchamia system, najpierw uruchamiane są wszystkie skrypty zamykające, a następnie ponownie się uruchamiany jest system. Polecenie `reboot` wykonuje bardzo szybki restart, odmontowując systemy plików i ponownie uruchamiając system.

```
[root@localhost ~]#
[root@localhost ~]# who
root     tty1         2015-10-14 19:42
[root@localhost ~]# who -umH
UŻYTKOWNIK TERM         CZAS              BEZCZYNNY        PID KOMENTARZ
root     tty1         2015-10-14 19:42    .          868
[root@localhost ~]# whoami
root
[root@localhost ~]#
```

Znaczenie opcji:

`-u` or `--users` wyświetla czas bezczynności dla każdego użytkownika i identyfikator procesu,

`-m` wyświetl jedynie informację o użytkowniku i hoście stowarzyszonym ze standardowym wejściem (terminalem), na którym użyto tego polecenia,

`-H`, or `--heading` wyświetl linię nagłówków kolumn.

Zadanie 2

Wykonaj zrzut ekranu prezentujący zalogowanych użytkowników i charakterystykę uruchomionych przez nich procesów.

Tworzenie dziennika sesji powłoki

Polecenie `script` umożliwia utworzenie dziennika (logu) rejestrującego wszystkie wykonywane podczas sesji polecenia oraz rezultaty ich działania. W celu zakończenia rejestracji zdarzeń w dzienniku należy użyć polecenia `exit`.

```
[root@localhost ~]#
[root@localhost ~]# script
Skrypt uruchomiony, plik to typescript
[root@localhost ~]# ls
anaconda-ks.cfg  install.log.syslog  plik.cap      prosty1   typescript
file.cap         jacek               procroot.txt  root.txt
install.log      log.20151011        prosty        test
[root@localhost ~]# ls -F
anaconda-ks.cfg  install.log.syslog  plik.cap      prosty1*  typescript
file.cap         jacek               procroot.txt  root.txt
install.log      log.20151011        prosty        test/
[root@localhost ~]# pwd
/root
[root@localhost ~]# exit
exit
Skrypt wykonany, plik to typescript
[root@localhost ~]#
```

Domyślnie wpisy do dziennika przechowywane są w pliku `typescript` w bieżącym katalogu. Można podać inną nazwę i lokalizacje tego pliku.

Przykładowa zawartość pliku `typescript`:

```
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# cat typescript
Skrypt uruchomiony śro, 14 paź 2015, 19:23:01
[root@localhost ~]# ls
anaconda-ks.cfg  install.log.syslog  plik.cap      prosty1   typescript
file.cap         jacek               procroot.txt  root.txt
install.log      log.20151011        prosty        test
[root@localhost ~]# ls -F
anaconda-ks.cfg  install.log.syslog  plik.cap      prosty1*  typescript
file.cap         jacek               procroot.txt  root.txt
install.log      log.20151011        prosty        test/
[root@localhost ~]# pwd
/root
[root@localhost ~]# exit
exit

Skrypt wykonany śro, 14 paź 2015, 19:23:27
[root@localhost ~]#
```

Pierwszy i ostatni wiersz pliku dziennika zawiera informacje o czasie rozpoczęcia i zakończenia przechwytywania sesji powłoki.

Zadanie 3

Załóż dziennik sesji powłoki i udokumentuj jego zawartość po zakończeniu aktualnej sesji związanej z wykonywaniem poleceń zawartych w opisie ćwiczeń ASK Lab1.

Nawigacja w systemie plików

Ścieżka, to ogólna forma nazwy pliku lub katalogu, określa ona w sposób jednoznaczny lokalizacją w systemie plików.
A ścieżka wskazuje na lokalizację w systemie plików przez następujący w hierarchii drzewa katalogów ciąg znaków, w którym elementy ścieżki, oddzielone znakiem separatora, reprezentujących poszczególne katalogu. Najczęściej jest to ukośnik wsteczny "/" backslash.

Ścieżki są używane do reprezentowania relacji katalogów/plików w systemach operacyjnych, są także niezbędne do budowy lokalizatorów w sieci www (Uniform Resource Locator URL).

- Ścieżka bezwzględna: jest to droga, która wskazuje lokalizację w systemie plików niezależnie od katalogu bieżącego. Zwykle w odniesieniu do katalogu głównego.
- Ścieżka względna zaczyna się od jakiegoś katalogu roboczego. Nazwa pliku może być traktowane jako ścieżka względna oparta na bieżącym katalogu roboczym. Jeśli katalog roboczy nie jest katalogiem nadrzędnym pliku, wystąpi błąd - pliku nie znaleziono.
- Katalog domowy to katalog systemu plików w systemie operacyjnym dla wielu użytkowników zawierający pliki danego użytkownika systemu.

