

System Linux od podszewki

Marek Gózdź

25 lutego 2025

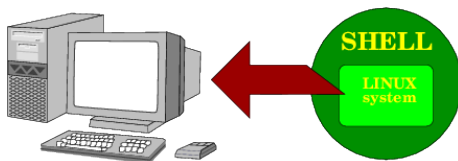
LITERATURA

- Alan Hicks i inni,
Slackware Linux Essentials 2nd ed. (czyli: SlackBook 2.0),
www.slackware.com
- Ramesh Natarajan,
Linux 101 Hacks 2nd Edition eBook,
www.thegeekstuff.com/linux-101-hacks-ebook/
- M. Camou, J. Goerzen, A. Van Couwenberghe,
Debian Linux. Księga eksperta, Helion 2001
- D.J. Barrett, R.E. Silverman, R.G. Byrnes,
Linux. Bezpieczeństwo. Receptury, Helion 2003
- C. Newham, B. Rosenblatt,
bash. Wprowadzenie, Helion 2006
- E. Foster-Johnson, J.C. Welch, M. Anderson,
Od podstaw. Skrypty powłoki, Helion 2006

System Linux – cz. 1

- bash
- skrypty powłoki
- dialog
- awk
- sed

Powłoka pośredniczy między systemem a użytkownikiem. Udostępnia interface: uruchamia aplikacje, obsługuje I/O.



POWŁOKA („linia komend”)

- sh i klony (csh, tcsh, ksh, pdksh, ash, zsh...)
- powłoka GNU bash Bourne Again Shell (od Steve’a Bourne’a)

Powłokę bash można wywołać na trzy sposoby. Każdy z nich powoduje inne działanie i inną konfigurację powłoki.

Interactive login shell

```
[bash]# bash -
```

czyta w kolejności pliki konfiguracyjne

- /etc/profile
- pierwszy z istniejących:
 - ~/.bash_profile
 - ~/.bash_login
 - ~/.profile
- ~/.bash_logout

Jest to powłoka logowania, ustawiająca całe środowisko pracy zgodnie z ustawieniami systemowymi. Jeśli w tej powłoce działa program login, to jej zabicie powoduje wylogowanie użytkownika.

Interactive shell

```
[bash]# bash
```

czyta ustawienia z pliku lokalnego `~/.bashrc`

Standardowa powłoka taka, jak w terminalach. Aby mieć w niej ustawienia globalne, dobrze jest umieścić w `.bashrc` odnośnik do `/etc/profile` i `/etc/bashrc`. Ale czasami użytkownik chce mieć zupełnie inne ustawienia...

Non-interactive shell

Środowisko wykonywania skryptów powłoki

czyta ustawienia ze zmiennej środowiskowej `$BASH_ENV`.

Jest to powłoka, w której wywołują się skrypty. Wraz z zakończeniem się skryptu jest ona również zamykana. Nie ma do niej dostępu chyba, że przez lukę w aplikacji (exploit).

Dodatkowe cechy i funkcje basha:

- kontrola procesów z możliwością ich usypiania i wybudzania
- przekierowania i potoki
- powiadomienia i komunikacja z innymi powłokami
- system pomocy `help`
- kontrola i obsługa historii poleceń `~/.bash_history`
- tryby pracy (skróty klawiszowe edytora) `emacs` i `vi`
- programowanie (pętle, instrukcje warunkowa, zmienne, funkcje...)
- autouzupełnianie klawiszem `TAB` (może być programowalne, patrz pakiet `bash completion`)

Przykładowe zawartości plików konfiguracyjnych powłoki

Plik lokalny .bash_profile

```
# ~/.bash_profile

if [ -f ~/.bashrc ]; then
. ~/.bashrc
fi

export BASH_ENV=$HOME/.bashrc
```

W tym pliku czytany jest (jeśli istnieje) plik lokalny .bashrc. Również tworzona jest globalna zmienna środowiskowa BASH_ENV ze ścieżką do tego pliku. W ten sposób plik .bashrc ustawia środowisko dla basha niezależnie od metody wywołania.

Plik lokalny .bashrc

```
# ~/.bashrc

if [ -r /etc/bashrc ]; then
. /etc/bashrc
fi

if [ -r /etc/profile ]; then
. /etc/profile
fi

alias poczta='ssh -l login poczta.umcs.lublin.pl'
```

Zawiera odnośniki do plików globalnych oraz dodatkowe ustawienia użytkownika.

Plik lokalny `.bash_logout`

```
# ~/.bash_logout
```

```
clear
```

Tu można umieścić zwalnianie prywatnych zmiennych, zabijanie procesów, kasowanie historii basha itp.

Globalny plik bashrc

```
# /etc/bashrc
alias ls='/bin/ls $LS_OPTIONS'
alias d='dir'
alias dir='/bin/ls $LS_OPTIONS --format=vertical'
alias vdir='/bin/ls $LS_OPTIONS --format=long'
alias ll='ls -la'
alias v='ls -l'
alias mc='. /usr/share/mc/bin/mc-wrapper.sh'
alias s='cd ..'
alias x='exit'
alias df='df -kh'
```

Aliaszy służą do tworzenia skrótów do komend. Można je przypisywać na poziomie systemu lub lokalnie w danej powłoce, np.:

```
[bash]# alias cdrom='mount /dev/cdrom ; cd /mnt/cdrom'
[bash]# unalias cdrom
```

Plik /etc/profile w slackware

```
# /etc/profile: This file contains system-wide defaults used
# by all Bourne (and related) shells.

# Set the values for some environment variables:
export MINICOM="-c on"
export MANPATH=/usr/local/man:/usr/man:/usr/X11R6/man
export HOSTNAME="'cat /etc/HOSTNAME'"
export LESSOPEN="|lesspipe.sh %s"
export LESS="-M"

# If the user doesn't have a .inputrc, use the one in /etc.
if [ ! -r "$HOME/.inputrc" ]; then
    export INPUTRC=/etc/inputrc
fi

# Set the default system $PATH:
PATH="/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/games"
```



```
if [ "'id -u'" = "0" ]; then
    echo $PATH | grep /usr/local/sbin 1> /dev/null 2> /dev/null
    if [ ! $? = 0 ]; then
        PATH=/usr/local/sbin:/usr/sbin:/sbin:$PATH
    fi
fi

if [ "$TERM" = "" -o "$TERM" = "unknown" ]; then
    TERM=linux
fi
```

```
# Set a default shell prompt:
#PS1='hostname':pwd'# '
if [ "$SHELL" = "/bin/pdksh" ]; then
    PS1='! $ '
elif [ "$SHELL" = "/bin/ksh" ]; then
    PS1='! ${PWD/#$HOME/~}$ '
elif [ "$SHELL" = "/bin/zsh" ]; then
    PS1='%n@m:%~%# '
elif [ "$SHELL" = "/bin/ash" ]; then
    PS1='$ '
else
    PS1='[\u@\h \W]\$ '
fi
PS2='\u \W> '
export PATH DISPLAY LESS TERM PS1 PS2
```

```
umask 022
```

```
# Append any additional sh scripts found in /etc/profile.d/:
for profile_script in /etc/profile.d/*.sh ; do
    if [ -x $profile_script ]; then
        . $profile_script
    fi
done
unset profile_script

# For non-root users, add the current directory
# to the search path:
if [ ! "`id -u`" = "0" ]; then
    PATH="$PATH:."
fi
```

Składnia basha

Metaznaki i wyrażenia regularne

<code>?</code>	dowolny 1 znak
<code>*</code>	dowolny ciąg znaków
<code>[12dsa]</code>	dowolny znak ze zbioru {1,2,d,s,a}
<code>[!12dsa]</code>	dowolny znak nie należący do zbioru {1,2,d,s,a}
<code>[a-c]</code>	zakres od a do c, czyli a, b lub c
<code>[a-z]</code>	wszystkie małe litery
<code>[!0-9]</code>	wszystkie nie cyfry
<code>[0-9!]</code>	cyfry i wykrzyknik
<code>[a-zA-Z0-9_-]</code>	wszystkie litery, cyfry, podkreślenie i myślnik

Przykład: *robert,daniel,ed,franek,fred*

<i>fr*</i>	<i>franek fred</i>
<i>*ed</i>	<i>ed fred</i>
<i>r*</i>	<i>robert</i>
<i>*e*</i>	<i>robert daniel ed franek fred</i>
<i>*r*</i>	<i>robert franek fred</i>
<i>*</i>	<i>robert,daniel,ed,franek,fred</i>
<i>d* </i>	<i>daniel</i>
<i>g*</i>	<i>g* (!!!)</i>

Inne znaki specjalne

~	katalog domowy
#	komentarz
\$	wartości zmiennych i wyrażeń
&	zadanie drugoplanowe
*?	metaznaki
(xxx)	wykonaj xxx w podpowłóce
\	kasuje specjalne znaczenie znaku
	potok
[abc]	zbiór znaków
{abc}	blok poleceń

Znaki specjalne c.d.

;	separator poleceń powłoki (w skrypcie =newline)
'	silny cudzysłów
"	słaby cudzysłów
`	odwrotny cudzysłów
<>	przekierowania
/	separator katalogów
!	zaprzeczenie logiczne

Przekierowanie wejścia–wyjścia: `cat plik1 plik2 > plik1+2`

<code>pol1 pol2</code>	potok (pipe)
<code>> plik</code>	<code>stdout=plik</code>
<code>< plik</code>	<code>stdin=plik</code>
<code>>> plik</code>	<code>stdout=plik</code> (dopisanie)
<code><> plik</code>	<code>stdout=plik, stdin=plik</code>
<code>n > &</code>	zduplikowanie <code>stdout</code> na deskryptor <i>n</i>
<code>n > &m</code>	deskryptor <i>n</i> będzie kopią deskryptora pliku wyjściowego
<code>n < &m</code>	deskryptor <i>n</i> będzie kopią deskryptora pliku wejściowego
<code>& >plik</code>	<code>stdout</code> i <code>stderr</code> do pliku

Standardowe strumienie danych:

Deskryptor	Opis
0	standardowe wejście
1	standardowe wyjście
2	standardowe wyjście błędów

Przykłady:

```
[bash]# make install 2 > error-log
[bash]# mount /dev/sdb1 > /dev/null 2>&1
[bash]# dmesg | less
[bash]# sort < plik1 | lpr
```

Zamiana strumieni stdout i stderr:

```
[bash]# make 3>&1 1>&2 2>&3
```

Manipulowanie procesami

- jobs
- bg, polecenie &
- fg
- ps (-a, -A, aux)
- top, htop
- nice
- nohup
- kill (-9), killall, ^z, ^c, ^d

Deklarowanie zmiennych lokalnych i globalnych:

```
[bash]# zmienna=cośtam
[bash]# export zmienna    ← będzie widoczna w podpowłocie!
[bash]# echo zmienna
zmienna
[bash]# echo $zmienna
cośtam
[bash]# bash              ← nowa podpowłoka
[bash]# echo $zmienna
cośtam                    ← stara znajoma
```

Zmienne środowiskowe i zmienne globalne pokaże polecenie set.

```
[bash]# set | less
```

```
BASH=/usr/bin/bash
```

```
BASH_COMPLETION=/etc/bash_completion
```

```
BASH_COMPLETION_DIR=/etc/bash_completion.d
```

```
BASH_ENV=/home/marek/.bashrc
```

```
BASH_LINENO=()
```

```
BASH_SOURCE=()
```

```
BASH_VERSION='3.1.17(2)-release'
```

```
COLUMNS=80
```

```
DESKTOP_SESSION=wmaker
```

```
(...)
```

```
zmienna=costam
```

Przykłady poprawnego użycia przekierowań

Polecenie

```
[bash]# echo 2 * 3 > 5 to nierownosc prawdziwa
```

utworzy plik o nazwie 5 zawierający:
2 [nazwy wszystkich plików z bieżącego katalogu] 3 to nierownosc prawdziwa

Polecenie

```
[bash]# echo 2*3 > 5 to nierownosc prawdziwa
```

utworzy plik o nazwie 5 zawierający:
2*3 to nierownosc prawdziwa

Polecenie

```
[bash]# echo "2 * 3 > 5 to nierownosc prawdziwa"
```

wyświetli całe zdanie.

Przykład poprawnego użycia cudzysłowów

```
[bash]# zmienna=costam ; export zmienna
```

```
[bash]# echo zmienna
```

```
zmienna
```

```
[bash]# echo $zmienna
```

```
costam
```

```
[bash]# echo "Moja zmienna to $zmienna"
```

```
Moja zmienna to costam
```

```
[bash]# echo 'Moja zmienna to $zmienna'
```

```
Moja zmienna to $zmienna
```

```
[bash]# unset zmienna ; echo $zmienna
```

```
[bash]# zmienna='ls'
```

```
[bash]# echo $zmienna
```

```
hibernate/ praca/ soft/ GNUstep/ kwant-alg.txt raport.pdf
```

```
tmp/ WFJ/
```

Zmienne wbudowane: tryb edycji

HISTCMD	numer bieżącego polecenia
HISTCONTROL	kontrola historii poleceń
HISTIGNORE	j.w.
HISTFILE	j.w.
HISTFILESIZE	j.w.
HISTSIZE	j.w.
HISTTIMEFORMAT	j.w.
FCEDIT	edytor dla polecenia <code>fc</code>

`fc` pozwala na edytowanie i używanie ponownie poleceń z historii.

Użyteczny alias to `r='fc -s'` tak, że np.

