

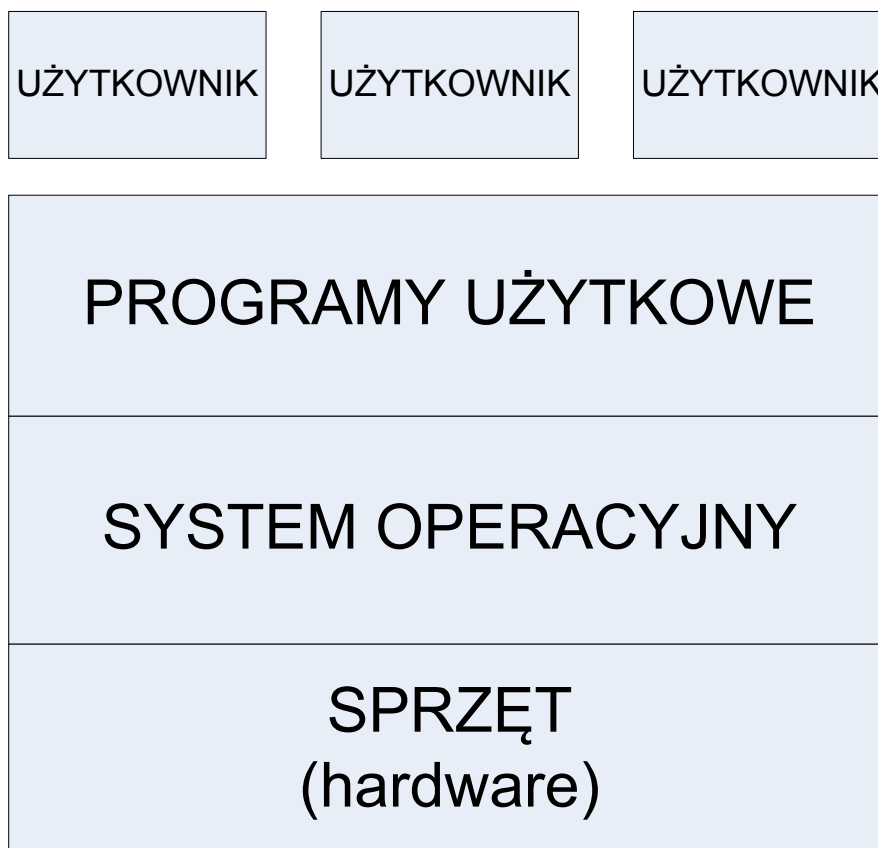
Systemy Operacyjne

Unix, Linux

1. Pojęcie systemu operacyjnego

System operacyjny jest programem, który działa jako pośrednik pomiędzy użytkownikiem komputera, a sprzętem komputerowym.

Podstawowym zadaniem systemu operacyjnego jest tworzenie środowiska, w którym użytkownik może wykonywać programy oraz sprawić aby system komputerowy był wygodny w użyciu.



Rys. 1 Sprzęt, system operacyjny, programy użytkowe, użytkownicy.

Czym jest system operacyjny?

- środowisko uruchamiania programów
- dystrybutor zasobów (czas procesora, pamięć)
- program sterujący – programami użytkownika i systemowymi, oraz zarządzający dostępem do urządzeń wejścia-wyjścia.

Co jest systemem operacyjnym? Tylko jądro (ang. kernel) działające w systemie komputerowym nieustannie, czy wszystko to do dostarcza producent systemu operacyjnego?

Historyczne spojrzenie na rodzaje systemów operacyjnych

- wsadowe
- wsadowe wieloprogramowe
- wielozadaniowe z podziałem czasu
- wielozadaniowe równoległe (wieloprocessorowe)
- rozproszone
 - podział zasobów
 - przyspieszenie obliczeń
 - niezawodność
 - komunikacja
- czasu rzeczywistego

2. Podstawowe zadania systemu operacyjnego

2.1. Zarządzanie procesami

- tworzenie i usuwanie procesów użytkowych i systemowych
- wstrzymywanie i wznowianie procesów
- dostarczanie mechanizmów synchronizacji procesów
- dostarczanie mechanizmów komunikacji procesów
- dostarczanie mechanizmów obsługi zakleszczeń

2.2. Zarządzanie pamięcią operacyjną

- ewidencjonowanie zajętych części pamięci z informacją o właścicielu
- przydzielanie i zwalnianie obszarów pamięci
- decydowanie o tym, które procesy mają być załadowane do zwalnianych obszarów pamięci

2.3. Zarządzanie plikami

- tworzenie i usuwanie plików
- tworzenie i usuwanie katalogów
- dostarczanie elementarnych operacji do manipulowania plikami i katalogami
- odwzorowanie plików na obszary pamięci pomocniczej
- składowanie plików na trwałych nośnikach

2.4. Zarządzanie systemem wejścia-wyjścia

W SO UNIX podsystem wejścia-wyjścia ukrywający przed systemem operacyjnym osobliwości urządzeń we-wy. Podsystem zawiera:

- moduły sterujące urządzeniami sprzętowymi
- interfejs modułów sterujących
- część zarządzającą pamięcią w tym: buforowanie, spooling i pamięć podręczna

2.5. Zarządzanie pamięcią pomocniczą

- zarządzanie wolnymi obszarami
- przydzielanie pamięci
- planowanie przydziału obszarów pamięci dyskowej

2.6. Praca sieciowa

- współpraca procesów rozproszonych, które nie dzielą wspólnej pamięci, procesora ani zegara.