Root Directory	Directory Separator	Current Directory	Parent Directory	Home Directory
/	/	.	. .	~

Lista standardowych katalogów w systemie Linux:

- `/bin` - przechowywane są w nim standardowe polecenia systemu linux,
- `/dev` - przechowywane są w nim pliki reprezentujące punkty dostępu do urządzeń

systemowych,

- `/etc` - przechowywane są w nim administracyjne pliki konfiguracyjne,
- `/home` - przechowywane są w nim katalogi przypisane do poszczególnych użytkowników,
- `/mnt` - zapewnia odpowiednią lokalizację do montowania urządzeń, takich jak zdalne systemy plików oraz nośniki wymienne,
- `/root` - jest to katalog domowy administratora systemu,
- `/sbin` - przechowywane są w nim polecenia służące do administrowania systemem oraz uruchamiające procesy tzw. demonów (ang. daemon),
- `/temp` - przechowywane są w nim pliki tymczasowe, wykorzystywane przez różne aplikacje,
- `/usr` - przechowywane są w nim pliki dokumentacji systemu, pliki graficznego interfejsu użytkownika,
- `/var` - przechowywane są w nim katalogi danych różnych aplikacji, w szczególności takich jak serwer FTP (katalog /var/ftp) czy Server WWW (katalog /var/www).

Operacje na plikach i katalogach

- Polecenie `cd` umożliwia zmianę katalogu bieżącego na katalog będący argumentem polecenia, np. `cd ~/` umożliwia przejście do katalogu domowego użytkownika, `cd /` umożliwia przejście do katalogu głównego (ang. root directory):

```
[root@localhost ~]#
[root@localhost ~]# pwd
/root
[root@localhost ~]# _
```

- Polecenie `pwd` wyświetla ścieżkę do katalogu bieżącego:

```
[root@localhost test]#
[root@localhost test]# pwd
/root/test
[root@localhost test]# _
```

- Zmienna `$HOME` zawiera nazwę katalogu domowego:

```
[root@localhost ~]#
[root@localhost ~]# echo "$HOME"
/root
[root@localhost ~]#
```

- Polecenie `mkdir` umożliwia utworzenie nowego katalogu:

```
[root@localhost ~]# mkdir jacek
[root@localhost ~]# ls -F
anaconda-ks.cfg  install.log.syslog  plik.cap      prosty1*  typescript
file.cap         jacek/              procroot.txt  root.txt
install.log      log.20151011        prosty        test/
[root@localhost ~]# _
```

Zadanie 4

Utwórz nowy katalog o wybranej nazwie.

- Polecenie `touch` służy do zmiany stempli czasowych: czas dostępu i modyfikacji tworzonych lub istniejących plików. Przy jego użyciu możliwe jest tworzenie nowych plików, np.:

```
[root@localhost test]# touch proba4
[root@localhost test]# ls -lF
razem 4
-rwxr-----. 1 root root 1220 2014-10-09  proba*
-rw-r--r--. 2 root root    0 2014-10-09  proba1
-rw-r--r--. 2 root root    0 2014-10-09  proba2
lrwxrwxrwx. 1 root root    6 10-14 21:55 proba3 -> proba1
-rw-r--r--. 1 root root    0 10-14 21:58 proba4
[root@localhost test]# _
```

Zadanie 4

We wcześniej utworzonym katalogu utwórz kilka nowych plików.

- Polecenie `cp` umożliwia kopiowanie plików wywołanie jest następujące: `źródło ->cel`:

```
[root@localhost test]# cp proba4 proba5
[root@localhost test]# ls -lF
razem 4
-rwxr-----. 1 root root 1220 2014-10-09  proba*
-rw-r--r--. 2 root root    0 2014-10-09  proba1
-rw-r--r--. 2 root root    0 2014-10-09  proba2
lrwxrwxrwx. 1 root root    6 10-14 21:55 proba3 -> proba1
-rw-r--r--. 1 root root    0 10-14 21:58 proba4
-rw-r--r--. 1 root root    0 10-14 22:00 proba5
[root@localhost test]# _
```

- Polecenie `mv` umożliwia przeniesienie plików lub katalogów:

```
[root@localhost test]# mv proba5 proba6
[root@localhost test]# ls -lF
razem 4
-rwxr-----. 1 root root 1220 2014-10-09  proba*
-rw-r--r--. 2 root root    0 2014-10-09  proba1
-rw-r--r--. 2 root root    0 2014-10-09  proba2
lrwxrwxrwx. 1 root root    6 10-14 21:55 proba3 -> proba1
-rw-r--r--. 1 root root    0 10-14 21:58 proba4
-rw-r--r--. 1 root root    0 10-14 22:00 proba6
[root@localhost test]# _
```

  Za pomocą polecenia `mv` można również zmienić nazwę pliku lub katalogu.

Zadanie 5

Do wspomnianego katalogu skopiuj kilka plików wykonywalnych.

- Polecenie `rename` także umożliwia zmianę nazwy pliku.
- Polecenie `ls` wyświetla zawartość katalogu. Kiedy jest wywołane be argumentów, `ls` wyświetla pliki z katalogu bieżącego. Jeśli nie podamy opcji, `ls` wyświetla informacje w uproszczonym formacie.