`r cc` wykona ostatnią komendę zaczynającą się od `cc`

`r` powtórzy ostatnią komendę

Zmienne wbudowane: poczta

MAIL	lokacja pliku poczty /var/spool/mail/login
MAILCHECK	domyślnie 60 (sek.)
MAILPATH	jak MAIL, ale lista rozdzielona :

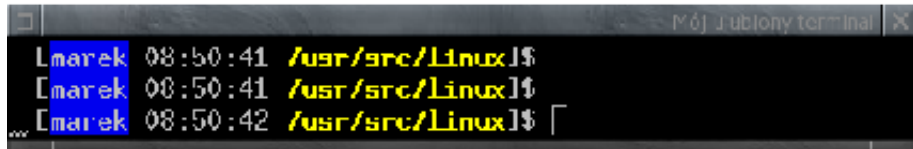
Zmienne wbudowane: znak zachęty

- PS1 standardowy prompt
- PS2 wtórny znak zachęty
- PS3 jeszcze wtórniejszy
- PS4 najbardziej wtórny z wtórnych

Przykład:

```
PS1=\
```

```
' \[\033[44m\]\u\[\033[0m\] \t \[\033[1;33m\]\w\[\033[0m\]\$ '
```



```
Mój ulubiony terminal X  
[marek 08:50:41 /usr/src/linux]$  
[marek 08:50:41 /usr/src/linux]$  
[marek 08:50:42 /usr/src/linux]$
```

man bash, man console_codes

Zmienne wbudowane: ścieżki

PATH poszukiwanie plików

CDPATH rozszerza poszukiwanie plików przez cd

```
[bash]# echo $PATH
```

```
/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/games:  
/usr/lib/java/bin:/usr/lib/java/jre/bin:/opt/kde/bin:  
/opt/xaralx/bin:/opt/openoffice.ux.pl2.2/program:  
/opt/maple9.5/bin:/usr/lib/qt/bin:/usr/share/texmf/bin:.
```

```
[bash]# PATH=$PATH:$HOME/bin
```

```
[bash]# PATH=.:$PATH
```

HOME	katalog domowy
SECONDS	czas od uruchomienia powłoki
BASH	ścieżka do powłoki uruchomionej przez użytkownika
SHELL	ścieżka do powłoki
BASH_VERSION	wersja
BASH_VERSINFO	wersja
PWD	bieżący katalog
OLDPWD	poprzedni katalog
COLUMNS	szerokość terminala (ekranu)
LINES	wysokość terminala (ekranu)
EDITOR	domyślny edytor tekstowy
TERM	typ terminala (linux,xterm,vt100...)

Skrypty powłoki

Ogólna struktura skryptu

```
#!/bin/bash
# komentarz

function f1{
    polecenie1 ; polecenie2
    polecenie3}

f2 ()
{
    polecenia}

<<polecenia skryptu>>
```

Zmienne specjalne dostępne wewnątrz skryptów powłoki:

\$0	nazwa skryptu (polecenie wywołujące skrypt)
\$n	n-ty argument wywołania skryptu
\$*	"\$1 \$2 ... \$N"
\$@	"\$1" "\$2" ... "\$N"
\$#	=N, liczba argumentów

Przykład:

```
# skrypt o nazwie alicja
echo "alicja: $@"
echo "$0: $1 $2 $3 $4"
echo "$# argumenty"
```

Wywołanie i wynik działania:

```
[bash]# ./alicja w krainie czarów
```

```
alicja: w krainie czarów
alicja: w krainie czarów
3 argumenty
```

Tablice są dostępne od bash 2.0. Definicja listy (tablicy):

```
lista[2]="drugi element"  
lista[0]=patelnia3  
lista[5]="4,"
```

lub

```
lista=(patelnia3 [5]="4," [2]="drugi element")
```

Kolejność nie jest ważna, ale w przypadku nie podania numeru brany jest kolejny. Numeracja zaczyna się od zera. Indeksy mogą być nieujemnymi wyrażeniami arytmetycznymi. Ograniczenie na największy element to 599 147 937 791 (prawie 600 mld.)

Odwołanie przez: `tablica[2]`, np. `echo ${tablica[2]}`

<code>"\${tablica[@]}"</code>	zwróci wszystkie zdefiniowane elementy w postaci odrębnych słów
<code>"\${tablica[*]}"</code>	zwróci wszystkie zdefiniowane elementy w postaci pojedynczego słowa
<code>\${!tablica[@]}</code>	podaje indeksy zadeklarowanych elementów (bash 3.0)
<code>\${#tablica[2]}</code>	podaje długość elementu 2

Przykład:

```
[bash]# lista[2]="drugi element"
```

```
[bash]# lista[5]="4,"
```

```
[bash]# echo ${lista[@]}
```

```
drugi element 4,
```

```
[bash]# echo "${!lista[@]}"
```

```
2 5
```


<code>\${zmienna#worzec}</code>	Jeśli wzorzec pasuje do początku zmiennej usuwa najkrótszą pasującą część i zwraca resztę.
<code>\${zmienna##worzec}</code>	Jeśli wzorzec pasuje do początku zmiennej usuwa najdłuższą pasującą część i zwraca resztę.
<code>\${zmienna%worzec}</code>	Jeśli wzorzec pasuje do końca zmiennej usuwa najkrótszą pasującą część i zwraca resztę.
<code>\${zmienna%%worzec}</code>	Jeśli wzorzec pasuje do końca zmiennej usuwa najdłuższą pasującą część i zwraca resztę.
<code>\${zmienna/worzec/ciąg}</code>	Najdłuższa część zmiennej pasująca do wzorca jest zastępowana przez ciąg.
<code>\${zmienna//worzec/ciąg}</code>	Wszystkie części zmiennej pasujące do wzorca są zastępowane przez ciąg.

Przykład:

ścieżka=/home/kamil/książka/długa.nazwa.pliku

<code>\${ścieżka##*/}</code>	długa.nazwa.pliku
<code>\${ścieżka#*/}</code>	kamil/książka/długa.nazwa.pliku
<code>\$ścieżka</code>	/home/kamil/książka/długa.nazwa.pliku
<code>\${ścieżka%.*}</code>	/home/kamil/książka/długa.nazwa
<code>\${ścieżka%%.*}</code>	/home/kamil/książka/długa
<code>\${ścieżka/./-}</code>	/home/kamil/książka/długa-nazwa.pliku
<code>\${ścieżka//./-}</code>	/home/kamil/książka/długa-nazwa-pliku

Po włączeniu w powłoce opcji `extglob` (patrz komenda `shopt`) działają rozszerzone operatory dopasowania:

`* + ? @ !`

Przykład:

WYRAŻENIE	PASUJE DO
<code>*(alicja kapelusznic zajac)</code>	<code>null, alicja, alicjakapelusznic,...</code>
<code>+(alicja kapelusznic zajac)</code>	<code>alicja, alicjakapelusznic,...</code>
<code>?(alicja kapelusznic zajac)</code>	<code>null, alicja, kapelusznic lub zajac</code>
<code>@(alicja kapelusznic zajac)</code>	<code>alicja, kapelusznic lub zajac</code>
<code>!(alicja kapelusznic zajac)</code>	<code>wszystko oprócz alicja, kapelusznic i zajac</code>

Zaawansowane odwołania do zmiennych – operatory podstawienia

Zwrócenie wartości domyślnej

`${zmienna:-słowo}` jeśli zmienna jest niepusta zwracana jest jej wartość, w przeciwnym razie zwracane jest słowo

Podstawienie wartości domyślnej

`${zmienna:=słowo}` jeśli zmienna jest niepusta zwracana jest jej wartość, w przeciwnym razie podstawiane jest pod nią słowo

Przechwytywanie niezdefiniowanych zmiennych

`${zmienna:?kom}` jeśli zmienna jest niepusta zwracana jest jej wartość, w przeciwnym razie wyświetlany jest komunikat a skrypt kończony

Sprawdzenie czy zmienna istnieje

`${zmienna:+słowo}` jeśli zmienna jest niepusta zwracane jest słowo

Wycięcie kawałka ciągu znaków

`${zmienna:przesunięcie:długość}`

Zwraca podciąg zmiennej zaczynając na przesunięciu, mający zadaną długość. Pierwszy znak zmiennej ma numer 0.

- Jeśli długość nie jest podana zakładane jest „aż do końca”.
- Jeśli przesunięcie jest ujemne, będzie naliczane od końca zmiennej.
- Jeśli zmienną jest @ to długość oznacza liczbę parametrów pozycyjnych (\$1, \$2, ...) począwszy od parametru przesunięcie.

Instrukcje warunkowe i pętle w bashu:

- if/else
- for
- while
- until
- case
- select

Instrukcja warunkowa if

```
if warunek
then
    instrukcje
[elif warunek2
    then instrukcje2...]
[else
    instrukcje3]
fi
```

warunek: polecenie test lub wyrażenie logiczne

UWAGA NA SKŁADNIĘ!

Polecenie test

Porównywanie stringów:

<code>str1 = str2</code>	identyczność
<code>str1 != str2</code>	nie identyczność
<code>str1 < str2</code>	str1 mniejsze od str2
<code>str1 > str2</code>	str1 większe od str2
<code>-n str1</code>	str1 niepusty (długość > 0)
<code>-z str1</code>	str1 pusty (długość =0)

Porównywanie następuje **leksykograficznie** (np. "6">"57").

Atrybuty plików:

<code>-a plik</code>	<code>(-e plik)</code>	plik istnieje
<code>-d plik</code>		plik istnieje i jest katalogiem
<code>-f plik</code>		plik istnieje i jest zwykłym plikiem
<code>-r plik</code>		użytkownik ma prawo do odczytu pliku
<code>-s plik</code>		plik istnieje i jest niepusty
<code>-w plik</code>		użytkownik ma prawo do zapisu pliku
<code>-x plik</code>		użytkownik ma prawo do wykonania pliku
<code>-N plik</code>		plik zmodyfikowany od ostatniego odczytu
<code>-O plik</code>		użytkownik jest właścicielem pliku
<code>-G plik</code>		użytkownik należy do grupy pliku
<code>plik1 -nt plik2</code>		porównanie czasu modyfikacji plików
<code>plik1 -ot plik2</code>		“newer than” i “older than”

Operatory testów arytmetycznych:

-lt	less than
-le	less or equal
-eq	equal
-ge	greater or equal
-gt	greater than
-ne	not equal

Przykład:

```
if [ 'id -u' -eq 0 ]; then
echo "root"
else
echo "nie-root"
fi
```

Pętla for

```
for nazwa [in lista]
do
    instrukcje zawierające $nazwa
done
```

Jeśli brak [in lista] przyjmowane jest "\$@"

Standardowe zastosowania:

```
for plik in `ls`; do ...
for dir in $PATH; do ...
for filename in "$@"; do
```

Numeryczny for (od bash 2.04)

```
for (( i=1 ; i<=120 ; i++ ))  
do  
    instrukcje  
done
```

Dla wcześniejszych wersji powłoki pozostaje konstrukcja:

```
for i in `seq 1 120`  
do  
    instrukcje  
done
```

Instrukcja warunkowa case

Często służy do sprawdzenia z jakim parametrem wywołano skrypt.

```
case wyrażenie in
    wzorzec1)
        instrukcje1 ;;
    wzorzec2)
        instrukcje2 ;;
    ...
esac
```

Przepis dla **pierwszego** dopasowanego wzorca jest wykorzystywany!
Dobrze jest dopisać jako ostatni wzorzec *)

Przykład zastosowania case (plik niepełny)

```
#!/bin/sh
# /etc/rc.d/rc.wicd

wicd_start() {...}
wicd_force_stop() {...}
wicd_stop() {...}

case $1 in
  start)
    wicd_start
    ;;
  stop)
    wicd_stop
    ;;
  *)
    wicd_force_stop
    wicd_start
    echo "Usage: $0 {start|stop|restart}"
    exit 1
  ;;
esac
```

Polecenie select

Służy do tworzenia prostych menu.

```
select nazwa [in lista]
do
    polecenia mogące używać $nazwa
done
```

Jeśli brak [in lista] przyjmowane jest "\$@"

Do zaawansowanego tworzenia menu i okienek w trybie tekstowym: dialog i biblioteka ncurses. To samo w trybie graficznym: biblioteki widgetów (qt, gtk, Lesstif, Motif, Tcl/Tk...).

Pętla warunkowa while

```
while warunek; do  
    instrukcje  
done
```

Będzie wykonana **zero** lub więcej razy.

Pętla warunkowa until

```
until warunek; do  
    instrukcje  
done
```

Będzie wykonana **jeden** lub więcej razy.

Bash i operacje arytmetyczne

bash potrafi wykonywać działania tylko na **liczbach całkowitych**.

Domyślnie wszystko jest traktowane jako zmienna napisowa (string) ale konstrukcja

`$((...))`

powoduje potraktowanie wszystkiego wewnątrz nawiasów jako wyrażenie arytmetyczne.

Wymiennie można używać polecenia `let`, np.

```
[bash]# x=$((17-2)) ; echo $x
```

15

```
[bash]# let x-- ; echo $x
```

14

```
[bash]# let y='x/3' ; echo $y
```

4

Dostępne operacje arytmetyczne

<code>+</code>	<code>-</code>	<code>*</code>	dodawanie, odejmowanie, mnożenie
<code>/</code>			część całkowita z dzielenia
<code>%</code>			reszta z dzielenia (dzielenie modulo)
<code>++</code>			pre- lub postinkrementacja <code>\$((++i))</code> <code>\$((i++))</code>
<code>-</code>			pre- lub postdekrementacja
<code>**</code>			podnoszenie do potęgi

Operacje bitowe

- « bitowe przesunięcie w lewo (mnożenie przez 2)
- » bitowe przesunięcie w prawo (dzielenie przez 2)
- & bitowa koniunkcja (mnożenie)
- | bitowa alternatywa (dodawanie)
- ~ bitowa negacja
- ^ dodawanie modulo 2 (bitowe XOR)

Dostępne operatory relacji

> <	nierówności mocne (ostre)
>= <=	nierówności słabe
==	równe
!=	różne
&&	logiczna koniunkcja (AND, mnożenie)
	logiczna alternatywa (OR, dodawanie)
!	logiczna negacja

Można zapisywać liczby w systemach pozycyjnych o podstawach z przedziału $B \in \langle 2, 36 \rangle$.

$$B\#N$$

oznacza liczbę N w zapisie z podstawą B (domyślnie 10). Oczywiście brak jest obsługi notacji prostej, dopełnieniowej i zmiennoprzecinkowej!

```
[bash]# let x='2#1001 + 16#f12' ; echo $x  
3867
```

Kalkulator bc

Kalkulator o dowolnej precyzji – język programowania bc. Opcje:

- l, -mathlib łąduje standardową bibliotekę
- q, -quiet mniej komunikatów
- s, -standard odrzuca rozszerzenia GNU

Precyzja liczb: *długość liczby* to liczba cyfr użyta do jej zapisania (bez znaków + - .); *skala* to liczba miejsc po kropce dziesiętnej.

- Obsługa zwykłych zmiennych i zmiennych tablicowych.
- Komentarze są ograniczone */* znakami ukośnika i gwiazdki */*.
- Obsługuje różne podstawy liczb (zmienna *ibase*).
- Obsługuje pętle.
- Obsługuje instrukcje warunkowe.
- Obsługuje definiowalne funkcje.