2.7. System ochrony

- ochrona plików, pamięci, procesora i innych zasobów

2.8. System interpretacji poleceń

- interpretacja poleceń wydawanych przez użytkownika - shell

3. Usługi systemu operacyjnego

- wykonanie programu
- operacje wejścia-wyjścia
- manipulowanie systemem plików
- komunikacja
- wykrywanie błędów
- przydzielanie zasobów
- rozliczanie
- ochrona

3.1. Funkcje systemowe

Funkcje systemowe albo wywołania systemowe (ang. system calls) tworzą interfejs pomiędzy wykonywanym programem a systemem operacyjnym.

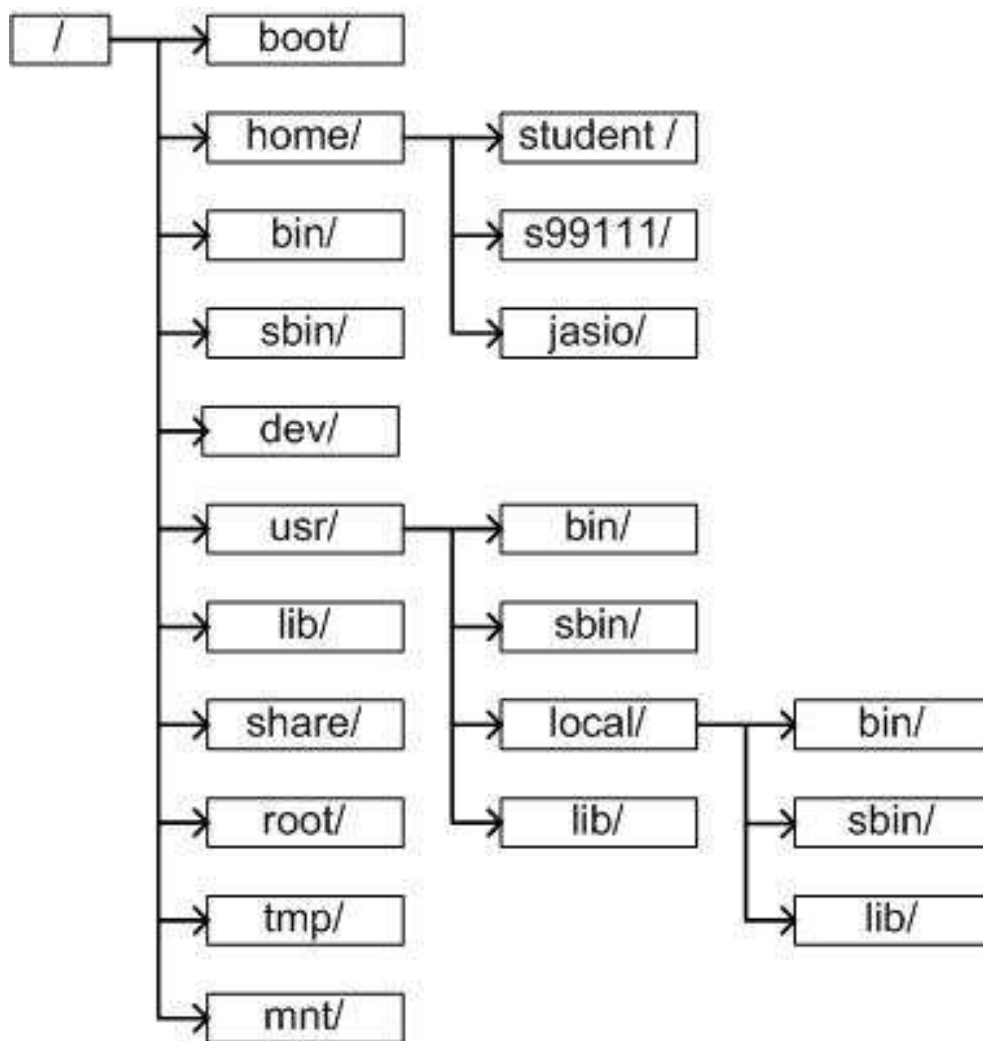
- Nadzorowanie procesów
 - Zakończenie, zaniechanie
 - Załadowanie, wykonanie
 - Utworzenie procesu, zakończenie procesu
 - Pobranie atrybutów procesu, określenie atrybutów procesu
 - Czekanie czasowe
 - Oczekiwanie na zdarzenie, sygnalizacja zdarzenia
 - Przydział i zwolnienie pamięci
- Operacje na plikach
 - Utworzenie pliku, usunięcie pliku
 - Otwarcie, zamknięcie
 - Czytanie, pisanie, zmiana położenia
 - Pobranie atrybutów pliku, określenie atrybutów pliku
- Operacje na urządzeniach
 - Zamówienie urządzenia, zwolnienie urządzenia
 - Czytanie, pisanie, zmiana położenia
 - Pobranie atrybutów urządzenia, określenie atrybutów urządzenia
 - Logiczne przyłączania lub odłączanie urządzeń
- Utrzymywanie informacji
 - Pobranie czasu lub daty, określenie czasu lub daty
 - Pobranie danych systemowych, określenie danych systemowych
 - Pobranie atrybutów procesu, pliku lub urządzenia
- Komunikacja
 - Utworzenie, usunięcie połączenia komunikacyjnego
 - Nadawanie, odbieranie komunikatów
 - Przekazanie informacji o stanie
 - Przyłączanie lub odłączanie urządzeń zdalnych

3.2. Programy systemowe

- Manipulowanie plikami
- Informowanie o stanie systemu
- Tworzenie i zmienianie zawartości plików
- Translatory języków programowania
- Ładowanie i wykonywanie programów
- Komunikacja

4. System operacyjny Unix

- System wieloprocesowy z podziałem czasu
- System wielodostępowy
- System napisany przez programistów dla programistów
- System napisany według zasady proste jest lepsze
- System z hierarchiczną strukturą katalogów



4.1. Atrybuty plików i katalogów

Polecenie **ls** wypisuje nazwy plików i katalogów.