  Najczęściej stosowane opcje polecenia `ls`:

- `-l` długi format, wyświetlanie typów plików Unixa, uprawnień, liczby twardych

```
[root@localhost ~]# ls -l
razem 64
-rw-------. 1 root root 1106 2014-10-08  anaconda-ks.cfg
-rw-r--r--. 1 root root 4581 2014-11-15  file.cap
-rw-r--r--. 1 root root 9635 2014-10-08  install.log
-rw-r--r--. 1 root root 3091 2014-10-08  install.log.syslog
drwxr-xr-x. 2 root root 4096 10-14 21:05 jacek
-rw-r--r--. 1 root root  890 10-10 20:51 log.20151011
-rw-r--r--. 1 root root 1422 2014-11-16  plik.cap
-rw-r--r--. 1 root root 5106 2014-10-17  procroot.txt
-rw-r--r--. 1 root root   69 2014-10-17  prosty
-rwxr-xr-x. 1 root root   62 2014-10-17  prosty1
-rw-r--r--. 1 root root    0 2014-10-16  root.txt
drwxr-xr-x. 2 root root 4096 2014-10-09  test
-rw-r--r--. 1 root root  654 10-14 19:23 typescript
[root@localhost ~]# _
```

  Info: uprawnienia, liczba dowiązań lub liczba podkatalogów, właściciel, grupa, rozmiar, data ostatniej modyfikacji, nazwa katalogu lub pliku.

- `-f` nie sortuj, przydatny w przypadku katalogów zawierających dużą liczbę plików.
- `-F` dołącza znak ujawniający charakter pliku, na przykład`*` dla plików wykonywalnych lub `/` dla katalogów. Zwykłe pliki nie mają sufiksu.
  Do wyniku wyświetlenia dołączany jest jeden z następujących wskaźników lub brak w przypadku zwykłych plików:

*	Plik wykonywalny,
/	Katalog,
=	Socket,
@	Dowiązanie symboliczne,
|	FIFO.

Przykład:

```
[root@localhost ~]# ls -F
anaconda-ks.cfg  install.log.syslog  plik.cap      prosty1*  typescript
file.cap         jacek/              procroot.txt  root.txt
install.log      log.20151011        prosty        test/
[root@localhost ~]# _
```

- `-a` wyświetla wszystkie pliki w danym katalogu, w tym te, których nazwy zaczynają się od „.” (które są ukrytymi plikami w Uniksie). Domyślnie pliki te są wykluczone z listy.
- `-R` rekurencyjnie wyświetla podkatalogi. Komenda `ls -R /` wyświetli listę wszystkich plików.
- `-d` pokazuje informacje o dowiązaniu symbolicznym lub katalogu.
- `-t` posortuje listę plików według czasu modyfikacji.
- `-h` poda rozmiary wydruku w formacie czytelnym dla człowieka. (na przykład.,1K, 234M, 2G, etc.)

Zadanie 5

Wykonaj zrzut ekranu dokumentujący aktualną zawartość wspomnianego katalogu.

- Polecenie `ln` umożliwia utworzenie nowego dowiązania do pliku.
  Składnia `ln [opcje] źródło cel`

  `źródło` – plik lub pliki, do których tworzy się dowiązania;
  `cel` – nazwa pliku lub katalogu; jeśli cel jest nazwą pliku, stworzony zostanie link o konkretnej – podanej nazwie, jeśli zaś cel jest folderem zostaną w nim utworzone dowiązania do źródła, o nazwach identycznych z nazwami źródłowymi plików.

```
[root@localhost test]# ls -lF
razem 4
-rwxr-----. 1 root root 1220 2014-10-09  proba*
-rw-r--r--. 2 root root    0 2014-10-09  proba1
-rw-r--r--. 2 root root    0 2014-10-09  proba2
[root@localhost test]# _
```

- `ln -s` tworzy dowiązanie symboliczne:

```
[root@localhost test]# ln -s proba1 proba3
[root@localhost test]# ls-lF
-bash: ls-lF: nie znaleziono polecenia
[root@localhost test]# ls -lF
razem 4
-rwxr-----. 1 root root 1220 2014-10-09  proba*
-rw-r--r--. 2 root root    0 2014-10-09  proba1
-rw-r--r--. 2 root root    0 2014-10-09  proba2
lrwxrwxrwx. 1 root root    6 10-14 21:55 proba3 -> proba1
[root@localhost test]# _
```

  Proszę zwrócić uwagę na różnicę pomiędzy dowiązaniem twardym i symbolicznym.[5]

- Polecenie `stat`
  Służy do pobierania statusu pliku. Informacje te są przechowywane w i-węźle pliku. Działanie tej funkcji sprowadza się do przepisania zawartości i-węzła do bufora. Polecenie `stat` podaje następujące informacje: typ pliku, właściciel pliku, prawa dostępu, rozmiar pliku, liczba dowiązań, numer i-węzła oraz czasy dostępu do pliku jego modyfikacji i zmiany statusu.

```
[root@localhost test]# stat proba
  File: `proba'
  Size: 1220            Blocks: 8          IO Block: 4096   zwykły plik
Device: fd00h/64768d    Inode: 9632        Links: 1
Access: (0740/-rwxr-----)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2015-10-14 22:13:35.814007079 +0200
Modify: 2014-10-09 13:51:45.593878649 +0200
Change: 2014-10-09 14:02:59.857767957 +0200
[root@localhost test]# _
```

Zadanie 6

Utwórz dowiązanie (twarde) do wybranego pliku we wspomnianym katalogu, a następnie utwórz dowiązanie symboliczne. Zbadaj statusy tych dowiązań i przedyskutuj podobieństwa i różnice obydwu typów dowiązań.