Standardowa biblioteka matematyczna bc definiuje skalę na 20, i predefiniuje następujące funkcje:

$s(x)$	$\sin(x)$, x w radianach
$c(x)$	$\cos(x)$, x w radianach
$a(x)$	$\arctg(x)$, wynik w radianach
$l(x)$	$\ln(x)$
$e(x)$	$\exp(x)$
$j(n, x)$	funkcja Bessel'a rzędu n z x

Przykład wywołania w skrypcie powłoki:

```
[bash]# pi=$(echo "scale=10; 4*a(1)" | bc -l)
[bash]# echo $pi
3.1415926532
```

Więcej: `man bc`

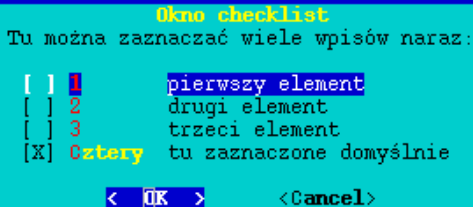
Program dialog

Program `dialog` służy do generowania prostych menu i list w trybie znakowym. Korzysta z biblioteki `ncurses`. Podstawowym zastosowaniem jest wywoływanie go wewnątrz skryptów, aby usprawnić komunikowanie się skryptu z użytkownikiem. Wiele instalatorów Linuksa (np. `slackware`) i konfiguratorów korzysta z tego polecenia.

Składnia: `dialog [opcje] typ-okna-i-jego-opis`

Kilka opcji:

- `-clear` czyści ekran
- `-title` nadaje tytuł oknu
- `-backtitle` dodaje podtytuł



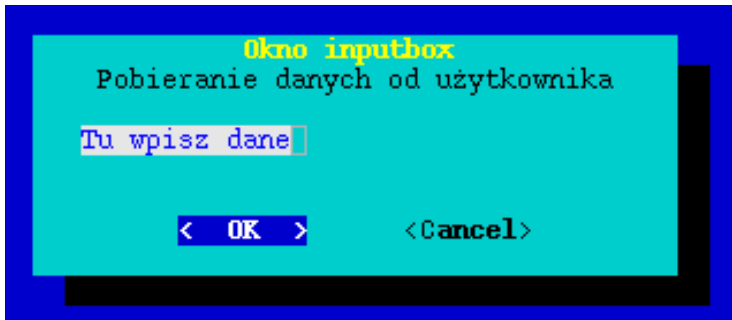
```
Okno checklist
Tu można zaznaczać wiele wpisów naraz:

[ ] 1      pierwszy element
[ ] 2      drugi element
[ ] 3      trzeci element
[X] Cztery tu zaznaczone domyślnie

< OK >      <Cancel>
```

--checklist tekst wysokość szerokość wysokość_listy

```
dialog --clear --separate-output --title "Okno checklist" \
--backtitle "Kurs programowania w shellu" \
--checklist "Tu można zaznaczać wiele wpisów naraz:" 10 70 4 \
"1" "pierwszy element" "off" \
"2" "drugi element" "off" \
"3" "trzeci element" "off" \
"Cztery" "tu zaznaczone domyślnie" "on" 2> wynik
```



--inputbox tekst wysokość szerokość

```
dialog --title "Okno inputbox" \  
--backtitle "Kurs programowania w shellu" \  
--inputbox "Pobieranie danych od użytkownika" 8 40 \  
"Tu wpisz dane"
```

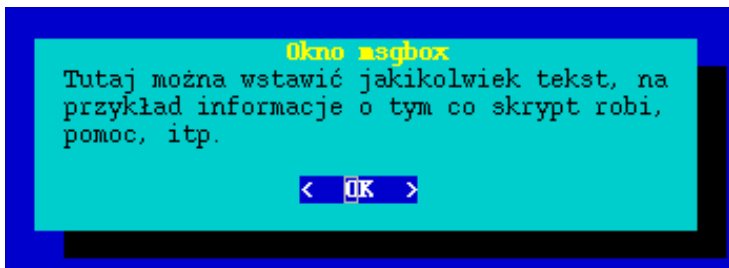


--menu tekst wysokość szerokość wysokość_menu element opis

```
dialog --title "Okno menu" \  
--backtitle "Kurs programowania w shellu" --menu "Menu" \  
9 50 3 1 jeden 2 dwa 3 trzy 4 cztery
```

Zwraca na strumieniu o deskrytorze 2 wartość wybranej opcji. Konieczne przekierowanie jej do pliku

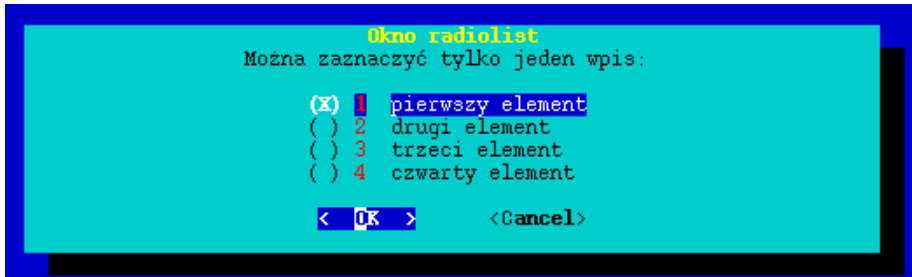
```
dialog --menu ... 2>out.plik
```



--msgbox tekst wysokość szerokość

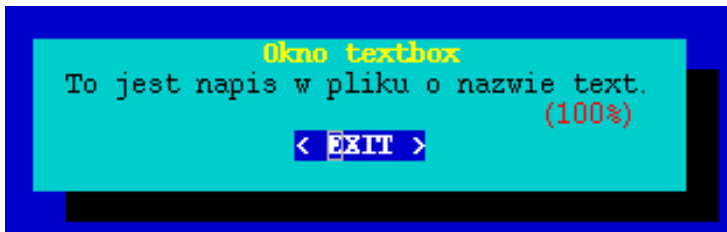
```
dialog --title "Okno msgbox" \  
--backtitle "Kurs programowania w shellu" \  
--msgbox "Tutaj można wstawić jakikolwiek tekst, na  
przykład informacje o tym co skrypt robi,  
pomoc, itp." 7 45
```

Bardzo podobne jest infobox ale nie ma przycisku OK i tekst zostaje na ekranie dopóki skrypt go nie wyczyści.



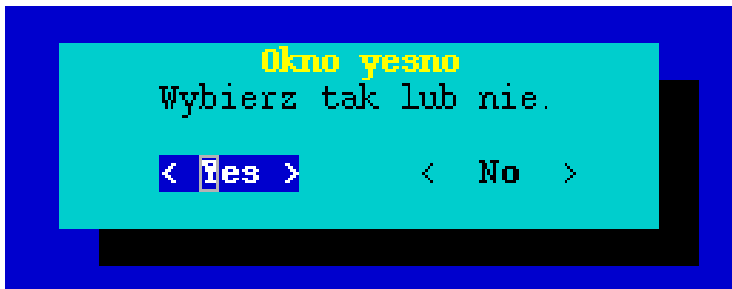
--radiolist tekst wysokość szerokość wysokość_listy

```
dialog --clear --title "Okno radiolist" \  
--backtitle "Kurs programowania w shellu" \  
--radiolist "Można zaznaczyć tylko jeden wpis:" 10 70 4 \  
"1" "pierwszy element" "off" \  
"2" "drugi element" "off" \  
"3" "trzeci element" "off" \  
"4" "czwarty element" "off"
```



--textbox plik wysokość szerokość

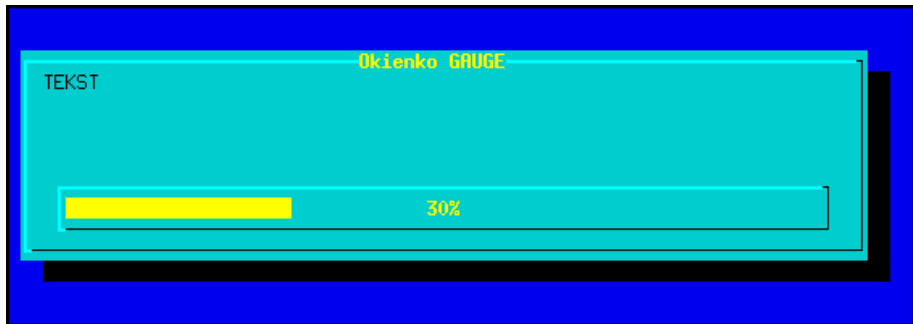
```
dialog --title "Okno textbox" \  
--backtitle "Kurs programowania w shellu" \  
--textbox text 5 40
```



--yesno tekst wysokość(linie) szerokość(znaki)

```
dialog --title "Okno yesno" \  
--backtitle "Kurs programowania w shellu" \  
--yesno "Wybierz tak lub nie." 5 30
```

Zwraca logiczną prawdę lub fałsz i może być użyte w konstrukcji
if ... then.



--gauge tekst wysokość szerokość procent

```
dialog --title "Okienko GAUGE" --gauge TEKST 10 75 30
```

Okienko gauge pokazuje miarkę procentową na dole ekranu.

Dialog jest dojrzałym i niemal zamrożonym projektem, choć ciągle aktywnym. Inne podobne programy to:

- `whiptail` – jest niemal w 100% kompatybilny z `dialogiem`, ale wykorzystuje bibliotekę `newt` zamiast `ncurses`
- `xdialog` – nie uaktualniana od 2006 r. wersja `dialoga` pracująca w trybie graficznym
- `zenity` – projekt GNOME działający pod `GTK+`
- `kdiallog` – projekt KDE działający pod `Qt`

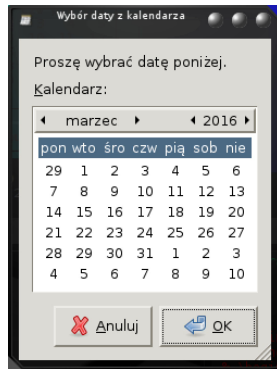
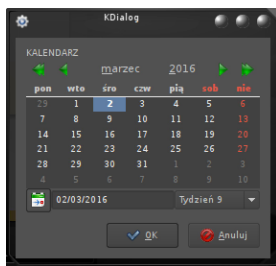
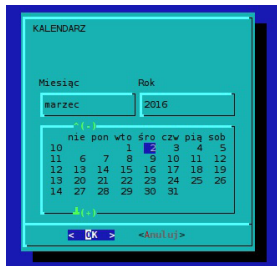
Wszystkie powyżej są obsługiwane niemal identycznie, mają bardzo podobną funkcjonalność.

Składnia może być przenoszona między tymi programami:

```
dialog --calendar "KALENDARZ" 6 40 2 3 2016
```

```
kdialog --calendar "KALENDARZ" 6 40 2 3 2016
```

```
zenity --calendar "KALENDARZ" 6 40 2 3 2016
```



AWK

AWK jest językiem skryptowym stworzonym z myślą o obróbce plików tekstowych.

Twórcami są Alfred Aho, Peter Weinberger, Brian Kernighan.

Wywołanie:

```
awk 'program' input1 input2...  
awk 'program' input1 input2 > output  
awk -f skrypt.awk input1 input2 > output
```

Struktura skryptu

```
# komentarz
BEGIN {komenda1 ; komenda2}
wzorzec1 {akcja1}
wzorzec2 {akcja21;
          akcja22;
          akcja23;} #inny komentarz
END {podprogramik końcowy}
```

Niekiedy cały program zamyka się w bloku BEGIN. Tam też ustawia się zmienne wbudowane i deklaruje wartości początkowe własnych zmiennych. Wzorce bazują na wyrażeniach regularnych (różne w zależności od wersji i systemu operacyjnego).

Każda linia pliku wejściowego jest przepuszczana przez cały skrypt awkowy. Po wykonaniu całego skryptu pobierana jest następna linia. Po obrobieniu pierwszego pliku wejściowego awk przechodzi do drugiego etc.

Zasada działania:

- 1 jeśli istnieje blok BEGIN jest on wykonywany przed wczytaniem pliku.in
- 2 każda linia pliku.in jest po kolei sprawdzana z wzorcami i jeśli któremuś odpowiada są wykonywane odpowiednie akcje na tej linii
- 3 gdy wszystkie linie pliku.in zostaną przerobione, wykonywany jest blok END

Pominięcie wzorca oznacza wykonanie akcji na każdej linii.

Pominięcie akcji oznacza wykonanie akcji domyślnej `{print $0}`

awk działa na **plikach**, **rekordach** i **polach** rekordów. Domyślnie są to:

- plik to plik wejściowy lub stdin
- rekord to linia pliku (ciąg zakończony znakiem nowej linii `\n` (*newline* LF; Unix) lub `\n\c` (*carriage return* LF CR; dos), lub `\c` (mac))
- pole to ciąg znaków ograniczony spacją

Zmienne wbudowane FS (field separator) i RS (record separator) pozwalają na zmianę sposobu podziału danych wejściowych. Np.

```
[bash]# awk 'BEGIN{FS=":"}; /root/ {print $0}' passwd
```

```
root:x:0:0::/root:/bin/bash
```

```
operator:x:11:0:operator:/root:/bin/bash
```

Wyrażenia regularne (RE)

\c	znak sterujący lub metaznak c
.	dowolny jeden znak
^	początek napisu
\$	koniec napisu
[abc]	dowolny ze znaków <i>a,b,c</i>
[^abc]	żaden ze znaków <i>a,b,c</i>
[a-z]	dowolna mała litera
r1 r2	RE r1 lub r2
r*	zero lub więcej napisów pasujących do r
r+	jeden lub więcej napisów pasujących do r
r?	zero lub jeden napisów pasujących do r
(r)	grupowanie wyrażeń

Escape sequences

<code>\\</code>	backslash <code>\</code>
<code>\/</code>	slash <code>/</code>
<code>\"</code>	cudzysłów <code>"</code>
<code>\a</code>	BEL alarm (ascii 7)
<code>\b</code>	BS backspace (ascii 8)
<code>\f</code>	FF formfeed (ascii 12)
<code>\n</code>	LF newline (ascii 10)
<code>\r</code>	CR carriage return (ascii 13)
<code>\t</code>	HT horizontal tab (ascii 9)
<code>\v</code>	VT vertical tab (ascii 11)
<code>\nnn</code>	kod ósemkowy znaku ascii
<code>\xhh</code>	kod szesnastkowy znaku ascii

Zmienne wbudowane

RS	record separator (oddziela linie, domyślnie NEWLINE)
ORS	output record separator
FS	field separator (oddziela kolumny w linii, domyślnie SPC)
OFS	output field separator
NF	number of fields (liczba kolumn w bieżącej linii)
FNR	numer bieżącego rekordu (obrabianej linii)
NR	liczba przeczytanych rekordów
\$x	kolumna numer x (x=0 oznacza całą linię)
FILENAME	nazwa aktualnie czytanego pliku
i inne...	