Polecenie **ls -l** wypisuje atrybuty plików w bieżącym katalogu

Polecenie **ls -la** wypisuje nazwy plików łącznie z ukrytymi

Jako nazwy ukryte w systemie Linux traktowane są wszystkie nazwy rozpoczynające się od kropki.

Przykład:

```
drwxr-xr-x   5 root root  4096 kwi 26  2005 mnt
-rw-r--r--   1 root root 10448 gru 30  2004 nsh5.1.tar.bz2
drwxr-xr-x   2 root root  4096 sie 12  2004 opt
```

Typ pliku:

- (f) - zwykły plik
- d - katalog
- l - dowiązanie symboliczne (link)
- c - specjalny plik znakowy
- b - specjalny plik blokowy
- s - gniazdo (semafor)
- p - łącze nazwane FIFO

Atrybuty:

- r - odczyt
- w - zapis
- x - wykonanie albo dostęp do katalogu

Uprawnieni:

- u - user - właściciel
- g - group - grupa
- o - others - inni
- a - all - wszyscy

Inne: Liczba dowiązań, Właściciel, Grupa, Rozmiar, Czas modyfikacji,
Nazwa

4.2. Zmiana uprawnień do plików i katalogów

Polecenie **chmod** [ugoa][+--=][rwx] nazwa

gdzie:

- + oznacza nadanie uprawnienia
- oznacza odebranie uprawnienia
- = oznacza nadanie uprawnienia i odebranie wszystkich innych

Uprawnienia można też podawać numerycznie (r=4,w=2,x=1)

chmod 750 nazwa

co jest równoważne poleceniu:

chmod u+rwx,g=rw,o-rwx nazwa

4.3. Znaczenie atrybutów w odniesieniu do plików i katalogów

Dla katalogów:

- r – możliwość odczytu zawartości katalogu (ls)
- w – możliwość modyfikacji zawartości katalogu (tworzenie i usuwanie plików (rm, mv, itp.)
- x – możliwość „wejścia” do katalogu (cd)

Dla plików:

- r – możliwość odczytu zawartości pliku (np. cat)
- w – możliwość modyfikowania zawartości pliku (np. vi)
- x – możliwość uruchamiania pliku (plik jest programem)

4.4. Modyfikowanie właściciela pliku

Zmiana właściciela

chown nowy_właściciel plik

zmiana grupy

chgrp nowa_grupa plik

albo równoczesna zmiana właściciela i grupy

chown nowy_właściciel.nowagrupa plik

4.5. Przeszukiwanie systemu plików

which nazwa

podaje pełną ścieżkę dostępu do programu nazwa

find ~/ -name 'skrypt*'

poszukuje, począwszy od katalogu domowego, plików, których nazwa rozpoczyna się od frazy skrypt

find ~/ -mtime -1

poszukuje, począwszy od katalogu domowego, plików, których czas modyfikacji jest nie większy niż 1 dzień

find ~/ -mtime +15

poszukuje, począwszy od katalogu domowego, plików, których ostatnia modyfikacja miała miejsce dawniej niż 15 dni temu

find ~/ -size +1M

poszukuje, począwszy od katalogu domowego, plików, których rozmiar jest większy od 1MB

find ~/ -size -100c

poszukuje, począwszy od katalogu domowego, plików, których rozmiar nie jest większy od 100 znaków

find ~/ -type f -exec ls -l {} /;

poszukuje wszystkich plików zwykłych i wypisuje dla nich atrybuty poleceniem ls -l

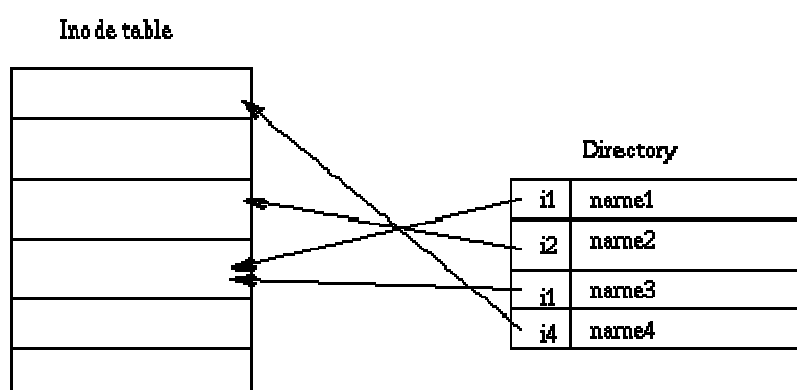
4.6. Budowa systemu plików

Macierzystym systemem plików Linuxa jest system ext2. Ostatnio stosowany system ext3 ma identyczną budowę, ale dodatkowo wprowadza „księgowanie” zmian wprowadzanych na dysku co umożliwia szybką naprawę systemu plików w przypadku np. awarii zasilania, i gwarantuje wysoki stopień bezpieczeństwa spójności zapisywanych danych.

Logiczna budowa systemu plików ext2:

Superblok	Deskryptory grup	Bitmapa bloków	Bitmapa i-węzłów	Tablica i-węzłów	Bloki danych
-----------	------------------	----------------	------------------	------------------	--------------

Chcąc odczytać dane z pliku system najpierw musi odczytać i-węzeł pliku, a dopiero na podstawie jego zawartości odszukiwane są bloki z danymi pliku.