- Polecenie `find` używane jest do lokalizacji plików o określonych parametrach:

```
[root@localhost ~]#
[root@localhost ~]# find -name proba1
./test/proba1
[root@localhost ~]# _
```

Przy poleceniu `find` warto powiedzieć o znakach specjalnych (metaznaki, wildcards) ułatwiających określenie ciągu nazw plików, które są zgodne z określonymi wzorami:

`*` zastępuje dowolną liczbę znaków, np. `ls /usr/bin/apli*`

`?` zastępuje jeden znak, np. `ls /usr/bin/plkikacja?`

nawias klamrowy `[…]` w argumencie polecenia umożliwia określenie listy znaków w mających wystąpić w nazwie, np. `ls /usr/bin/aplikacja[123]`

```
[root@localhost test]# find proba[123]
proba1
proba2
proba3
[root@localhost test]#
[root@localhost test]#
[root@localhost test]# find proba[123]
proba1
proba2
proba3
[root@localhost test]# find prob[a123]
proba
[root@localhost test]# find proba?
proba1
proba2
proba3
proba4
proba6
[root@localhost test]# _
```

Przełączniki polecenia `find` dotyczące ostatnio dokonanej operacji na pliku:

Data ostatniej modyfikacji (modification time)	`-mtime`
Data ostatniego dostępu (access time)	`-atime`
Data ostatniej zmiany statusu (change time)	`-ctime`

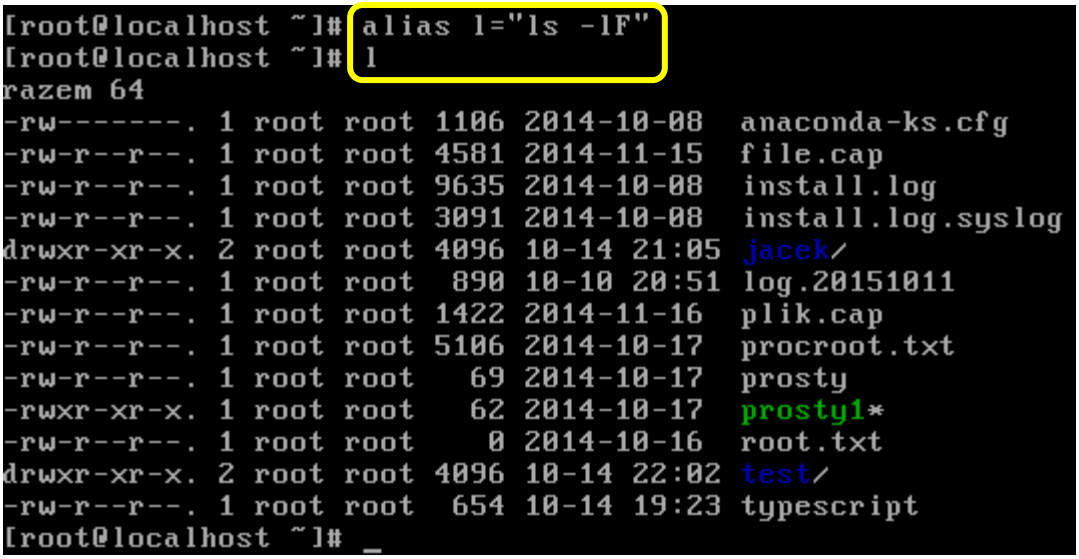
Zadanie 7

Zademonstruj działanie polecenia `find`.

Aliasy poleceń

Znaczna liczba poleceń, posiada rozbudowaną strukturę, a ponadto często zachodzi potrzeba ich wykonywania. W przypadku takich poleceń pomocnym okazuje się mechanizm tworzenia nazw skróconych, czyli aliasów.

Do definiowania nazw skróconych służy polecenie: `alias`. Jako przykład zastosowania aliasu rozważmy polecenie: `ls -lF`, gdzie poszczególne parametry oznaczają: `l` informacja o plikach, `F` wyróżnianie plików specjalnych. Możemy utworzyć nazwę skróconą:



Konieczne było zastosowanie cudzysłowu, ponieważ wewnątrz definicji aliasu występuje spacja. Wywołanie: `# l j` est równoznaczne wywołaniu wyjściowego polecenia.: `# ls -lF` Do usunięcia aliasu stosuje się polecenie: `unalias`