Operatory

+ - * /	zwykłe działania arytmetyczne
%	dzielenie modulo
^	potęgowanie
++	inkrementacja
--	dekrementacja
==	porównanie
!=	nie równa się
< <= >= >	zwykłe nierówności (mocne i słabe)
&&	koniunkcja
	alternatywa
!	zaprzeczenie logiczne
a?b:c	operator warunkowy (if a then b else c)

Instrukcje sterujące

<code>if (warunek) instrukcja1 else instrukcja2</code>	warunkowa
<code>while (warunek) instrukcja</code>	pętla
<code>do instrukcja while (warunek)</code>	pętla
<code>for (a ; b ; c) instrukcja</code>	pętla
<code>for (zmienna in tablica) instrukcja</code>	pętla
<code>break</code>	przerwanie pętli
<code>continue</code>	wznowienie pętli
<code>exit(wyrażenie)</code>	przerwanie programu
<code>next</code>	następny rekord
<code>nextfile</code>	następny plik (gawk)

Funkcje napisowe

<code>gsub (r,s,t)</code>	zamienia RE r na napis s w wyrażeniu t
<code>gensub(r,s,a,t)</code>	uogólnione gsub
<code>index(s,t)</code>	podaje gdzie się zaczyna napis t w napisie s
<code>length(s)</code>	podaje długość napisu s
<code>match(s,t)</code>	podobnie jak index
<code>split(s,a,fs)</code>	z napisu s tworzy tablicę napisów a z FS=fs
<code>sub(r,s,t)</code>	najdłuższy napis w t pasujący to RE r jest podmieniany na s
<code>substr(s,p,n)</code>	wycina z s ciąg n znaków zaczynając od znaku p
<code>tolower(s)</code>	zamiana na małe litery
<code>toupper(s)</code>	zamiana na wielkie litery

Wejście

<code>getline</code>	zwraca 1 jeśli wczytano następny rekord, 0 jeśli EOF
<code>getline < plik</code>	inny plik niż bieżący
<code>program getline</code>	potok np. z <code>ls -la</code>

Przykład:

```
while (getline < plik.in > 0) {...}
```

wykona {...} na całym pliku `plik.in`

Wyjście

<code>print</code>	output na stdout, plik, potok
<code>printf</code>	j.w. ale z formatowaniem (patrz manual)

Matematyka

awk przekształca **napis** w **liczbę** i na odwrót zawsze jeśli jest taka potrzeba. Nie trzeba się martwić deklarowaniem typów zmiennych!

atan2(x,y) arctg($\frac{x}{y}$)

cos(x)

sin(x)

exp(x) e^x

int(x) [x] cecha (część całkowita) liczby

log(x) $\ln(x)$

sqrt(x) $\sqrt{x}$

rand() losowa liczba z przedziału $< 0, 1 >$

srand(x) liczba z generatora liczb pseudolosowych
inicjowanego przez x

Tablice w awk są to pary (indeks, wartość) przy czym:

- indeksy mogą być dowolnymi **napisami** czyli nie muszą być numerowane, a nawet jeśli są, to numery nie muszą być nadawane w żadnej określonej kolejności
- wartościami mogą być dowolne **napisy**

Sprawdzenie czy dany indeks jest zadeklarowany w tablicy:

```
if (2 in tab1) print "Istnieje element o indeksie 2"
```

BŁĘDNY kod

```
if (tab1[2] != "") print "Istnieje element o indeksie 2"  
  
stworzy nowy element tab1[2]!
```

Deklaracja elementu: `tablica[indeks] = wartość`

Odwołanie poprzez: `tablica[indeks]`

Przeskanowanie całej tablicy (pod zmienną podstawiane są po kolei wszystkie elementy z tablicy)

```
for (zmienna in tablica)
    cośtam
```

Usunięcie elementu: `delete tablica[indeks]`

UWAGA: to nie jest to samo co `tablica[indeks] = ""`

Nie można wykasować tablicy i użyć jej nazwy jako innej zmiennej:

```
a[1] = 3; delete a; a = 3
```

Można operować na tablicach wielowymiarowych

```
tab[i1,i2] = 3
```


drukuje potęgę 2

```
BEGIN{  
    for (i = 1; i <= 100; i *= 2)  
        print i  
}
```

wyświetla 10 pierwszych linii pliku

```
{if(NR<=10) print$0}
```

wyświetla 10x każdą linię pliku

```
{ i = 1  
    do {  
        print $0  
        i++  
    } while (i <= 10)  
}
```

```
# find smallest divisor of num
{
    num = $1
    for (div = 2; ; div++) {
        if (num % div == 0) {
            printf "Smallest divisor of %d is %d\n", num, div
            break
        }
        if (div*div > num) {
            printf "%d is prime\n", num
            break
        }
    }
}
```

```
# Po każdej linii dokłada linię pustą  
awk '{print $0 ; print ""}' plik.in
```

```
# Numerowanie linii w każdym pliku z osobną (\t=TAB)  
awk '{print FNR "\t" $0}' *.in  
# FNR-->NR da numerację ciągłą
```

```
# Numerowanie tylko niepustych linii  
awk '{if (NF!=0) {print NR" : "$0} else print}' plik.in  
awk 'NF{$0=++a " : " $0};{print}' plik.in
```

```
# Liczenie linii (emulacja "wc -l")  
awk 'END{print NR}' plik.in
```

Wykorzystanie awk wewnątrz skryptu bashowego

```
zmienna=5
```

```
awk -v x=$zmienna 'BEGIN{print x+4}'
```

da w wyniku 9.

SED

sed jest *edytorem strumieniowym*

- służy do zaawansowanego *znajdź / podstaw*
- potrafi usuwać i dodawać tekst
- rozpoznaje wzorce zbudowane z RE
- działa z linii komend lub skryptu
- przetwarza plik wejściowy linia po linii

Podstawowe opcje wywołania:

`sed -n` nie drukuje przetwarzanych linii na stdout

`sed -quiet`

`sed -silent`

`sed -e 'polecenia'`

`sed -f skrypt.sed`

Podstawowe komendy:

`p` print (wyświetla na stdout)

`d` delete (usuwa)

`s` search / replace (zamienia wyrażenia)

`i` insert (wstawia tekst przed)

`a` append (wstawia tekst po)

`c` change (podmienia linię w buforze)

Bufory sed'a

- Linie pliku wejściowego wczytywane są do *bufora roboczego* (inaczej *bufora wzorca*).
- Istnieje możliwość skorzystania ze schowka — *bufora zapasowego* — w celu wykonania kopii przetwarzanej linii.

Learning by doing – przykłady:

- Kasowanie wszystkich linii (pusty output):

```
cat /etc/passwd | sed 'd'  
sed -e 'd' /etc/passwd
```

- Wyświetlanie wszystkiego: `cat /etc/passwd | sed 'p'`

```
root:x:0:0::/root:/bin/bash  
root:x:0:0::/root:/bin/bash  
bin:x:1:1:bin:/bin:  
bin:x:1:1:bin:/bin:  
daemon:x:2:2:daemon:/sbin:  
daemon:x:2:2:daemon:/sbin:
```

sed domyślnie wyświetla przetwarzane linie na ekran, stąd podwojenie wszystkich linii.

`sed -n -e 'p' /etc/passwd` nie będzie podwajało linii.

Adresowanie wierszy do obróbki:

- Skasowanie pierwszej linii: `sed '1d' plik`
- Skasowanie linii 1–5: `sed '1,5d' plik`
- `sed '9,1d' plik` skasuje tylko linię 9. bo plik jest przetwarzany po kolei
- Skasowanie co 3. wiersza (1., 4., 7.,...): `sed '1~3d' plik`
- Podstawianie: `sed 's/root/admin/' plik` zamieni **pierwsze wystąpienie** słowa *root* na *admin*
- Podstawianie: `sed 's/root/admin/g' plik` zamieni **każde wystąpienie** słowa *root* na *admin* (opcja global)
- Podstawienie: `sed 's:/root:/toor' plik` zamieni */root* na */toor* (zdefiniowany separator ciągów to :)

Znaczniki opcji podstawiania s/wzorzec/na-co/znacznik:

g	global – wszystkie wystąpienia
liczba	tylko wskazane wystąpienie
p	wypisuje dodatkowo linię którą obrobił
i	ignoruje wielkość liter
m	pozwała na użycie znaków ^ \$ we wzorcu

Przykład – txt2html.sed

```
#!/bin/sed -f
1 i\
<html>\
<head><title>Skonwertowany sedem</title></head>\
<body><pre>\

$ a\
</pre></body>
</html>
```

Polecenie 1i oznacza wstawienie czegoś **przed 1.** wierszem pliku, \$a wstawienie **za końcem pliku**.

Dopasowanie linii za pomocą RE podobnie jak w awk.

RE ograniczone ukośnikami `/.../`

`^` oznacza początek wiersza

`$` oznacza koniec wiersza

`.` oznacza pojedynczy znak

`*` oznacza 0 lub więcej wystąpień poprzedniego znaku (!)

`[abc]` oznacza a lub b, lub c

`[a-z]` zakres

Klasy znaków – uwzględniają lokalizacje, w tym polskie litery.

<code>[:alnum:]</code>	litery i cyfry
<code>[:alpha:]</code>	litery (alfabet)
<code>[:blank:]</code>	spacje i tabulatory
<code>[:cntrl:]</code>	znaki sterujące
<code>[:digit:]</code>	cyfry
<code>[:graph:]</code>	wszelkie widoczne znaki (bez znaków odstępu)
<code>[:lower:]</code>	małe litery
<code>[:print:]</code>	znaki drukowalne (bez sterujących)
<code>[:punct:]</code>	znaki interpunkcyjne
<code>[:space:]</code>	wszystkie znaki odstępów (?)
<code>[:upper:]</code>	wielkie litery
<code>[:xdigit:]</code>	znaki HEX: 0–9, a–f, A–F

Dopasowania – polecenia specjalne

`[[[:digit:]]\{3\}`

pasuje do ciągu **trzech** cyfr; wewnątrz skryptu można zrezygnować z ukośników przed nawiasami.

`&`

odwołanie się do dopasowanego tekstu, np.

`s/[[[:digit:]]\{3\}/\*&\*/g`

podmieni globalnie wszystkie liczby trzycyfrowe na te same liczby ale otoczone gwiazdkami.

Jeśli dopasowywane wyrażenie ujmie się w nawiasy, to można się później do niego odwoływać poprzez `\1`, `\2` etc., np.

`s/\([[:digit:]]\{3\}\)/\*\1\*/g`

robi to samo co poprzednie.

Inne polecenia

- g zastępuje bufor roboczy buforem zapasowym
- G dopisuje bufor zapasowy do roboczego,
oddzielając je znakiem nowej linii
- h zastępuje bufor zapasowy buforem roboczym
- H dopisuje bufor roboczy do zapasowego,
oddzielając je znakiem nowej linii
- N dodanie do obszaru roboczego `\n` i następnego wiersza tekstu
- X wymiana zawartości obszarów roboczego i zapasowego

i jeszcze kilka innych, m.in. etykiety, skoki do etykiet, zapisanie obszaru wzorca do pliku etc.

Przykłady

```
# Podwójny odstęp między wierszami  
sed G plik
```

```
# Konwersja znaków końca linii DOS --> Unix  
sed 's/.$//' plik # zakładając że każda linia zawiera CR/LF  
sed 's/^M$//' plik # znak CR (^M) wpisuje się jako Ctrl-v Ctrl-m
```

```
# Wyrównanie tekstu do lewej  
# usunięcie SPC i TAB z początku każdego wiersza  
# TAB = Ctrl-v Ctrl-t  
sed 's/^[ ^t]*//' plik
```

```
# J.w. na obu końcach wiersza  
sed 's/^[ ^t]*//;s/[ ^t]*$' plik # uwaga na miejsce wystąpienia *
```


System Linux – cz. 2

- Co to jest system operacyjny?
- Historia systemów Unix i Linux.
- Organizacje i społeczności rozwijające Linuksa.
- Jądro Linux.
- Instalacja.
- Podstawowa budowa systemu. System plików.
- Program rozruchowy i start systemu.
Poziomy uruchomieniowe, proces init.

System operacyjny to program pośredniczący pomiędzy użytkownikiem a komputerem.

- Tworzy maszyny wirtualne (abstrakcje) aby ukryć szczegóły warstwy sprzętowej (np. jednolity sposób dostępu do urządzeń zewnętrznych, współbieżne wykonywanie programów...).
- Zarządza zasobami komputera (CPU, RAM, pliki na dysku...).
- Dostarcza przyjazny interfejs użytkownika.

Przykłady:

minix, os/2, dos/windows, Unix (HP UX, Solaris, Digital Unix...), GNU, Linux, bsd (Free BSD, Net BSD, Open BSD)...

Historia w skrócie:

- 1969 Bell Laboratories UNIX PDP-7: prywatny projekt nowego systemu operacyjnego pisanego głównie w języku wysokiego poziomu (C) a nie w całości w assemblerze (przenośność kodu!)
- 1971 Unix Time-Sharing System 1–7 (1979), z klonami PWB/UNIX oraz 1BSD
- 1983 powstaje GNU
- 1984 powstał UNIX System III, 4.2BSD (Univ. of California – dołożyli obsługę sieci TCP/IP) i jego klon SunOS 1.0
- 1988 UNIX System V Release 4
- 1989 Unix Time-Sharing System 10
- 1991 BSD Net/2, Linux 0.0.1
- 2009 Solaris 10, NetBSD 5.0, FreeBSD 8.0, OpenBSD 4.5, GNU/Linux 2.6.31.x

- 2014 Oracle Solaris 11.1 – pierwszy Unix natywnie wspomagający chmury danych (www.oracle.com)
- 2025 Oracle Linux 9.5, Oracle Solaris 11.4
- 2025 NetBSD 10.1 (www.netbsd.org)
- 2025 FreeBSD 14.2 (www.freebsd.org)
- 2025 OpenBSD 7.6 – wykryto tylko dwie luki bezpieczeństwa w domyślnej instalacji OpenBSD na przestrzeni lat! (www.openbsd.org)
- 2025 Linux 6.xx (www.kernel.org)

Galeria sław:



Richard Stallman
(GNU, FSF)



Linus Torvalds
(Linux)



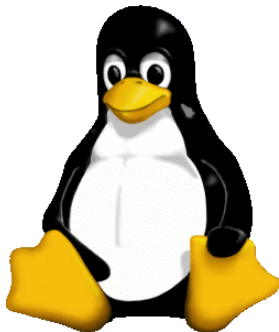
Ian Murdock
(Debian, † 2015)



Patrick Volkerding
(Slackware)

Stan na dzień dzisiejszy:

- Linux i GNU (GNU's Not Unix)
 - ▶ system GNU jest z powodzeniem rozwijany od 1984 r.
 - ▶ jądro Hurd nie jest (i pewnie nie będzie) gotowe do użytku
 - ▶ jądro Linux łączy się z systemem GNU tworząc GNU/Linux
- open source
- Free Software Foundation (FSF), licencja GNU GPL



Jądro Linuksa (kernel)

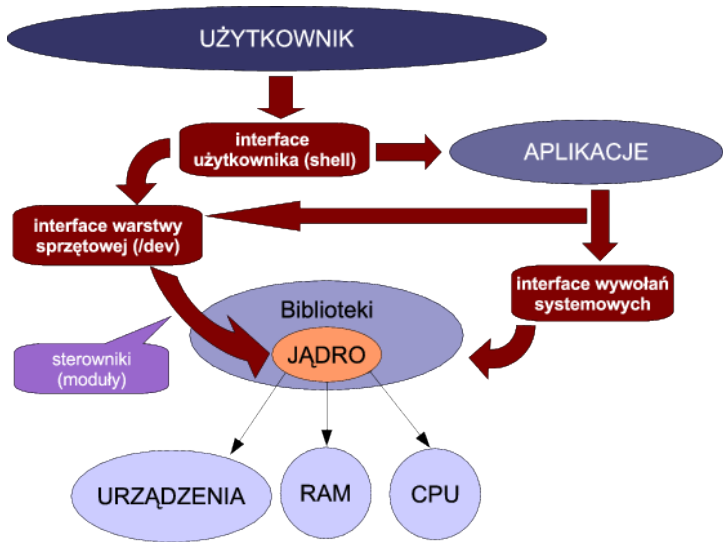
26 sierpnia 1991 **Linus Torvalds** wysłał list na grupę dyskusyjną `comp.os.minix` rozpoczynający się słowami:

"Właśnie piszę (wolny) system operacyjny (dla zabawy, nie będzie wielki i profesjonalny jak gnu) dla klonów AT 386(486)..."