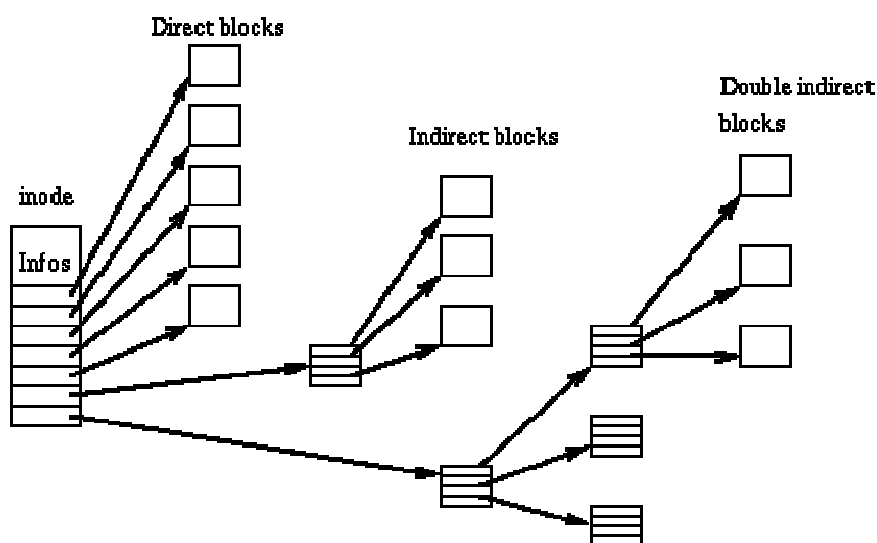
Każdy wpis w katalogu rozpoczyna się od numeru i-węzła danego pliku.

Struktura wpisu w katalogu:

Numer i-węzła	Rozmiar wpisu	Długość nazwy	Nazwa pliku
---------------	---------------	---------------	-------------

Wpis w katalogu o zmiennej długości pozwala na łatwą realizację długich nazw plików bez marnowania miejsca w przypadku gdy nazwy plików są krótkie.

Do przechowywania adresów bloków z danymi w i-węźle wykorzystywana jest 15 elementowa tablica. 12-cie pierwszych elementów tej tablicy wskazuje na 12-cie pierwszych bloków danych pliku, 13-ty element jest wykorzystywany do adresowania pośredniego, 14-ty do adresowania podwójnie pośredniego, a 15-ty do adresowania potrójnie pośredniego.



Przedstawiony sposób adresowania bloków danych pozwala z jednej strony zwiększyć wydajność obsługi małych plików (do 12 bloków), z a drugiej umożliwia przechowywanie bardzo dużych plików.

Dla przykładu: dla rozmiaru bloku wynoszącego 4096 bajtów największy plik adresowany bezpośrednio to $12 \cdot 4 = \mathbf{48KB}$,
 adresowany pośrednio to $12 \cdot 4KB + 4096/4 \cdot 4KB = \mathbf{4MB} + 48KB$,
 adresowany podwójnie pośrednio to
 $12 \cdot 4KB + 4096/4 \cdot 4KB + 4096/4 \cdot 4096/4 \cdot 4KB = \mathbf{4GB} + 4MB + 48KB$,
 a adresowany potrójnie pośrednio to ponad **4TB**.

4.7 Tworzenie i utrzymanie systemu plików.

4.7.1. Nazwy dysków

/dev/cdrom - domyślny napęd optyczny
/dev/fd0 - pierwszy napęd dyskietek
/dev/hda - master w pierwszym kontrolerze IDE
/dev/hdb - slave w pierwszym kontrolerze IDE
/dev/hdc - master w drugim kontrolerze IDE
/dev/sda - pierwszy dysk typu SCSI (SATA, USB)
/dev/sdb - drugi dysk typu SCSI (SATA, USB)

4.7.2. Nazwy partycji

/dev/hda1 - pierwsza partycja na pierwszym dysku IDE
/dev/sdb4 - czwarta partycja na drugim dysku SCSI
/dev/sdb5 - pierwsza partycja rozszerzona (5) na drugim dysku SCSI

4.7.3. Tworzenie partycji

fdisk /dev/hda

- polecenia

m – lista poleceń
p – lista partycji
n – utwórz nową partycję
w – zapisz zmiany i zakończ

4.7.4. Tworzenie systemu plików (formatowanie partycji)

- partycja wymiany

mkswap /dev/hda1

- partycja z systemem plików ext2

mkfs /dev/hda2

- partycja z systemem plików ext3 (z transakcyjnym systemem plików)

mkfs -j /dev/hda3

4.7.5. Podłączanie partycji wymiany do pamięci wirtualnej

swapon /dev/hda1

4.7.6. Montowanie systemu plików w strukturze katalogów

mount /dev/hda2 /folder

4.7.7. Odmontowywanie partycji

swapoff /dev/hda1

umount /dev/hda2

albo

umount /folder

4.7.8. Sprawdzanie spójności systemu plików na partycji

(system plików powinien być odmontowany)

fsck /dev/hda2

5. Wprowadzenie do wyrażeń regularnych

Wyrażenie regularne (ang. *regular expression*) jest zbiorem uogólnionych reguł opisujących napis (ciąg znaków). Jeżeli pewien napis spełnia reguły wyrażenia to mówimy, że *pasuje* do tego wyrażenia.