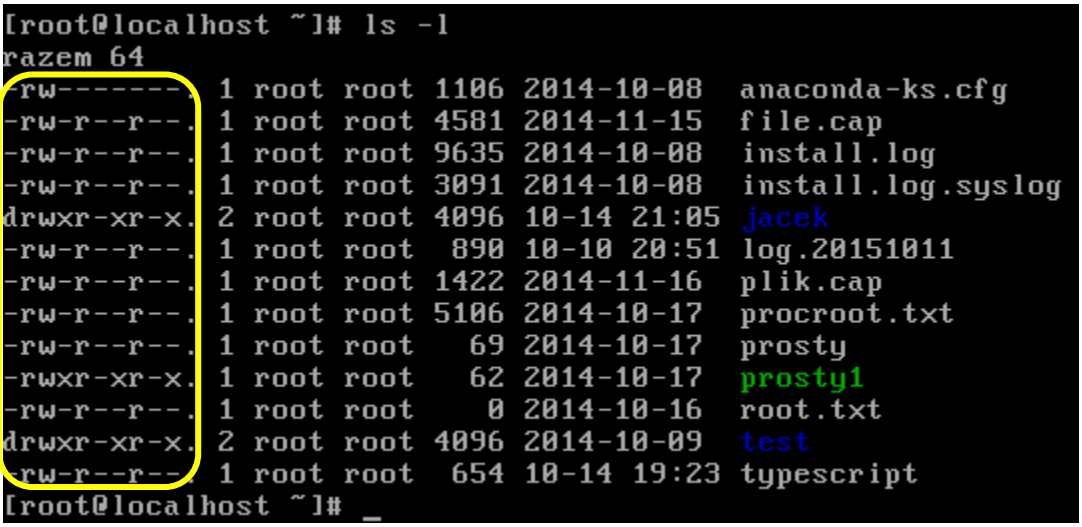
Parametryzacja aliasów.

Nazwy skrócone mogą posiadać swoje argumenty. Domyślnie lista argumentów jest podawana na prawo od wywołania aliasu, np.:

```
# l abc.txt
# ls -lF abc.txt
```

Uprawnienia (Permissions)

Każdy plik w systemie Unix ma przypisany zestaw uprawnień. Polecenie `ls -l` wyświetla informacje o uprawnieniach do pliku:



Gdzie pierwsza kolumna określa rodzaj uprawnień w stosunku do tego pliku:
‘`r`’ wskazuje, że użytkownik posiada uprawnienia do czytania (*read*) pliku.
‘`w`’ wskazuje na uprawnienia zapisywania (*write*),
‘`x`’ wskazuje na uprawnienia do uruchomienia (*execute*).

- Rodzaje uprawnień tworzą zbiór 10. znaków, o następujących znaczeniach:

Kolejny znak	Znaczenie
1	Typ pliku.
2–4	Rodzaje uprawnień do pliku dla właściciela pliku.
5–7	Rodzaje uprawnień do pliku dla członków grupy użytkowników.
8–10	Rodzaje uprawnień do pliku dla pozostałych użytkowników.

Symbole oznaczające rodzaje plików:

`-` — zwykły plik;

`b` — specjalny plik blokowy;

`c` — specjalny plik znakowy;

`d` — katalog;

`l` — dowiązanie symboliczne;

`p` — nazwany potok;

`s` — gniazdo.

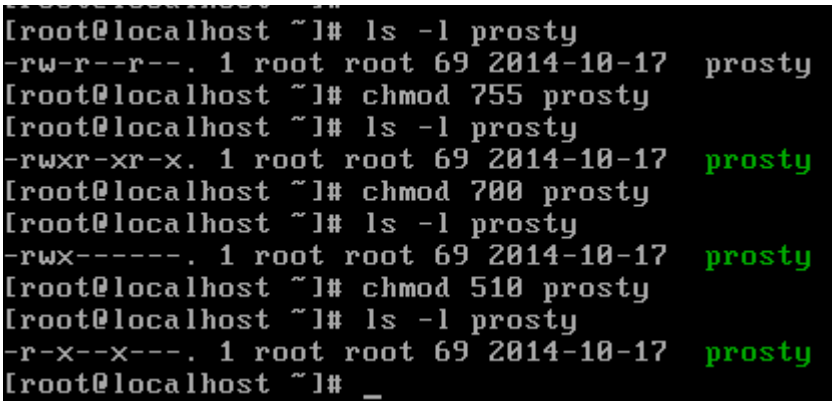
Zadanie 8

Przedyskutuj uprawnienia przypisane do kilku wybranych plików.

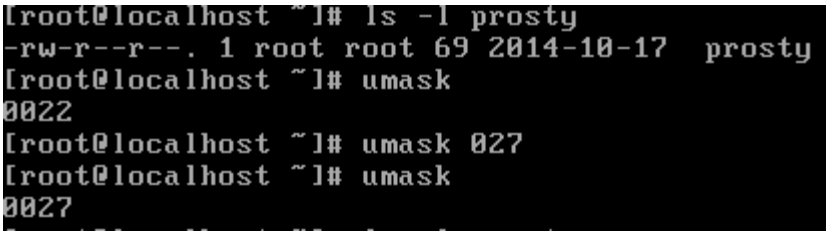
- Zmiana uprawnień
  Polecenie `chmod` może być użyte przez właściciela pliku do zmiany uprawnień dostępu. Każdemu uprawnieniu przypisana jest liczba:
  - zezwolenie na czytanie pliku = 4,
  - pisanie = 2,
  - wykonywanie = 1.