CECHY:

- **modularne** (wcześniej monolityczne),
- prawdziwie **wielozadaniowe**,
- obsługuje dzielone biblioteki, pamięć wirtualną, wątki...
- kryptografia, sieć (firewall), gen. liczb losowych
- o otwartym kodzie źródłowym (język C) – **przenośne**

Schemat warstw i interface'ów w linuxsie



Dystrybucja Linuksa = kernel + oprogramowanie (nawet 15+ GB)
zobacz: distrowatch.com



Ciekawe propozycje na 2016 rok wg S.Bhartiya www.linux.com

- open SUSE Leap – mariaż openSUSE i SUSE Linux Enterprise
- elementary OS – tworzona przez grafików z nastawieniem na wygląd
- Chrome OS – do urządzeń sieciowych działających w chmurze
- Ubuntu MATE – dla laptopów, niskie wymagania
- Snappy Ubuntu Core – dla Internet of Things
- Steam OS – dla graczy
- Tails – dla wzmocnionego bezpieczeństwa, oparta o Tor

Instalacja Linuksa – typowe etapy

- boot z nośnika (cd/dvd, pendrive) lub podmontowanie partycji/dysku sieciowego (Preboot Execution Environment PXE, NetBoot w komputerach Mac)
- <partycjonowanie dysku twardego> (często robi to instalator)
- <formatowanie partycji> (często robi to instalator)
- uruchomienie instalatora (ekran tekstowy lub graficzny)
- przygotowanie partycji, jeśli jeszcze tego nie zrobiono
- definiowanie punktów montowania
- wybór pakietów i ich instalacja
- pre-konfigurowanie pakietów
- ustawienie domyślnie uruchamianych usług
- założenie konta root i konta użytkownika
- założenie i konfiguracja bootloadera
- reboot

Dystrybucje typu live (Knoppix, Slax,...)

Różne wykonanie, ta sama filozofia:

- cały system na płycie CD/DVD
- tworzy wirtualne dyski w pamięci RAM
- nie wymaga HDD ale może z niego korzystać, zapisywać i czytać dane z istniejących partycji
- kompresja systemu plików
- bardzo rozbudowany moduł samokonfiguracji systemu (wykrywanie i podłączanie urządzeń, konfiguracja sieci, drukarek...)
- blokuje napęd, chyba że mieści się cała w RAMie
- może służyć jako płyta ratunkowa, podręczny Linux, system ćwiczeniowy na pracowni komputerowej, system do kafejki internetowej...

- projekt Linux from scratch (LFS, www.linuxfromscratch.org)
- projekt Linux live (www.linux-live.org)
 - ▶ przygotowanie kernela (squashfs i aufs) lub wykorzystanie prekompilowanego gotowca
 - ▶ zbudowanie dystrybucji
 - ▶ zbudowanie jej obrazu
 - ▶ nagranie całości na cd/dvd/usb w trybie bootowalnym (el torito)

Przygotowanie i formatowanie partycji:

- system plików ext2/3/4, reiserfs, jfs, xfs – **właściciel, grupa, prawa dostępu**
- programy fdisk (dos/Linux), cfdisk (Linux), partition magic (dos), gparted (również GParted Live; Linux – graficzny)

NA TYM ETAPIE TWORZONA JEST PARTYCJA NA DYSKU TWARDYM

- w większości przypadków jest to jeden z etapów instalacji systemu!
- mke2fs wraz z aliasami mkfs.ext2, mkfs.ext3 i mkfs.ext4 tworzą nowy system plików typu ext2 lub ext3/ext4 (ext2 z journalingiem) na podanym urządzeniu

NA TYM ETAPIE PARTYCJA JEST FORMATOWANA POD WYBRANYM FS

- mkfs.reiserfs j.w. dla systemu plików reiserfs
- fsck sprawdza i naprawia znalezione błędy; musi działać na odmontowanym systemie plików
- tune2fs zmienia parametry już istniejącego systemu plików

```
[bash]# fdisk /dev/sda
```

Command (m for help): p

Disk /dev/sda: 120.0 GB, 120034123776 bytes

255 heads, 63 sectors/track, 14593 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	1	4742	38090083+	7	HPFS/NTFS
/dev/sda2		4743	14593	79128157+	5	Extended
/dev/sda5		4743	7296	20514973+	83	Linux
/dev/sda6		7297	12160	39070048+	83	Linux
/dev/sda7		12161	14471	18563076	83	Linux
/dev/sda8		14472	14593	979933+	82	Linux swap

Command (m for help): m

Command action

- a toggle a bootable flag
- b edit bsd disklabel
- c toggle the dos compatibility flag
- d delete a partition
- l list known partition types
- m print this menu
- n add a new partition
- o create a new empty DOS partition table
- p print the partition table
- q quit without saving changes
- s create a new empty Sun disklabel
- t change a partition's system id
- u change display/entry units
- v verify the partition table
- w write table to disk and exit
- x extra functionality (experts only)

Command (m for help): l

0	Empty	10	OPUS
1	FAT12	11	Hidden FAT12
2	XENIX root	12	Compaq diagnost
3	XENIX usr	14	Hidden FAT16 <3
4	FAT16 <32M	16	Hidden FAT16
5	Extended	17	Hidden HPFS/NTF
6	FAT16	18	AST SmartSleep
7	HPFS/NTFS	1b	Hidden W95 FAT3
8	AIX	1c	Hidden W95 FAT3
9	AIX bootable	1e	Hidden W95 FAT1
a	OS/2 Boot Manag	24	NEC DOS
b	W95 FAT32	39	Plan 9
c	W95 FAT32 (LBA)	3c	PartitionMagic
e	W95 FAT16 (LBA)	40	Venix 80286
f	W95 Ext'd (LBA)	41	PPC PReP Boot

42	SFS	70	DiskSecure Mult
4d	QNX4.x	75	PC/IX
4e	QNX4.x 2nd part	80	Old Minix
4f	QNX4.x 3rd part	81	Minix / old Lin
50	OnTrack DM	82	Linux swap
51	OnTrack DM6 Aux	83	Linux
52	CP/M	84	OS/2 hidden C:
53	OnTrack DM6 Aux	85	Linux extended
54	OnTrackDM6	86	NTFS volume set
55	EZ-Drive	87	NTFS volume set
56	Golden Bow	88	Linux plaintext
5c	Priam Edisk	8e	Linux LVM
61	SpeedStor	93	Amoeba
63	GNU HURD or Sys	94	Amoeba BBT
64	Novell Netware	9f	BSD/OS
65	Novell Netware	a0	IBM Thinkpad hi

be	Solaris boot	a7	NeXTSTEP
bf	Solaris	a8	Darwin UFS
c1	DRDOS/sec (FAT-	a9	NetBSD
c4	DRDOS/sec (FAT-	ab	Darwin boot
c6	DRDOS/sec (FAT-	b7	BSDI fs
c7	Syrinx	b8	BSDI swap
da	Non-FS data	bb	Boot Wizard hid
db	CP/M / CTOS / .	ee	EFI GPT
de	Dell Utility	ef	EFI (FAT-12/16/
df	BootIt	f0	Linux/PA-RISC b
e1	DOS access	f1	SpeedStor
e3	DOS R/O	f4	SpeedStor
e4	SpeedStor	f2	DOS secondary
eb	BeOS fs	fd	Linux raid auto
a5	FreeBSD	fe	LANstep
a6	OpenBSD	ff	BBT

Podział dysku twardego: ile partycji potrzeba?

- Minimalnie jedna: główne drzewo katalogów / (typ 83)
- Zazwyczaj przynajmniej dwie: / i swap (typ 82)
- Wygoda sugeruje wydzielenie co najmniej:

/ /home /usr/local

- Bezpieczeństwo dokłada do tego:

/root /var /etc /tmp

Jak duży swap?

- zwyczajowo $\text{swap} = 2 \times \text{RAM}$ przy niewielkiej ($\sim 64\text{MB}$) ilości RAMu
- obecnie swap w zależności od potrzeb
- jeśli chcemy uruchomić hibernację w urządzeniach mobilnych, swap minimalnie w ilości RAM

Systemy plików

`en.wikipedia.org/wiki/Comparison_of_file_systems`

- istnieje kilkadziesiąt różnych systemów plików
- część z nich można założyć na partycjach typu Linux
- od wyboru właściwego fs zależy szybkość działania systemu, trwałość dysku twardego, podatność na utratę danych etc.

System plików ext2

- wprowadzony w 1993 r. (Rémy Card)
- max. plik 16 GB (32b) / 2 TB (64b)
- max. partycja 2 TB (32b) / 32 TB (64b)
- wymaga okresowego sprawdzania i defragmentacji
- wciąż polecany dla pamięci przenośnych

System plików ext3

- wprowadzony w 2001 r. (Stephen Tweedie), Linux Kernel 2.4.15
- obsługuje księgowanie (journaling) – część partycji jest wydzielona na zapis zmian dokonywanych w systemie plików
- księgowanie typu journal – metadane i zawartość są przechowywane w księdze
- księgowanie typu ordered – tylko metadane są przechowywane w księdze, zapisywane tam po zapisie danych na dysku (domyślne ustawienie)
- księgowanie typu writeback – tylko metadane są przechowywane w księdze, zapisywane tam po lub przed zapisem danych na dysku
- max. plik 16 GB (32b) / 2 TB (64b)
- max. liczba podkatalogów 32000
- max. partycja 2 TB (32b) / 32 TB (64b)
- wymaga okresowego sprawdzania i defragmentacji
- można przekonwertować ext2 ↔ ext3

System plików ext4

- wprowadzony w 2008 r., Linux Kernel 2.6.19
- max. plik 16 GB (32b) / 16 TB (64b)
- max. liczba podkatalogów 64000
- max. partycja 1 EB (exabyte) = 1024 PB (petabyte) = $1024 \cdot 1024$ TB
- można zamontować ext3 jako ext4 bez konwersji
- wymaga okresowego sprawdzania i defragmentacji
- nowe opcje stabilizujące system: alokacja wielu bloków, alokacja opóźniona, suma kontrolna księgi, szybkie fsck, możliwość wyłączenia księgi...

Przykłady:

1) sformatowanie partycji /dev/sda1 pod ext2

```
[bash]# mke2fs /dev/sda1
```

2) sformatowanie partycji /dev/sda1 pod ext3

```
[bash]# mke2fs -j /dev/sda1
```

lub

```
[bash]# mkfs.ext3 /dev/sda1
```

3) sformatowanie partycji /dev/sda1 pod ext4

```
[bash]# mke2fs -t ext4 /dev/sda1
```

lub

```
[bash]# mkfs.ext4 /dev/sda1
```

4) konwersja ext2 → ext3

```
[bash]# tune2fs -j /dev/sda1
```

UWAGA: formalnie konwersję można przeprowadzić na zamontowanej partycji, ale dla pewności lepiej jest ją najpierw odmontować (`umount /dev/sda1`); oczywiście nie można odmontować głównego systemu plików na działającym systemie

5) konwersja ext3 → ext4

```
[bash]# umount /dev/sda1
```

```
[bash]# tune2fs -O extents,uninit_bg,dir_index /dev/sda1
```

```
[bash]# e2fsck -pf /dev/sda1
```

```
[bash]# mount /dev/sda1 /home
```

UWAGA

Wszelkie operacje na systemie plików są obarczone ryzykiem, więc zawsze należy się liczyć z możliwością utraty danych!

System plików ReiserFS (Reiser3)

- wprowadzony w 2001 r. (zespół Namesys kierowany przez Hansa Reisera), Linux Kernel 2.4.1
- pierwszy fs z księgowaniem w linuksie
- tail packing – potrafi upychać końcówki plików tak, aby skuteczniej wypełniały pełne bloki – efektywna obsługa małych plików (cache, kolejki)
- online resizing – można powiększyć przestrzeń działającego fs o niewykorzystane partycje lub urządzenia (zmniejszenie wymaga odmontowania fs)
- 28.01.2038 – po tej dacie fs nie będzie poprawnie tworzył znaków czasu dla plików
- naprawa i defragmentacja automatyczna, nie wymaga okresowego sprawdzania
- bardzo szybki i skuteczny sposób odzyskiwania danych po awarii

Ze względów polityczno–moralnych od 2008 r. Reiser3 nie jest oficjalnie wspierany przez Linux kernel. Jego nowsza wersja Reiser4 nie weszła do źródeł kernela (patrz: sourceforge.net/projects/reiser4/, reiser4.wiki.kernel.org).

System plików Reiser4

- różne testy pokazują, że jest 10–15x szybszy od ext3 w kwestii niewielkich plików
- przy innych operacjach potrafi być sporo wolniejszy
- skuteczniejsze tail packing
- skuteczniejsze księgowanie
- opóźniona alokacja (allocate on flush)
- wspomaga wtyczki zapewniające m.in. kompresję i kodowanie partycji

Najnowszy stabilny: system plików btrfs

- tworzony przy współpracy Oracle, RedHat, Fujitsu, Intel, SuSE, Strato i innych
- max. partycja $16 \text{ EB} = 2^{64} \text{ B}$
- efektywne przechowywanie małych plików
- efektywne indeksowanie katalogów
- dynamiczna alokacja węzłów (inode)
- podpartycje (internal fs roots)
- sumy kontrolne danych (crc32c)
- kompresja partycji (zlib, lzo)
- RAID
- natywna obsługa SSD
- efektywne przyrostowe kopie zapasowe
- defragmentacja online
- online/offline naprawa błędów
- konwersja z ext3/ext4

Jaki system plików wybrać?