Elementy budowy wyrażeń regularnych:

Atomy (ang. *atoms*)

- . - kropka
- dowolny pojedynczy znak alfanumeryczny
- [] - lista znaków ujętych w nawiasy kwadratowe
- pojedynczy znak z listy w nawiasach kwadratowych
- () - wyrażenie regularne ujęte w nawiasy okrągłe
- wszystko co pasuje do wyrażenia w nawiasie
- znak - znak alfanumeryczny
- dokładnie ten znak

Zakotwiczenia (ang. *assertions*)

- ^ - początek tekstu
- \$ - koniec tekstu

Kwantyfikatory – powtórzenia

- * - zero lub więcej wystąpień atomu
- + - jedno lub więcej wystąpień atomu
- ? - zero lub jedno wystąpienie atomu
- {2} - dokładnie dwa wystąpienia atomu
- {2,} - przynajmniej dwa wystąpienia atomu
- {2,5} - od dwóch do pięciu wystąpień atomu

Znaki specjalne poprzedzamy symbolem \ (*backslash*)

\< - spacje na początek wyrazu

\> - spacje na końcu wyrazu

Przykłady:

al

- każdy tekst zawierający parę liter al. np. kalka, albo, malwa

^ko

- każdy tekst rozpoczynający się od znaków ko. np. korba, kok

\<to\>

- wyraz to.

^[0-9]{7}\$

- napis składający się z dokładnie siedmiu cyfr. np. 3456789

^[0-9]{2}-[0-9]{3}\$

- napis odpowiadający polskiemu kodowi pocztowemu.

5.1. Symbole wieloznaczne w nazwach plików

? – zastępuje jeden znak na danej pozycji

* - zastępuje dowolną liczbę znaków

[znaki] – na danej pozycji może wystąpić tylko znak z listy w nawiasach kwadratowych

[!znaki] – na danej pozycji nie może wystąpić znak z listy rozpoczynającej się od symbolu !

()|() – wyrażenia ujęte w nawiasy okrągłe i połączone symbolem | oznaczają logiczną alternatywę

6. Potoki i przekierowania

6.1. Przekierowania

Przekierowania umożliwiają zmianę domyślnego wyjścia lub wejścia programów. Pozwalają np. zapisywać wyniki działania poleceń lub przygotowywać zestawy danych wejściowych dla programów.

Przykłady:

Przekierowanie wyjścia do nowego pliku

```
ls >plik
```

Przekierowanie wyjścia na koniec pliku

```
ls >>plik
```

Przekierowanie wejścia z pliku

```
cat <plik
```

Przekierowanie równocześnie wejścia i wyjścia

```
cat <plik1 >plik2
```

Przekierowanie komunikatów o błędach do urządzenia /dev/null

```
ls -R / 2>/dev/null
```

Przekierowanie komunikatów o błędach do standardowego wyjścia

```
ls -R / 2>&1
```

6.2. Potoki

W przetwarzaniu potokowym możemy uczynić standardowe wyjście jednego programu wejściem kolejnego, rozdzielając odpowiednie polecenia symbolem |.

Przykłady:

```
cat plik | more
ls -l / | sort -n k5,5
ls -l / | sort -r k9
cat /etc/passwd | sort -t: -n k3,3 | more
cat /etc/passwd | grep Anna | sort | more
```

6.3. Polecenia przydatne do przetwarzania potokowego

more – polecenie zatrzymuje wyświetlanie po wypełnieniu każdego pełnego ekranu, kontynuacja klawiszem spacji, zakończenie klawiszem q.

less – polecenie podobne do more dodatkowo umożliwia przewijanie linia po linii w górę i w dół oraz przeszukiwanie zawartości wyświetlanego tekstu (/ , ?).

sort – sortuje zawartość pliku

- r odwrotny kierunek sortowania

- n porównuj wartości numeryczne

- t<znak> traktuj znak jako separator pól

- K<n> rozpocznaj porównywanie od n pola

- k<n>,<m> porównywanie pól od n do m

np. `sort k=3,3 plik`

- wypisuje na ekranie zawartość pliku posortowaną według 3-go wyrazu w kolejnych liniach

grep – wyszukuje linii tekstu pasujących do wyrażenia

- v wyszukuj niepasujące

- c tylko zliczaj liczbę linii

- n wypisuj numery linii

- i ignoruj różnice dużych małych liter

np. `grep -c napis plik`

- wypisuje liczbę linii w pliku zawierających napis

cut - wycina z linii pliku wybrane pola

- d<znak> traktuj znak jako separator pól

- f<n> wybierz tylko pole n

- f<n>,<m> wybierz pola n i m

-f<n>-<m> wybierz pola od n do m

-c<n>- wybierz od n tego znaku linii

np. `cut -d: -f5 /etc/passwd`

-wypisuje tylko 5-te pole z linii pliku traktując jako separator pól :

sed - program do przetwarzania linii tekstu

np. `sed s/wzor/nowy/g plik`

- wyszukuje w kolejnych liniach pliku napisów pasujących do wyrażenia regularnego wzor i wymienia je na napis nowy

awk - język programowania program do przetwarzania tekstów

-F<znak> traktuj znak jako separator pól

BEGIN blok przetwarzania przed analizą linii

END blok przetwarzania po analizie linii

Dostępne polecenia: print, if, for, operacje arytmetyczne itp.