  Warto sprawdzić jak te liczby wyglądają w systemie binarnym.

  Zbiór uprawnień do zbioru dla polecenia `chmod` ma postać liczby trzycyfrowej. Pierwsza cyfra określa uprawnienia właściciela pliku. Powinna to być cyfra z zakresu od 0 do 7, która stanowi sumę uprawnień dla właściciela (patrz tabela). Druga i trzecia cyfra liczby będącej argumentem `chmod` określają uprawnienia dla członków grupy i pozostałych użytkowników, np.:



  Domyślna maska uprawnień i miana domyślnej maski uprawnień:



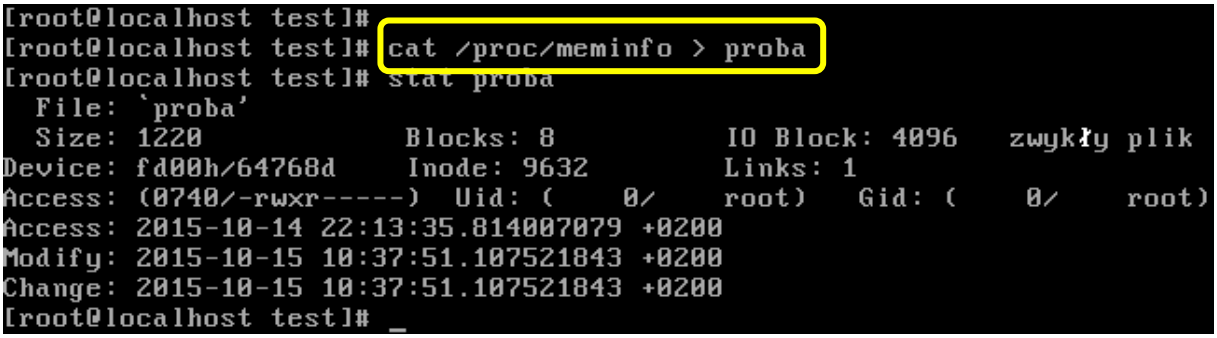
Zadanie 9

Dokonaj zmiany uprawnień kilku plików, których jesteś właścicielem. Przedyskutuj te zmiany.

Polecenia przetwarzania tekstu

Przetwarzanie tekstu jest drugim, pod względem częstości występowania, obok manipulacji plikami zadaniem stawianym przed użytkownikiem systemu Unix.

W celu prześledzenia tej grupy poleceń systemowych utwórzmy katalog `/test` i plik o nazwie `proba`, do którego wczytajmy zawartość wirtualnego pliku `/proc/meminfo`

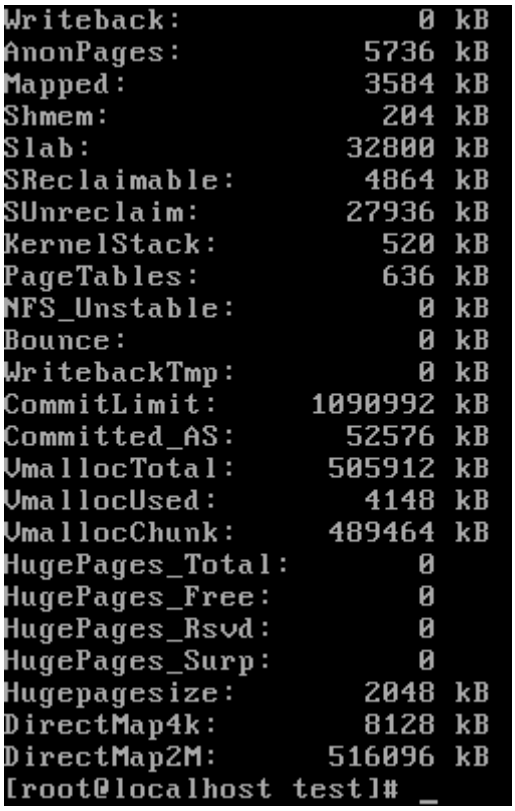


Następnie obejrzymy zawartość pliku `proba` przy użyciu różnych narzędzi.

Polecenie `cat`

Polecenie wyświetla zawartość zadanego pliku tekstowego na standardowe wyjście. Najczęściej używane opcje:

- `-A` lub `--show-all`;
- `-n` – numeruje wszystkie linie;
- `-b` lub `--number-nonblank` – numeruje niepuste linie;
- `-E` lub `--show-ends` – wyświetla znak `$` na końcu każdej linii;