- dla większości zastosowań najlepszą obecnie opcją to ext4
- można użyć również jfs (64b system IBM z księgowaniem, przystosowany do dużych dysków)
- można użyć również xfs (64b system Silicon Graphics z księgowaniem)
- jeśli nie zależy nam na najnowszych rozwiązaniach, stabilny i sprawdzony jest też Reiser3, ale od kilku lat nie był on poprawiany, za wyjątkiem wykrytych luk bezpieczeństwa
- dla urządzeń przenośnych zostaje ext2, ew. vfat, jeśli zależy nam na kompatybilności

W obecnej chwili sterowniki w linuxie stabilnie obsługują wszystkie główne fs z różnymi stronami kodowymi. Ostatnio zapis na partycje ntfs został uznany za w miarę bezpieczny (sterownik ntfs3g).

Atrybuty plików (chmod):

d rwx rwx rwx

typ pliku, prawa dostępu dla właściciela, grupy i pozostałych

–	brak lub zwykły plik	r w x	read (4), write (2), execute (1)
d	katalog	s	suid lub gid
l	link	X	x jeśli katalog lub już miał x
c	węzeł (nod)	t	sticky bit
s	gniazdo (socket)		

Przykłady:

```
chmod 644 plik ustawia prawa -rw-r-r-
```

```
chmod u+rw plik ; chmod g+r plik ; chmod o+r plik
```

```
chmod 755 katalog ustawia prawa drwxr-xr-x
```

```
chmod u+rwx plik ; chmod g+rx plik ; chmod o+rx plik
```

Linki:

- twarde – kopia powiązana z oryginałem, działa jak osobny plik
- symboliczne – dowiązanie do pliku; po skasowaniu linku nic się nie dzieje, po skasowaniu oryginału link pozostaje pusty (błędny)

Składnia:

```
ln [OPTION]... [-T] TARGET LINK_NAME
```

```
ln [OPTION]... TARGET
```

```
ln [OPTION]... TARGET... DIRECTORY
```

```
ln [OPTION]... -t DIRECTORY TARGET...
```


Podstawowe opcje polecenia `ln`:

`-i`, `-interactive`

`-s`, `-symbolic`

`-b`, `-backup`

`-t`, `-target-directory`

`-v`, `-verbose`

Linki symboliczne („miękkie”) wymagają opcji `-s`

[bash] `# ln -s -t zzz /var/log/* ; ls -gG zzz`

```
lrwxrwxrwx 1 14 2008-02-19 21:52 acpid -> /var/log/acpid
```

```
lrwxrwxrwx 1 13 2008-02-19 21:52 bttmp -> /var/log/btmp
```

```
lrwxrwxrwx 1 13 2008-02-19 21:52 cron -> /var/log/cron
```

```
lrwxrwxrwx 1 14 2008-02-19 21:52 debug -> /var/log/debug
```

```
lrwxrwxrwx 1 14 2008-02-19 21:52 dmesg -> /var/log/dmesg
```

```
lrwxrwxrwx 1 16 2008-02-19 21:52 faillog -> /var/log/faillog
```

Linux Standard Base

www.linuxfoundation.org/collaborate/workgroups/lsb

- w chwili obecnej jest dostępna wersja LSB 5.0 Beta 1
- próba ustandaryzowania Linuksa
- pomoc dla programistów
- porządek w systemie plików
- narzędzia do sprawdzania dystrybucji (Distribution Checker), SDK, audytu aplikacji (Application Checker)

Certyfikacja dystrybucji:

www.linuxbase.org/lsb-cert/productdir.php?by_lsb

m.in.: Mandriva Ent. Server 5, Oracle Linux 5/6, Red Flag 6, Red Hat Ent. 5.6/6.0, SuSE 11.0, Ubuntu 8.04/9.04...

Główny system plików:

drwxr-xr-x	2	root	bin	2232	2007-03-01	20:50	bin/
drwxr-xr-x	2	root	root	280	2006-12-15	20:00	boot/
drwxr-xr-x	15	root	root	62592	2007-10-08	14:09	dev/
drwxr-xr-x	27	root	root	3200	2007-10-08	14:09	etc/
drwxr-xr-x	6	root	root	120	2007-03-03	18:02	home/
drwxr-xr-x	4	root	root	2688	2006-10-02	21:11	lib/
drwxr-xr-x	9	root	root	216	2002-03-16	08:34	mnt/
drwxr-xr-x	8	root	root	240	2006-12-29	14:57	opt/
dr-xr-xr-x	69	root	root	0	2007-10-08	16:09	proc/
drwx--x---	24	root	root	1160	2007-07-27	11:10	root/
drwxr-xr-x	2	root	bin	3672	2007-03-01	20:50	sbin/
drwxr-xr-x	11	root	root	0	2007-10-08	16:09	sys/
drwxrwxrwt	25	root	root	1512	2007-10-08	14:56	tmp/
drwxr-xr-x	17	root	root	520	2002-02-13	06:44	usr/
drwxr-xr-x	11	root	root	376	2006-09-16	22:07	var/

- Katalog `/bin` zawiera binaria systemowe – podstawowe komendy i programy takie jak `bash`, `cat`, `dd`, `ls`, `ln`...
- Katalog `/boot` zawiera zwyczajowo jądro systemu, obraz `ram-disk`, system map oraz katalog z konfiguracją programu rozruchowego grub.
- Katalog `/dev` zawiera węzły do urządzeń; jest to interface pomiędzy użytkownikiem a jądrem systemu, np. `/dev/mouse`, `/dev/sda1`...
- Katalog `/etc` zawiera wszystkie pliki konfiguracyjne systemu, łącznie ze skryptami startowymi i konfiguracją sprzętową; większość programów swoje pliki konfiguracyjne również umieszcza tutaj.
Wg nowych standardów, podstawowa część konfiguracji niektórych aplikacji umieszczona jest w `/lib/` (`modprobe`, `udev`). Pliki z `/etc` zawsze mają pierwszeństwo przed tymi z `/lib`.
- Katalog `/home` zawiera katalogi domowe i pliki użytkowników.
- Katalog `/lib` zawiera podstawowe biblioteki systemowe, w tym `glibc`, oraz moduły (sterowniki) urządzeń.

- Katalog `/mnt` zawiera punkty montowania stałych zdalnych systemów plików (np. dodatkowe partycje, dyski sieciowe). Partycje te zazwyczaj są automatycznie montowane podczas startu systemu.
- Katalog `/media` to punkty montowania tymczasowych partycji, czyli urządzeń pamięci masowej tj. pendrive, cdrom/dvdrom, kamera, aparat, tel. komórkowy itp. Często jest obsługiwany przez system automontowania. UWAGA: *Najnowsza wersja `udisks2` używa `/run/media/<user>`.*
- Katalog `/opt` jest miejscem gdzie zwyczajowo umieszcza się programy nie będące częścią dystrybucji, rozprowadzane jako gotowe prekompilaty (np. firefox, open office).
- Katalogi `/proc` i `/sys` to kolejne interface'y pomiędzy jądrem a użytkownikiem, zawierające informacje o stanie systemu. Obecnie `procfs` jest zarzucany i całość informacji przenoszona jest do znacznie bardziej rozbudowanego systemu `sysfs`.

- Katalog `/root` to katalog domowy administratora.
- Katalog `/sbin` zawiera binaria programów z których powinien korzystać tylko administrator, np. `adduser`, `ifconfig`...
- Katalog `/tmp` przechowuje pliki tymczasowe wszystkich użytkowników i aplikacji działających w systemie. Ulega samoczynnie zaśmiecaniu, dlatego często w cronie umieszcza się zlecenie jego okresowego czyszczenia.
- Katalog `/usr` zawiera całe oprogramowanie dystrybucyjne. `/usr/local` zawierać będzie programy kompilowane samodzielnie przez administratora. `/usr/src` zawiera źródła jądra.
- Katalog `/var` zawiera różne wpisy potrzebne do działania systemu, w tym PIDy (`/var/lock`), kolejki poczty, kolejki drukowania (`/var/spool`), logi systemowe (`/var/log`)...

Start systemu

```
[bash]# ls -la /boot
```

```
drwxr-xr-x  2 root root    280 ./
drwxr-xr-x 18 root root    408 ../
-rw-r--r--  1 root root    512 boot.0800
-rw-r--r--  1 root root     16 boot.message
lrwxrwxrwx  1 root root     30 config -> config-3.10.17-smp
-rw-r--r--  1 root root 141822 config-3.10.17-smp
-rwxr-xr-x  1 root root 217646 elilo-ia32.efi
-rwxr-xr-x  1 root root 250510 elilo-x86_64.efi
-rw-r--r--  1 root root 3916355 initrd.gz
-rw-----  1 root root   35840 map
lrwxrwxrwx  1 root root     34 System.map -> System.map-3.10.17-smp
-rw-r--r--  1 root root 1760315 System.map-3.10.17-smp
-rw-r--r--  1 root root   33192 tuxlogo.bmp
lrwxrwxrwx  1 root root     31 vmlinuz -> vmlinuz-3.10.17-smp
-rw-r--r--  1 root root 3272208 vmlinuz-3.10.17-smp
```

Program rozruchowy (bootloader): **LILO** lub **GRUB** wczytuje jądro systemu do pamięci.

Znaczenie plików z katalogu boot

- `System.map` – przechowuje mapę symboli i adresów dla jądra. To samo znajduje się w `/proc/kallsyms`. Zobacz: rlworkman.net/system.map/
- `vmlinuz` – jądro systemu (również `linux`, `kernel`, `jajko-miszczka-2.6.19` i inne)
- `initrd.gz` – spakowany obraz ramdysku
- `map` – wskazuje lilo gdzie znajduje się obraz rozruchowy, jest tworzony z pliku `/etc/lilo.conf` komendą `lilo`
- `boot.0300`, `boot.0800` – kopia zapasowa MBR; `0300=/dev/hda`, `0301=/dev/hda1`, `0800=/dev/sda...` Odzyskanie MBR komendą `dd`
- `elilo-ia32.efi`, `elilo-x86_64.efi` – pliki elilo umożliwiające rozruch w standardzie UEFI
- `grub/` – katalog z konfiguracją grub

To ja mam jakiś bootloader na dysku?

Zobacz: en.wikipedia.org/wiki/Comparison_of_boot_loaders

- każdy system operacyjny musi wykorzystywać jakiś program, który wgrałby do pamięci podstawowe sterowniki, umożliwiające załadowanie os
- w windows bootloader jest częściowo ukryty, ale też można znaleźć na dysku plik konfiguracyjny do niego
- można też przerwać ładowanie systemu (F8) i przejść do konsoli, gdzie dostępne są różne opcje (uruchom normalnie, uruchom w trybie awaryjnym etc.)
- w linuxie bootloader można sobie wybrać i oprogramować zgodnie z własną wolą
- instalator dystrybucji najczęściej robi to za nas

LILO

- przez długi czas wiodący bootloader w linuxie; obecnie (2007 r.) już nie rozwijany i zastąpiony prawie wszędzie przez **GRUBa**

Prace nad lilo wznowiono w czerwcu 2010 i zakończono około XII.2015 (lilo 24.2). Gałąź elilo zarzucono w 2014; wspomaga ona UEFI, RAID i GPT.

- obsługuje do 16 zdefiniowanych obrazów startowych
- może być wgrane w MBR lub sektor startowy dowolnej partycji (wtedy MBR musi zawierać coś co załaduje LILO)
- polega na BIOSie aby uzyskać dostęp do HDD – stare BIOSy nie pozwalały na adresowanie powyżej 1023 bloku dwóch pierwszych dysków; nowe (opcja LBA32, *logical block addressing*) dają dostęp do całości
- pokazuje graficzne, semi-graficzne lub tekstowe menu
- plik konfiguracyjny `/etc/lilo.conf` wgrywany jednorazowo (komendą `/sbin/lilo`); przy każdej zmianie konfiguracji konieczne jest uruchomienie `/sbin/lilo`

```
#####| root = /dev/hdb1
# LILO config '/etc/lilo.conf'. | image = /boot/vmlinuz-2.6.17mdk
lba32 | label = "mandrake 10.2"
boot = /dev/hda |
map = /boot/System.map | root = /dev/hdb2
delay = 40 | vga = ask
vga = normal | other = /dev/hda3
default=linux | label = windows
|
root = /dev/hda1 |
read-only |
image = /boot/kernel-2.6.21.5 |
label = "linux" |
append = "hdc=ide-scsi" |
image = /boot/probny-2.4.18 |
label = "slack_proba" |
```

GRUB

- zgodny ze standardem multiboot
- obsługuje wiele systemów operacyjnych bezpośrednio bądź pośrednio (mechanizm chain-loading)
- możliwość bootowania przez sieć
- możliwość działania zdalnego przez wirtualny terminal
- jednolite odwołanie do urządzeń (hdd, fd,...)
- czyta plik konfiguracyjny przy każdym starcie
- pokazuje menu (i ew. obrazek w tle)
- posiada wbudowaną powłokę wzorowaną na bash

```
#####  
# GRUB configuration file '/boot/grub/menu.lst'.  
# generated by 'grubconfig'.  Sun Jun 17 16:36:11 2007  
  
# timeout 30  
# splash-image=/boot/grub/splash/slacks.xpm.gz  
  
color light-gray/black light-green/black  
  
title slackware kernel 2.6  
root (hd0,4)  
kernel /boot/kernel-2.6.17.13 root=/dev/sda5 ro vga=773  
  
title slackware kernel 2.4 nofb  
root (hd0,4)  
kernel /boot/vmlinuz-2.4.33.3 root=/dev/sda5 ro vga=normal
```

```
title windows
rootnoverify (hd0,0)
makeactive
chainloader +1
```

```
title slackware 12.0
root (hd0,6)
kernel (hd0,6)/boot/vmlinuz root=/dev/sda7 ro vga=773
#####
```

Obrazy startowe grub'a

- stage1 – obraz ładujący stage1.5 lub stage2, siedzi w MBR lub w boot sektorze (dokładnie 512 bajtów)
- stage1.5 – opcjonalny, ładowany przez stage1, potrafi obsłużyć jeden konkretny system plików (np. reiserfs_stage1_5), co pozwala przesunąć stage2 na partycję tego typu
- stage2 – główny obraz grub'a
- stage2_eltorito – jak stage2 tylko bootowanie z CD/DVD
- nbgrub – j.w. ale zgodne ze standardem Network Image Proposal (używane np. przez Etherboot)
- pxegrub – j.w. ale zgodne z Preboot Execution Environment (używane przez wiele Netboot ROM'ów)

Konfiguracja grub'a: `/boot/grub/menu.lst`. Komendy w konsoli to podzbiór komend rozpoznawanych przez plik konfiguracyjny. Niektórych z nich nie można użyć w pliku konfiguracyjnym.