Zgodnie ze składnią języka C

np. `awk -F: '{ print $1 }' /etc/passwd`

- wypisuje pierwszy wyraz z każdej linii pliku /etc/passwd, traktując jako separator wyrazów : (dwukropek)

np. `awk -F: '{if($4==501) print $1" "$5}' /etc/passwd`

- wypisuje z pliku /etc/passwd pierwszy i piąty wyraz pod warunkiem, że czwarty wyraz to 501

tee - rozdzielenie standardowego wyjścia równocześnie na ekran i do pliku

`ls` - wyjście tylko na ekran

`ls >plik` - wyjście tylko do pliku

`ls | tee plik` - wyjście na ekran i do pliku

7. Archiwizacja i kompresja danych

tar - utworzenie jednoplikowego archiwum z wielu plików i katalogów oraz wyodrębnianie plików z archiwum

c - tworzy archiwum

u - aktualizuje archiwum

x - wyodrębnia pliki z archiwum

f <nazwa> zapisuje/odczytuje do/z pliku

v - wypisuje nazwy przetwarzanych plików na ekranie

z - stosuje kompresję gzip

j - stosuje kompresję bzip2

gzip - bezstratna kompresja zawartości pliku

-1 - najszybsza kompresja

-9 - najlepsza kompresja

gunzip - dekompresja zawartości pliku

bzip2 - jak gzip inny, nowszy algorytm kompresji

bunzip2 - jak gunzip inny, nowszy algorytm kompresji

zip - popularny, występujący w wielu systemach operacyjnych, program do kompresji danych umożliwiający kompresję wielu plików i katalogów w jednym archiwum

unzip - dekompresor programu zip

Przykłady:

Utworzenie archiwum z wszystkich plików i podkatalogów katalogu

/home/s99111

```
tar -cf archiwum.tar /home/s99111/
```

jw. ale dodatkowo z kompresją

```
tar -c /home/s99111/ | gzip > archiwum.tgz
```

```
tar -czf archiwum.tgz /home/s99111/
```

```
tar -c /home/s99111/ | bzip2 > archiwum.tar.bz2
```

rozpakowanie skompresowanego archiwum

```
tar -xzf archiwum.tgz
```

```
gunzip < archiwum.tar.gz | tar -x
```

```
bunzip2 < archiwum.tar.bz2 | tar -x
```

7. Edycja tekstów

7.1. Edytor vi (vim)

Przedstawione tu polecenia stanowią wybrany zbiór najczęściej stosowanych, po kompletny opis wszystkich odsyłam do podręcznika.

7.1.1. Uruchamianie

view nazwa	- tylko do odczytu pliku nazwa
vi nazwa	- otwarcie pliku do edycji
vi +23 nazwa	- otwarcie do edycji od linii numer 23
vi +/słowo	- otwarcie do edycji od pierwszego wystąpienia słowa w tekście

7.1.2. Nawigacja po tekście

h, ←	- znak w lewo
j, ↓	- linia w dół
k, ↑	- linia w górę
l, →	- linia w prawo
^	- początek linii
\$	- koniec linii
^F, PgDn	- strona w dół
^B, PgUp	- strona w górę
<nr>G	- do linii numer
<nr>	- do kolumny numer
w, W	- do następnego słowa – duża litera to słowo łącznie z innymi znakami
b, B	- do poprzedniego słowa – duża litera jw.
e, E	- do końca słowa – duża litera jw.
(,)	- do początku/końca zdania – zdanie kończy się kropką i dwoma spacjami albo znakiem końca linii
%	- do pasującego nawiasu

7.1.3. Przejście do trybu edycji

i	- wstawianie znaków przed kursorem
I	- wstawianie znaków przed bieżącą linią
a	- wstawianie znaków za kursorem
A	- wstawianie znaków za bieżącą linią

o	- wstawianie do nowej linii poniżej bieżącej
O	- wstawianie do nowej linii powyżej bieżącej
r	- nadpisywanie od kursora
R	- nadpisywanie całej linii

7.1.4. Przejście do trybu poleceń

ESC

7.1.5. Polecenia

Zastępowanie i usuwanie

s	- zastąpienie pojedynczego znaku
S	- zastąpienie całej linii
x	- usunięcie pojedynczego znaku
d kursor	- usunięcie zgodnie z kierunkiem kursora
dd	- usunięcie całej linii

Poszukiwanie

/wzór	- poszukiwanie wzoru w kierunku końca tekstu
/	- powtórne poszukiwanie
?wzór	- poszukiwanie wzoru w kierunku początku tekstu
?	- powtórne poszukiwanie

Wzór jest traktowany jak wyrażenie regularne.

Poszukiwanie i zamiana

:g/szu/s//zam/
 - wyszukaj pierwszego wystąpienia szu w kolejnych liniach i
 zamień na zam

:g/szu/s//zam/g
 - wyszukaj każdego wystąpienia szu w kolejnych liniach i
 zamień na zam

:g/szu/s//zam/gc
 - wyszukaj każdego wystąpienia szu w kolejnych liniach i
 zamień na zam po potwierdzeniu przez użytkownika

:g/wzor/s/szu/zam/gc

- wyszukaj każdego wystąpienia szu w liniach pasujących do wzoru i zamień na zam po potwierdzeniu przez użytkownika

7.1.6. Inne polecenia

- J - połącz z kolejną linią
- u - cofnij ostatnią operację zmiany tekstu
- U - cofnij wszystkie zmiany w bieżącej linii
- ^G - wypisz aktualny status
- ZZ - zapisz zmiany i zakończ
- :q - zakończ – tylko jeśli nie było zmian
- :q! - zakończ i porzuć wprowadzone zmiany
- :w - zapisz
- :w nazwa - zapisz jako
- :x - zakończ zapisując jeśli były zmiany

Poprzedzenie polecenia liczbą powoduje wielokrotne powtórzenie polecenia np. 23dd – usuwa kolejne 23 wiersze.

7.2. Inne edytory

Do innych popularnych edytorów systemu Linux należą:

pico, nano, mcedit, emacs

Są to edytory pełnoekranowe, najczęściej z opisem obsługi wypisanym w linii stanu.

8. Skrypty powłoki

Dowolny zestaw poleceń zapisanych w pliku z nadanym atrybutem wykonywalności staje się wykonywalnym skryptem powłoki.