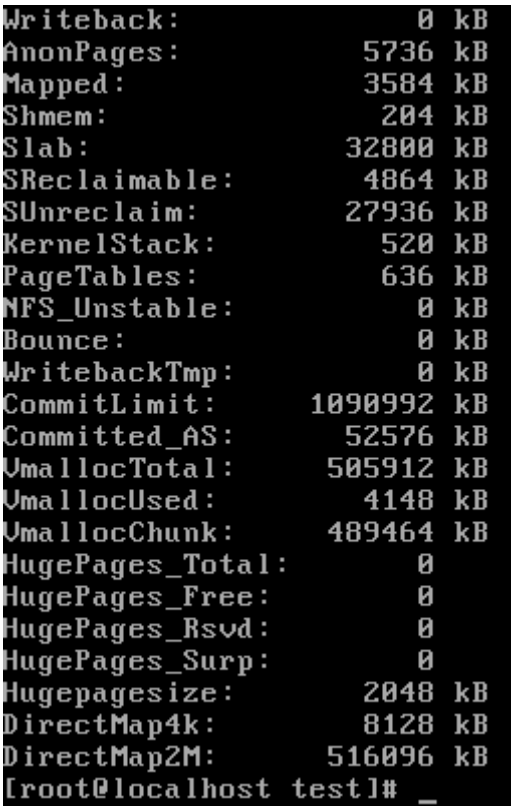
Nazwa `cat` od angielskiego *catenate* – łączyć wskazuje, ze polecenie to pozwala łączyć pliki. Opis takiego działania `cat`. [6]

Zadanie 10

Przeanalizuj przykład podany w publikacji [6] w spisie literatury i dokonaj połączenia dwu plików tekstowych przy pomocy polecenia `cat`.

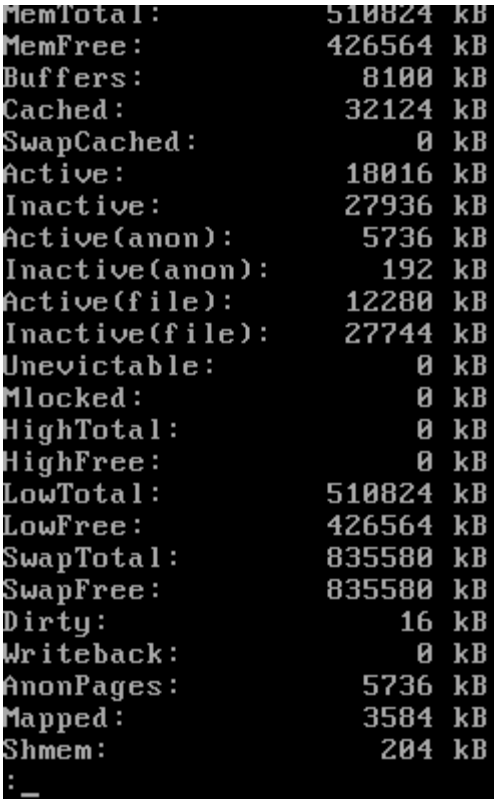
Polecenie `more`

Polecenie `more` wypisuje zawartość pliku dzieląc go na porcje odpowiadające rozmiarom ekranu. Liczba linii ekranu znajdowana jest w bazie dostępnych terminali. Jeśli nie jest to możliwe system przyjmuje, że terminal ma 24 linie.



Każdy wypisany pełny ekran kończony jest linią z następującym tekstem: --More-- oraz informacją, ile procent tekstu już wypisano. Nawigacja w dół za pomocą ENTER

Polecenie `less`:



`less` nie wczytuje całego pliku przy starcie, dzięki czemu szybciej wczytuje duże pliki.

W odróżnieniu od `more` zezwala na nawigację po pliku w obu kierunkach w dowolnym momencie za pomocą klawiszy strzałek, Page Down, Page Up, End i Home. Zakończenie działania `less`: q+ENTER.

Filtry `head` i `tail`

```
[root@localhost test]#
[root@localhost test]# head -4 proba
MemTotal:         510824 kB
MemFree:          426564 kB
Buffers:            8100 kB
Cached:            32124 kB
[root@localhost test]# tail -4 proba
HugePages_Surp:        0
Hugepagesize:       2048 kB
DirectMap4k:        8128 kB
DirectMap2M:      516096 kB
[root@localhost test]# _
```

Zadanie 11

Obejrzyj zawartość pliku `proba` przy użyciu różnych narzędzi: `cat`, `more`, `less` oraz filtrów `head` i `tail`.

Przedstawienie pliku tekstowego w formacie szesnastkowym.
`hexdump [nazwa pliku]`

```
[root@localhost ~]# hexdump prosty1
0000000 2123 622f 6e69 732f 0a68 6365 6f68 2220
0000010 3024 0a22 6365 6f68 2220 2324 0a22 6365
0000020 6f68 2420 0a31 6365 6f68 2420 0a32 6365
0000030 6f68 2420 0a33 6365 6f68 2420 0a24
000003e
[root@localhost ~]# _
```

Prefiks `0x` oznacza notację heksadecymalną – *hex dump*. – zrzut szesnastkowy.

W zrzucie szesnastkowym każdy bajt (8 bitów) jest reprezentowany jako dwucyfrowa liczba szesnastkowa. Format ten ma na celu pomieszczenie maksymalnej ilości danych na standardowym ekranie o szerokości 80 znaków.