<code>default 0</code>	startuj domyślnie z 1. wpisu (saved)
<code>timeout 10</code>	czekaj 10 sek.
<code>fallback 1</code>	w razie niepowodzenia przejdź do wpisu 2
<code>color</code>	zmienia kolory menu
<code>hiddenmenu</code>	ukrywa menu (ESC przywraca)
<code>title</code>	etykieta zaczynająca wpis
<code>password -md5 PASS</code>	ustawia hasło (szyfrowane)
<code>password -md5 PASS /boot/grub/root.menu.lst</code>	j.w. ale ładuje dodatkowy plik konfiguracyjny

lock	blokuje dalsze działanie do czasu wpisania hasła
bootp	bootowanie przez sieć (protokół bootp)
chainloader plik	wczyta plik jako chain-loader'a
chainloader +1	poszuka chain-loader'a w 1. sekt. dysku
initrd plik	ładuje linuksowy ramdisk
kernel plik opcje	wskazuje na jądro
makeactive	ustawia partycję jako root device
root dev	ustawia i montuje główną partycję
rootnoverify dev	j.w. bez montowania
savedefault	zapisuje bieżące jako domyślne
setup	instaluje grub'a na dysku
halt	wyłącza komputer (opcja -no-apm)
help -all	dostępne komendy

md5crypt	pozwala zaszyfrować hasło
device	pozwala przypisać plik lub urządzenie do urządzenia
np. device (fd0)	/floppy-image
np. device (hd0)	/dev/sd0
partnew	tworzy nową partycję główną
np. partnew (hd0,2)	83 1200 1700
parttype	zmienia typ partycji
setkey	zmienia mapowanie klawiatury
terminal	zmienia typ terminala
terminfo	j.w. dla terminali innych niż vt-100
reboot	restart komputera
read addr	wyświetla zawartość komórki pamięci

(un)hide	ukrycie/odkrycie partycji
boot	po wpisaniu opcji – boot
cat /etc/fstab	pozwala obejrzeć plik
cmp p1 p2	porównanie plików p1 i p2
configfile plik	wczytuje plik konfiguracyjny
debug	wł./wył. debugowanie
displayapm	wyświetla info o APM z BIOSu
displaymem	wyświetla info o pamięci
find ścieżka	sprawdza istnienie pliku na wszystkich partycjach
fstest	tryb sprawdzania systemu plików
geometry dysk	info o geometrii dysku
quit	wyjscie z powłoki grub'a

Oznaczenia urządzeń:

(hd0,0)	pierwszy hdd, pierwsza partycja
(hdi,j)	(i+1)-szy hdd, (j+1)-sza partycja
(hd0,2,i)	(i+1)-sza partycja, 3. kawałek 1. dysku (BSD)
(fd0)	pierwsza stacja dyskietek
(cd0)	cdrom

Rozkład i typy partycji zależą od systemu plików!

Można używać kodów liczbowych (dziesiętnych lub hex) z BIOSu,
np. (hd0) = (0x80) = (128)

Składnia polecenia `color`:

```
color normal_fg/normal_bg highlight_fg/highlight_bg
```

Dostępne kolory:

black, blue, green, cyan, red, magenta, brown, light-grey.

Tylko jako kolory czcionek:

dark-grey, light-blue, light-green, light-cyan, light-red, light-magenta, yellow, white.

Dodanie `blink`- do koloru czcionki powoduje mruganie.

Każdy wpis menu może mieć swój zestaw kolorów.

Minimalny wpis dla Linuksa:

```
# For booting GNU/Linux
title   GNU/Linux
kernel (hd1,0)/vmlinuz root=/dev/hdb1
```

Wpis dla windowsów:

```
# For booting Windows NT or Windows95
title Windows NT / Windows 95 boot menu
root      (hd0,0)
makeactive
chainloader +1
# For loading DOS if Windows NT is installed
# chainload /bootsect.dos
```

Wpis specjalny: instalacja grub'a na dysku

```
# For installing GRUB into the hard disk
title Install GRUB into the hard disk
root      (hd0,0)
setup     (hd0)
```

Obecnie GRUB (0.97) jest projektem zamrożonym i nazywanym GRUB Legacy. Od 2007 część dystrybucji zaczęła używać **GRUB2**:

- nowa składnia pliku konfiguracyjnego `/boot/grub/grub.cfg`; wewnętrzny język skryptowy
- `grub-mkconfig` tworzy automatycznie plik konfiguracyjny, wykrywając nowe wersje jądra założone w systemie
- nowy schemat nazywania urządzeń; numery partycji zaczynają się od 1 (ale nr-y urządzeń dalej od 0)
- ulepszony system wyszukiwania plików na dyskach, lepsza obsługa wielu systemów plików (np. ext4, ntfs, hfs+)
- dostępny na inne platformy niż PC BIOS: PC EFI, PC coreboot, PowerPC, SPARC i MIPS Lemote Yeeloong
- bezpośrednia obsługa LVM i RAID
- graficzne menu i terminal
- języki narodowe w menu i terminalu
- inna filozofia obrazów startowych, dynamicznie ładowane moduły

Obrazy startowe GRUB2

- `boot.img` – 512 B kodu do umieszczenia w MBR lub sektorze rozruchowym partycji; `grub-setup` wgrywa w `boot.img` położenie pierwszego sektora `core.img`
- `diskboot.img` – 1.sektor `core.img`, jeśli ten znajduje się na hdd; czyta resztę `core.img` i ładuje jądro systemu do pamięci
- `cdboot.img` – j.w. dla bootowania z CD
- `pxeboot.img` – j.w. dla bootowania przez sieć
- `lnxboot.img` – j.w. jeśli chcemy, aby `core.img` imitował jądro Linuksa, przez co można go załadować za pomocą LILO
- `kernel.img` – główny kod GRUB2, wkodowany w `core.img`
- `core.img` – tworzony z `kernel.img`, któregoś z nagłówek i dowolnej listy modułów (budowa modułarna pozwala sterować jego wielkością)

Nazwy urządzeń zawierają typ podziału na partycje, np.

(fd0)	cała dyskietka
(hd0,msdos2)	1. hdd, 2. partycja (rzeczywista)
(hd0,msdos5)	1. hdd, 1. partycja (logiczna)
(hd1,msdos1,bsd1)	2. hdd, 1. kawałek, 1. partycja (podział BSD)

UWAGA: GRUB2 zawsze numeruje partycje logiczne zaczynając od numeru 5, niezależnie ile partycji rzeczywistych istnieje.

Jeden z wielu wpisów w `/boot/grub/grub.cfg` (linia „menuentry” nie powinna być złamana)

```
#
# DO NOT EDIT THIS FILE
#
# It is automatically generated by /usr/sbin/grub-mkconfig using
# templates from /etc/grub.d and settings from /etc/default/grub
#
menuentry 'Linux 2.6.32-24-generic' --class ubuntu
    --class gnu-linux --class gnu --class os {
    recordfail
    insmod ext2
    set root='(hd1,5)'
    search --no-floppy --fs-uuid --set daa61b41-6397-4240-89cd
    echo      'Wczytywanie systemu Linux 2.6.32-24-generic...'
    linux    /vmlinuz-2.6.32-24-generic root=UUID=b7db94b2-f4c ro
    echo      'Wczytywanie początkowego dysku RAM...'
    initrd   /initrd.img-2.6.32-24-generic
}
```

```
menuentry "Microsoft Windows XP Professional (on /dev/sdb1)" {  
  insmod ntfs  
  set root='(hd1,1)'  
  search --no-floppy --fs-uuid --set 1a3cd2033cd1da39  
  drivemap -s (hd0) ${root}  
  chainloader +1  
}
```

Obecnie wykorzystywane standardy: GPT, EFI/UEFI.

- GPT (Globally unique identifier Partition Table) – identyfikator partycji wykorzystywany zamiast rozruchu z MBR; jest częścią specyfikacji EFI (w linuxie użyj parted lub gdisk, jeśli chcesz wykorzystać mechanizm GPT)
- EFI/UEFI (Unified Extensible Firmware Interface) – to inny sposób dostępu do urządzenia (bez pośrednictwa starych mechanizmów BIOS); oparty na zakodowanym w firmware zestawie programików („graficzny bios”, obsługa myszy i grafiki), które pozwalają na obsługę sprzętu (wykrywanie urządzeń podpiętych w różnych portach, monitoring parametrów sprzętu, upgrade firmware’u, start systemu)

Na poziomie UEFI można zablokować dostęp do sprzętu. Microsoft wymusił używanie płyt głównych z obsługą UEFI dla komputerów z Windows 8, potencjalnie mając możliwość zablokowania uruchomienia na tych płytach Linuksa. W chwili obecnej Windows 8+ (64b) jest jedynym systemem wymagającym UEFI. Nowe wersje bootloaderów w linuxie obsługują ten standard bootowania.

Secure Boot w UEFI

- aby zapobiec wirusom rezydującym w MBR i uaktywniającym się na wczesnym etapie rozruchu, mechanizm Secure Boot może sprawdzać podpisy cyfrowe wszystkich procedur EFI jakie uruchamia
- Microsoft wymusił umieszczenie swoich kluczy (podpisów) wkodowanych w płyty główne, jako wymóg do uruchomienia na tym sprzęcie Windows 8
- w teorii, jeśli klucze innych bootloaderów nie zostałyby wkodowane, jedynym bootloaderem mogącym działać na tym sprzęcie byłby ten w Windows 8
- ostatecznie Microsoft we współpracy z Verisign udostępnił klucz, którym każdy może podpisać własne oprogramowanie (koszt 99\$) i zniósł wymóg włączonego Secure Boot na komputerach x86 i x86-64. Urządzenia typu ARM z Windows 8 dalej muszą mieć Secure Boot włączone na stałe

Dobre artykuły napisał na ten temat Roderick W. Smith

www.rodsbooks.com

Jak uruchomić UEFI w linuxsie?

- Często wymogiem jest wyłączenie opcji Secure Boot oraz Fast Boot w UEFI płyty głównej.
- Konieczne jest utworzenie partycji, nazywanej esp (EFI System Partition): rozmiar ok. 100MB, typ System EFI (EF00) jeśli używamy GPT lub EFI (FAT-12/16/32) jeśli chcemy użyć MBR. Zalecany jest system GPT. Ważne: system plików to fat32!
- Partycję montujemy zwyczajowo w `/boot/efi/`.

Każdy system tworzy na tej partycji swoje katalogi z plikami startowymi. Powinny tam się znaleźć m.in. jądro systemu, obraz bootloadera, czasami konfiguracja bootloadera oraz dodatkowe pliki. Każdy wpis, po wgraniu odpowiednich plików, należy zarejestrować w pamięci NVRAM systemu UEFI.

Obsługa systemu bootowania na poziomie UEFI: `efibootmgr`.

Sprawdzenie zarejestrowanych wpisów

```
efibootmgr -v
```

```
BootCurrent: 0002
```

```
Timeout: 3 seconds
```

```
BootOrder: 0001,0003,0002,0000
```

```
Boot0000* CD/DVD Drive    BIOS(3,0,00)
```

```
Boot0001* Hard Drive      BIOS(2,0,00)
```

```
Boot0002* Gentoo          HD(1,800,61800,6d98f360)
```

```
File(\EFI\boot\bootx64.efi)
```

```
Boot0003* Hard Drive      BIOS(2,0,00)P0: ST1500DM003-9YN16G
```


Zmiana kolejności bootowania

```
efibootmgr -o 0002,0000,0003,0001
```

Usunięcie wpisu z pamięci UEFI

```
efibootmgr -b 0001 -B
```

Stworzenie nowego wpisu o nazwie Grub i obrazie grubx64.efi

```
efibootmgr -c -L "Grub" -l '\EFI\Grub\grubx64.efi'
```

Program `efibootmgr` nie zmienia plików na partycji `esp`. Jest to tylko rejestrowanie i kasowanie odpowiednich wpisów w bazie systemu UEFI. Podobne narzędzie w Windows nazywa się `bcdedit.exe`.

Żeby uruchomić system potrzebny jest bootloader zgodny z UEFI.

ELILO

Jest to wersja lilo obsługująca UEFI. W Slackware jest skrypt `eliloconfig`, który instaluje w esp znalezione jądra.

Po instalacji, w `/boot/efi/EFI/Slackware` znajdują się pliki

<code>elilo.conf</code>	konfiguracja
<code>elilo.efi</code>	obraz bootowania
<code>vmlinuz</code>	jądro systemu + ew. ramdisk

Niektóre opcje elilo, które pojawią się w `elilo.conf`:

<code>timeout=200</code>	# opóźnienie w trybie interaktywnym
<code>delay=200</code>	# opóźnienie w trybie nieinteraktywnym
<code>prompt</code>	# włącza tryb interaktywny
<code>chooser=simple</code>	# tryb tekstowy wyboru opcji bootowania
<code>chooser=textmenu</code>	# tabela wyboru opcji bootowania
<code>message=plik</code>	# wygląd tabelki

Przykładowy wpis dla Linuksa

```
image=vmlinuz  
label=slackware  
description="Slackware64 14.2"  
read-only  
root=/dev/sdb1  
append="vt.default_utf8=1 vga=normal ro"
```

Przykładowy wpis dla Windows

```
image=no_windows  
label=windows  
description="Windows"  
read-only  
root=/dev/sda1
```

GRUB 2

UWAGA

Grub w wersji 1 nie obsługuje UEFI. Grub w wersji 2.00 bootuje Linuksa, ale nie potrafi uruchomić Windowsów. Grub w wersji 2.04 działa poprawnie z nowymi (rok 2020) płytami głównymi, GPT i Windows 10 64b.

Aby założyć grub'a należy wydać komendę

```
grub-install --efi-directory=/boot/efi
```

lub jej „bogatszą” wersję

```
grub-install --target=x86_64-efi --efi-directory=/boot/efi  
--bootloader-id=GRUBBEK --recheck --debug
```

która umieści pliki w /boot/efi/EFI/GRUBBEK. Komenda grub-install wywoła efibootmgr aby zarejestrować się w systemie UEFI.

Aby stworzyć plik konfiguracyjny, należy wydać komendę

```
grub-mkconfig -o /boot/grub/grub.cfg
```

Ważne, aby plik konfiguracyjny znalazł się w podanej ścieżce, inaczej grub nie będzie go czytał.