Przykład:

```
#!/bin/bash  
echo hello!
```

8.1. Instrukcje sterujące stosowane w przetwarzaniu wsadowym

```
if cmd; then cmd; fi  
if cmd; then cmd; else cmd; fi
```

Jeśli polecenie po if zakończy się sukcesem, to wykonane zostaje polecenia po then, w innym przypadku polecenia po else.

```
for nazwa in words; do cmd; done
```

Dla nazwy przyjmującej wartości z listy słów wykonuj polecenia po do.

```
for (( wyr ; wyr ; wyr )); do cmd; done
```

Pętla for jak w języku C.

np. `for ((i=0; $i<10; i=$((i+1)))); do echo $i; done`

```
while cmd; do cmd; done  
until cmd; do cmd; done
```

W czasie gdy (while) albo aż do (until) polecenie przed do kończy się sukcesem wykonuj polecenia po do.

```
case word in; nazwa) cmd;; nazwa[|nazwa]) cmd;; esac
```

w zależności od wartości słowa, poszukiwana jest nazwa i wykonywane polecenie po).

nazwa=wartość – przypisanie wartości do symbolu nazwa

\$nazwa – wartość pamiętana w symbolu nazwa

\$(polecenie) – wykonanie polecenia i wstawienie jego wyniku w miejscu wywołania

\$((wyrażenie)) – obliczenie wartości wyrażenia arytmetycznego.

Dopuszczalne są wszystkie operatory z języka C oraz ** jako potęgowanie.

\$# - liczba parametrów przekazanych do skryptu przy uruchomieniu

\$@ - wszystkie parametry jako jeden łańcuch znaków

\$1, \$2 .. \$9 - wartość pierwszego, drugiego itd. parametru przekazanego do skryptu

\$0 – nazwa skryptu

\$\$ - PID powłoki

\$? – kod zakończenia ostatniego potoku

#! – PID procesu ostatnio uruchomionego do pracy w tle

8.2. Wyrażenia warunkowe

- [-d plik] – plik istnieje i jest katalogiem
- [-f plik] – plik istnieje i jest zwykłym plikiem
- [-s plik] – plik istnieje i ma rozmiar większy od zera
- [-r plik] – plik istnieje i możemy go czytać
- [-w plik] – plik istnieje i możemy go modyfikować
- [-x plik] – plik istnieje i możemy go uruchamiać
- [plik1 -nt plik2] – plik1 jest nowszy od pliku2
- [plik1 -ot plik2] – plik1 jest starszy od pliku2
- [-z napis] – napis jest pusty
- [-n napis] – napis nie jest pusty
- [napis1 == napis2] – napis1 jest identyczny z napisem2. Można stosować relacje ==, !=, <, >
- [wart1 -eq wart2] – wartość1 jest numerycznie równa wartości2.

Można stosować następujące relacje:

-eq	równe
-ne	różne
-lt	mniej niż
-le	mniej lub równe
-gt	więcej niż
-ge	więcej lub równe

Wyrażenia warunkowe możemy łączyć relacjami logicznymi:

-o	lub
-a	i

8.3. Przykłady skryptów

```
#!/bin/bash
i=1
while [ $i -le 10 ]; do
    echo $i
    i=$((i+1))
    sleep 1
done
```

-wypisuje liczby od 1 do 10, kolejno co 1 sekundę

```
#!/bin/bash
if [ -z $1 ]; then
    echo Używaj podając wyrażenie do obliczenia
fi
echo $(( $1 ))
```

- oblicza wartość wyrażenia arytmetycznego podanego jako pierwszy parametr uruchomienia programu

```
#!/bin/bash
for nazwa in $(ls); do
    if [ -f $nazwa -a -x $nazwa ]; then
        echo $nazwa
    fi
done
```

- wypisuje tylko te nazwy plików, z bieżącego katalogu, które są programami

9. Informacja o systemie i jego stanie

uname	- nazwa systemu
uname -a	- pełne informacje o systemie i wersji
df	- wolna przestrzeń dyskowa
du	- zajęta przestrzeń w katalogu
uptime	- czas od restartu systemu i statystyki obciążenia procesora
top	- lista działających procesów plus statystyki
lspci	- lista urządzeń PCI
lsmod	- lista załadowanych sterowników urządzeń

Informacji dostarczają również pliki z katalogu proc np.:
/proc/cpuinfo, **/proc/meminfo**, **/proc/devices** itp.

9.1. Zakładanie i usuwanie kont użytkowników

groupadd	- utworzenie nowej grupy
groupdel	- usunięcie grupy
useradd	- założenie nowego konta
usermod	- modyfikacja parametrów konta
userdel	- sunięcie konta
adduser	- skrypt automatyzujący zakładanie konta
chfn	- zmiana informacji dodatkowych o koncie
finger	- sprawdzenie informacji o koncie
passwd	- zmiana hasła użytkownika
chgrp	- zmiana grupy pliku
chown	- zmiana właściciela pliku
chmod	- zmiana atrybutów uprawnień do pliku

Informacje o kontach użytkowników są przechowywane w pliku `/etc/passwd`, a informacje o hasłach i terminach ważności konta w pliku `/etc/shadow`. Plik `/etc/passwd` jest czytelny dla wszystkich, pliku `/etc/shadow`, ze względów bezpieczeństwa, nie może odczytywać żaden użytkownik.

Jeżeli przy zakładaniu konta jest tworzony katalog użytkownika wraz z opcją `m`, to są do niego kopiowane wszystkie pliki z katalogu `/etc/skel/` (oczywiście użytkownik staje się właścicielem tych plików).