- Dwubajtowe liczby są oddzielone białymi znakami i zorganizowane w wiersze po 16 bajtów.
- Pierwsza kolumna po lewej stronie określa szesnastkowe przesunięcie całego wiersza.
- Ostatni wiersz wyświetla całkowitą liczbę przetworzonych bajtów.

W systemie szesnastkowym wyróżniamy szesnaście cyfr:
`0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.`

Potok (ciąg poleceń, pipe)

Programy mogą działać jednocześnie bez konieczności pośredniczenia poprzez dodatkowy plik. Jednym ze sposobów realizacji takiego przetwarzania jest potok, który tworzy się poprzez oddzielenie poszczególnych poleceń znakiem potoku `'|'` (pipe).

```
[root@localhost test]#
[root@localhost test]#
[root@localhost test]# last root | less_
```

Potok bierze rezultat działania polecenia znajdującego się po jego lewej stronie jako wejście dla polecenia znajdującego się po prawej jego stronie.

```
root     tty1                          Thu Nov  6 11:24 - crash (1+05:29)
root     tty1                          Wed Nov  5 21:58 - crash  (13:25)
root     tty1                          Wed Nov  5 15:58 - crash  (05:57)
root     tty1                          Wed Nov  5 13:14 - crash  (02:39)
root     tty1                          Mon Nov  3 22:38 - crash (1+14:35)
:_
```

Filtr `sort`

Polecenie umożliwia sortowanie w kolejności alfabetycznej,

```
[root@localhost test]#
[root@localhost test]#
[root@localhost test]# cat proba | sort_
```

```
Inactive(anon):      192 kB
Inactive(file):    27744 kB
KernelStack:         520 kB
LowFree:          426564 kB
LowTotal:         510824 kB
Mapped:             3584 kB
MemFree:          426564 kB
MemTotal:         510824 kB
Mlocked:               0 kB
NFS_Unstable:          0 kB
PageTables:          636 kB
```

opcja `-r` pozwala odwrócić porządek sortowania.

Filtr `grep`

Polecenie `grep` wyświetla wszystkie linie pliku, które zawierają określony łańcuch znaków, np.:

```
[root@localhost test]# grep ir proba
Dirty:                16 kB
DirectMap4k:        8128 kB
DirectMap2M:      516096 kB
[root@localhost test]# _
```

Filtr `wc`

Narzędzie do liczenia słów, znaków, linii lub bajtów w pliku lub potoku. Opcje:

`-l` zliczana jest liczba wierszy,

`-w` zliczana jest liczba słów,

`-c` zliczana jest liczba znaków.

Opcje filtra mogą być stosowane łącznie: `-lwc`.

```
[root@localhost test]# cat proba | wc -l
44
[root@localhost test]# wc -l proba
44 proba
[root@localhost test]# _
```

Zadanie 12

Zademonstruj działanie filtrów `sort`, `grep` i `wc` na wybranym pliku/plikach tekstowych wykorzystując potokowe łączenie poleceń.

Przekierowania

Deskryptory strumieni związanych z procesami:
0 – standardowe wejście – `stdin`,
1 – standardowe wyjście – `stdout`,
2 – wyjście błędów – `stderr`,
Za pomocą przekierowania (redirection) „.\>,<”związane z procesami strumienie można przekierować do plików. Objaśnienie: [7]
`polecenie > plik` zapisuje `stdout` polecenia od początku pliku `plik`, więc plik zostanie nadpisany;
`polecenie >> plik` dopisuje `stdout` na końcu pliku `plik;`
`polecenie < plik` otwiera zawartość `plik` jako `stdin` polecenia;

Przykłady zastosowań:
Przekierowanie zawartości pliku `proba` do polecenia `cat`:

```
[root@localhost test]#
[root@localhost test]# cat < proba_
```

Przekierowanie wyjścia zostanie użyte na nieistniejący plik to taki plik zostanie utworzony:

```
[root@localhost test]#
[root@localhost test]# cat proba > archive_
```

Polecenie `tee`
`Przykład działania polecenia tee,` które wyświetla wynik na ekranie (`stdout`) i umieszcza go w pliku:

```
[root@localhost test]#
[root@localhost test]#
[root@localhost test]# sort < proba | tee proba.sorted_
```

Zadanie 13

Zademonstruj działanie przekierowań standardowego wejścia i wyjścia.

Literatura:

---

[1] Cezary Sobaniec, *System operacyjny Linux – przewodnik użytkownika*, Nakom, Poznań, 2002.

[2] Zenon Grodzki, Zbigniew Kacprzyk, Andrzej Kurowski, Maciej Winiarski, *Łagodne wprowadzenie do systemu Unix*, Politechnika Warszawska, Warszawa, 1991.

[3] http://home.agh.edu.pl/~remigius/Materialy/Unix/Polecenia.pdf

[4] http://kik.pcz.pl/so-add/KSL/kompendium.html

[5] https://pl.wikipedia.org/wiki/Ln_(Unix)#Dowi%C4%85zania_symboliczne

[6] http://linuxwiki.pl/wiki/Cat

[7] http://www.cs.put.poznan.pl/anstroinski/data/uploads/sop1/materials/sop1_lab5-wyklad-filtry-i-potoki.html