9.2. Zlecenie poleceń do wykonania w określonym czasie

- at HH:MM** - dodanie nowego zadania do kolejki
- atq** - wyświetlenie kolejki zadań do wykonania
- atrm N** - usunięcie zadania o numerze N z kolejki

Dodając nowe zadanie podajemy kolejne polecenia z wiersza poleceń i kończymy kombinacją klawiszy Ctrl-D.

Polecenia **at** mogą używać wyłącznie użytkownicy wymienieni w pliku `/etc/at.allow` albo wszyscy, którzy nie są wymienieni w pliku `/etc/at.deny`.

9.3. Okresowe wykonywanie poleceń

- crontab -e** - edycja tablicy zadań okresowych
- crontab -r** - usunięcie tablicy zadań okresowych
- crontab -l** - wypisanie tablicy zadań okresowych

Polecenia **crontab** mogą używać wyłącznie użytkownicy wymienieni w pliku `/etc/cron.allow` albo wszyscy, którzy nie są wymienieni w pliku `/etc/cron.deny`.

Składnia pliku crontab:

MINUTA GODZINA DZIEŃ MIESIĄC DZIEŃ-TYGODNIA POLECENIE

Przykłady:

* * * * * polecenie_co_minute

0 0 * * * polecenie_o_polnocy

0 12 * * 0 polecenie_w_południe_w_kazda_niedziele

45 6 * 1-7,9-12 1-5 w_dni_robocze_bez_sierpnia

59 8-20/3 * * * co_trzecia_godzine_od_8-mej_do_20-tej

0 0 13 * 5 w_kazdy_piątek_i_kazdego_13-tego

Dopuszczalne jest używanie liczb rozdzielonych przecinkami, zakresów rozdzielonych minusem i wyborów co n-tej wartości po znaku / (ang. *slash*). W miejsce miesięcy i dni tygodnia można używać angielskich skrótów trzyliterowych np.: *jan, feb, ..., mon, tue, ...*. W większości systemów zarówno liczba 0 jak i liczba 7 mogą pełnić funkcję oznaczenia niedzieli. Jeżeli określony jest równocześnie dzień miesiąca i dzień tygodnia to polecenie działa niezależnie dla obu warunków.

9.4. Systemowe okresowe wykonywanie poleceń

W pliku `/etc/crontab` przechowywana jest systemowa tablica zadań do okresowego wykonania. Składnia podobna jak w poleceniu `crontab`, ale przed poleceniem podajemy nazwę użytkownika z uprawnieniami którego polecenie zostanie wykonane. W pliku tym zwykle są umieszczone wywołania uruchamiające wszystkie skrypty umieszczone w katalogach `/etc/cron.hourly/`, `/etc/cron.daily/`, `/etc/cron.weekly/`,

/etc/cron.monthly/ odpowiednio co godzinę, codziennie, co tydzień i raz w miesiącu.

9.5. Inicjacja systemu

Tryby pracy procesu init:

- 0 - zamknięcie systemu
- 1 - tryb pracy jednego użytkownika (root – bez hasła)
- 2-4 - tryby pracy wielu użytkowników bez trybu graficznego
- 5 - tryb pracy wielu użytkowników w środowisku X-Window
- 6 - restart systemu

Plik */etc/inittab* zawiera konfigurację domyślnego trybu pracy systemu. Tryb domyślny można zmienić dopisując w menu startowym za nazwą jądra systemu odpowiedni numer.

W katalogu */etc/init.d/* znajdują się skrypty startowe usług systemowych. Katalogi */etc/rc0.d/*, */etc/rc1.d/*, ..., */etc/rc6.d/* zawierają linki symboliczne do skryptów w katalogu *init.d* o nazwach poprzedzonych znakiem K albo S i dwucyfrowym numerem. Skrypty o nazwach rozpoczynających się od K są uruchamiane z parametrem stop, a skrypty rozpoczynające się od S z parametrem start, w celu, odpowiednio, zatrzymania i uruchomienia określonych usług w czasie inicjacji określonego trybu pracy systemu.

Przy starcie systemu jako ostatni, niezależnie od numeru trybu, jest uruchamiany skrypt o nazwie */etc/rc.d/rc.local*.

Lokalizacja katalogów typu *rc0.d* może być w różnych systemach różna. Najczęściej jest to katalog */etc/* albo */etc/rc.d/*.

10. Praca w sieci

10.1. Konfiguracja do pracy w sieci

```
ifconfig eth0 <IP> netmask <MASKA>
route add default gw <BRAMA>
/etc/resolv.conf
    nameserver <DNS1>
    nameserver <DNS2>
```

10.2. Poczta elektroniczna

```
mail user, mail user@serwer.domena
echo "Pozdrawiam" | mail -s "Temat" user
mail -s "Temat" user < plik
```

Czytanie poczty

mail

- h – lista e-maili
- 1 – czytaj pierwszy list
- 2 – czytaj drugi list itd.
- n – czytaj następny list
- d – usuń ostatnio czytany list
- q – wyjdź

Standardowo raz przeczytana poczta jest przenoszona ze skrzynki pocztowej serwera do pliku mbox w katalogu domowym użytkownika.

Czytanie przeczytanej poczty (z pliku mbox)

mail -f mbox

Inne konsolowe programy pocztowe

mutt

pine

10.3. Transfer plików

ftp

```
>open ftp.task.gda.pl
>user anonymous
>pass user@localhost
>passive
>ls
>cd pub
>ls
>binary
>get plik
>bye
```

10.4. WWW

Przeglądarki pracujące w trybie tekstowym:

elinks, links, lynx

10.5. Automatyczne pobieranie treści

wget ftp://192.168.2.1/pub/128k