Uwagi:

- plik konfiguracyjny składany jest z plików `/etc/grub.d/*`; jeśli chcemy wpisać coś do `grub.cfg`, trzeba zmienić te pliki i uruchomić `grub-mkconfig`
- własne wpisy niewykrywane przez grub dodajemy do `/etc/grub.d/40_custom`
- jeśli partycja `esp` nie znajduje się na tym samym dysku, co Windows, należy te dyski wirtualnie zamienić, np.
`drivemap -s (hd0) (hd1)`
Ta metoda nie zawsze działa, zależy od sprzętu i wersji UEFI.

Schemat wpisu dla Windows

```
menuentry "Windows 10"{  
    insmod part_gpt  
    insmod fat  
    insmod chain  
    search --no-floppy --fs-uuid --set=root <hints> <uuid>  
    chainloader /EFI/Microsoft/Boot/bootmgfw.efi  
}
```

gdzie <hints> i <uuid> to wyniki zwrócone przez komendy

```
grub-probe --target=hints_string /boot/efi/EFI  
           /Microsoft/Boot/bootmgfw.efi
```

```
grub-probe --target=fs_uuid /boot/efi/EFI  
           /Microsoft/Boot/bootmgfw.efi
```

Zanim wystartuje system operacyjny, musi zostać uruchomiony komputer, czyli rozpoznany i zainicjowany sprzęt. Podstawowe sterowniki do głównych komponentów zawiera w sobie BIOS.

Piszę BIOS, ale mam tu na myśli zarówno tradycyjny BIOS jak i nowe systemy EFI/UEFI. W tym wypadku pełnią one taką samą rolę.

Start systemu na komputerze typu PC:

- BIOS resetuje komponenty sprzętowe, szuka urządzeń startowych
- BIOS ładuje bootloader (lilo, grub, isolinux – jeśli w MBR)
- bootloader wyświetla menu dostępnych opcji bootowania
- bootloader wczytuje skompresowane jądro w pierwszy MB RAMu
- to co się wgrało uruchamia CPU w trybie chronionym udostępniając mu więcej pamięci (architektura x86)
- dogrywa i dekompresuje się reszta kernela
- kernel ustawia pamięć wirtualną i główny system plików (moduły muszą być wkompilowane w jądro)
- <opcjonalnie> dogrywany jest RAM-dysk (initrd) i uruchamiane więcej sprzętu
- ładowanie reszty modułów (sterowników)
- uruchamiany jest proces init (PID1; poziomy uruchomieniowe runlevel) czytając konfigurację z `/etc/inittab`

```
# /etc/inittab
# These are the default runlevels in Slackware:
#   0 = halt
#   1 = single user mode
#   2 = unused (but configured the same as runlevel 3)
#   3 = multiuser mode (default Slackware runlevel)
#   4 = X11 with KDM/GDM/XDM (session managers)
#   5 = unused (but configured the same as runlevel 3)
#   6 = reboot

# Default runlevel. (Do not set to 0 or 6)
id:4:initdefault:

# System initialization (runs when system boots).
si:S:sysinit:/etc/rc.d/rc.S

# Script to run when going single user (runlevel 1).
su:1S:wait:/etc/rc.d/rc.K
```

```
# Script to run when going multi user.  
rc:2345:wait:/etc/rc.d/rc.M  
  
# What to do at the "Three Finger Salute".  
ca::ctrlaltdel:/sbin/shutdown -t5 -r now  
  
# Runlevel 0 halts the system.  
l0:0:wait:/etc/rc.d/rc.0  
  
# Runlevel 6 reboots the system.  
l6:6:wait:/etc/rc.d/rc.6  
  
# What to do when power fails.  
pf::powerfail:/sbin/genpowerfail start  
  
# If power is back, cancel the running shutdown.  
pg::powerokwait:/sbin/genpowerfail stop  
  
# These are the standard console login getties
```

```
# in multiuser mode:
c1:12345:respawn:/sbin/agetty 38400 tty1 linux
c2:12345:respawn:/sbin/agetty 38400 tty2 linux
c3:12345:respawn:/sbin/agetty 38400 tty3 linux
c4:12345:respawn:/sbin/agetty 38400 tty4 linux
c5:12345:respawn:/sbin/agetty 38400 tty5 linux
c6:12345:respawn:/sbin/agetty 38400 tty6 linux

(...)
# End of /etc/inittab
```

Format wpisów w `/etc/inittab`:

`id:runlevel:akcja:proces`

[id] unikatowa nazwa (1-4 znaki, 1-2 w starszych wersjach); dla konsol logowania (getty) będzie to numer konsoli (np. c1 dla tty1)

[runlevel] do których poz. uruchomieniowych wpis się odnosi

[akcja] patrz niżej

[proces] odwołanie do skryptu

Możliwe akcje (niekompletne):

- **respawn** = restart (np. tty)
- **wait** = uruchomienie i oczekiwanie na zakończenie
- **once** = uruchomienie
- **boot** = uruchomienie w trakcie startu systemu
- **off** = nic
- **initdefault** = domyślny poziom uruchomieniowy
- **powerwait** = po sygnale np. z UPS init czeka na zakończenie
- **powerfail** = j.w. ale init nie czeka na zakończenie procesu
- **powerokwait** = zasilanie znowu działa
- **powerfailnow** = zasilanie i UPS padają
- **ctrlaltdel** = SIGINT dla init'a

Skrypty

Runlevel (System V)	Opis

0 /etc/rc.d/rc0.d/	shutdown/halt system
1 /etc/rc.d/rc1.d/	Single user mode
2 /etc/rc.d/rc2.d/	Multiuser with no network services exported
3 /etc/rc.d/rc3.d/	Default text/console only. Full multiuser
4 /etc/rc.d/rc4.d/	Reserved for local use. Also X-window (Slackware/BSD)
5 /etc/rc.d/rc5.d/	XDM X-window GUI mode (Redhat/System V)
6 /etc/rc.d/rc6.d/	Reboot
s S	Single user/Maintenance mode (=1, Slackware)
M	Multiuser mode (=3, Slackware)

Tradycyjny init obsługuje do 32 poziomów uruchomieniowych. Podobno nikt nie przetestował poziomów powyżej 6.

Skrypty startowe wg System V (debian, redhat...)

- wszystkie skrypty znajdują się w `/etc/init.d`
- dla poszczególnych poziomów uruchomieniowych istnieją katalogi `/etc/rc1.d`, `/etc/rc2.d` itd. zawierające dowiązania symboliczne do skryptów z `init.d`
- nazwy dowiązań to `K10cron` (kill cron), `S18portmap` (start portmap)
- taki system porządkuje zatrzymywanie i uruchamianie usług w trakcie zmiany poziomów uruchomieniowych
- istnieją programiki graficzne pomagające układać te schematy

Przykładowy katalog skryptów dla 0 runlevela (debian)

```
[bash]# ls /etc/rc0.d/
```

```
K01gdm -> ../init.d/gdm
K11cron -> ../init.d/cron
K19nis -> ../init.d/nis
K20acpid -> ../init.d/acpid
K20cupsys -> ../init.d/cupsys
K20dbus -> ../init.d/dbus
K20makedev -> ../init.d/makedev
K20ssh -> ../init.d/ssh
K25hwclock.sh -> ../init.d/hwclock.sh
K50proftpd -> ../init.d/proftpd
K79nfs-common -> ../init.d/nfs-common
K89klogd -> ../init.d/klogd
K90sysklogd -> ../init.d/sysklogd
S20sendsigs -> ../init.d/sendsigs
S30urandom -> ../init.d/urandom
S31umountnfs.sh -> ../init.d/umountnfs.sh
(...)
S90halt -> ../init.d/halt
```

Skrypty startowe wg. systemu BSD (BSD, slackware)

- wszystkie skrypty w jednym miejscu (slackware: `/etc/rc.d`)
- zasadniczo jeden skrypt dla każdego poziomu uruchomieniowego, ale dla lepszej przejrzystości niektóre kawałki (np. konfiguracja połączeń sieciowych, start mysql...) są wyrzucone do zewnętrznych plików
- polecenia zawarte w skrypcie są wykonywane zgodnie z kolejnością
- ten system jest bardziej przejrzysty i łatwiejszy w obsłudze

Katalog skryptów startowych (slackware, system BSD)

```
[bash]# ls /etc/rc.d
```

rc.0 -> rc.6*	rc.keymap*
rc.4*	rc.local*
rc.6*	rc.M*
rc.cups*	rc.modules*
rc.firewall*	rc.netdevice*
rc.font*	rc.S*
rc.gpm	rc.serial*
rc.inet1*	rc.sshd*
rc.inet1.conf	rc.syslog*
rc.inet2*	rc.sysvinit*
rc.ip_forward	rc.udev
rc.K*	

Włączenie/wyłączenie usługi to dodanie lub usunięcie flagi wykonywalnej dla danego skryptu.

Zamienniki starego inita:

InitNG, Runit, **Upstart** (Ubuntu, Fedora, Debian – jako opcja)

- usługi i zadania są uruchamiane i zatrzymywane przez określone zdarzenia zaistniałe w systemie
- zdarzenia mogą być wygenerowane przez dowolny proces działający w systemie
- zdarzeniami są również uruchomienie lub zatrzymanie usługi
- usługi mogą być wznowione, jeśli z jakiegoś powodu zostały wyłączone
- nadzór nad demonami, które są odłączone od swoich procesów nadrzędnych
- obecnie potrafi współpracować z initem poprzez demona D-Bus

Ostatnie stabilne wydanie: IX.2014. W przygotowaniu:

- start/stop usług zgodnie z harmonogramem czasowym
- start/stop usług gdy zmiany na systemie plików
- start/stop usług przez normalnego użytkownika

Zamiennik starego inita: **systemd**

Potrafi (albo będzie potrafił):

- kontroluje bootowanie
- uruchamia i usypia usługi/demony na żądanie
- kontroluje procesy
- kontroluje punkty (auto)montowania
- kontroluje usługi na podstawie list zależności
- prowadzi logi
- kontroluje krytyczną konfigurację systemu (hostname, data...)
- kontroluje zalogowanych użytkowników
- kontroluje sieci
- kontroluje maszyny wirtualne, kontenery
- kontroluje katalogi systemowe

Współpraca z bootloaderem:

- zestaw narzędzi pozwalający na uruchomienie poprzedniej wersji konfiguracji, jeśli rozruch się zaczyna:
`kernel-install`, `systemd-boot`,
`systemd-boot-check-no-failures.service`,
`systemd-bless-boot.service`, `systemd-bless-boot-generator`
- czytanie zmiennych i parametrów z bootloadera, o ile korzystamy z EFI; wszystkie zmienne identyfikują się przez
`vendor UUID=4a67b082-0a4c-41cf-b6c7-440b29bb8c4f`

Komunikacja między procesami: mechanizmy uwierzytelniania procesów między sobą (login, hasło, klucz, certyfikat...) nie poprzez zmienne w skryptach, ani nie poprzez nieszyfrowane pliki.

systemd jest kompatybilny z plikami SystemV oraz skryptami startowymi LSB. Jeśli zostanie uruchomiony jako init z PID=1, wywoła `telinit` i przekaże mu wszystkie swoje opcje wywołania. Konfigurację czyta z:

```
/etc/systemd/system.conf  
/etc/systemd/system.conf.d/*.conf  
/run/systemd/system.conf.d/*.conf  
/lib/systemd/system.conf.d/*.conf
```

Jeśli jest wywołany w przestrzeni użytkownika, konfiguracja jest czytana z:

```
$HOME/.config/systemd/user.conf  
/etc/systemd/user.conf  
/etc/systemd/user.conf.d/*.conf  
/run/systemd/user.conf.d/*.conf  
/usr/lib/systemd/user.conf.d/*.conf
```

Bazową jednostką jest unit. Jest ich 11 typów:

- service (usługa)
- scope (usługa zewnętrzna, już uruchomiona)
- target (grupowanie i synchronizacja unitów)
- slice (grupowanie unitów z jednego procesu systemowego)
- device
- mount, automount, swap
- socket (aktywacja usług na podstawie gniazd)
- timer (aktywacja usług na podstawie zegara)
- path (aktywacja usług na podstawie systemu plików)

Pomiędzy unitami jest system zależności, zarówno statycznych, jak i czasowych. Większość jest skonfigurowana w plikach, ale **część jest tworzona automatycznie z innych plików konfiguracyjnych, stanu systemu lub jako wynik działania programów przy starcie systemu.**

Zależności konfigurowane są wpisami

Requires=

Conflicts=

After=

Before=

UWAGA: unity mogą być tworzone dynamicznie podczas bootowania.

Przykładowy unit: /lib/systemd/system/ssh.service

[Unit]

Description=OpenBSD Secure Shell server

After=network.target auditd.service

ConditionPathExists=!/etc/ssh/sshd_not_to_be_run

[Service]

EnvironmentFile=-/etc/default/ssh

ExecStartPre=-/bin/sh -c 'if ! ls /etc/ssh/ssh_host_*

>/dev/null 2>&1; then /usr/sbin/dpkg-reconfigure openssh-server; fi'

ExecStart=/usr/sbin/sshd -D \$SSHD_OPTS

ExecReload=/bin/kill -HUP \$MAINPID

KillMode=process

Restart=on-failure

RestartPreventExitStatus=255

Type=notify

[Install]

WantedBy=multi-user.target

Alias=sshd.service

Podczas bootowania uruchamiany jest unit `default.target`, który zazwyczaj jest linkiem do `multi-user.target` lub `graphical.target`. Procesy uruchamiane przez `systemd` są umieszczane w prywatnych grupach kontrolnych (zobacz `cgroups.txt` w dokumentacji jądra). Grupy te kontroluje jądro, co można sprawdzić poleceniem:

```
[bash]# ps xawf -eo pid,user,cgroup,args
```

Więcej informacji:

systemd.io

www.dobreprogramy.pl/

[@mariushko/poradnik-systemd-cz.-1,67627](https://mariushko.com/poradnik-systemd-cz.-1,67627)

Operacje na unitach, uruchomionych procesach, stanach itd. można wykonać poleceniem `systemctl`.

Lista unitów i ich stanów:

```
[bash]# systemctl list-unit-files --all
```

Uruchomienie, aktywacja, sprawdzenie stanu:

```
[bash]# systemctl {start|stop|restart|reload} usługa.typ
```

```
[bash]# systemctl {status|enable|reenable|disable} usługa.typ
```

Lista uruchomionych, aktywnych:

```
[bash]# systemctl {is-active|is-enabled} usługa.typ
```

Zablokowanie, odblokowanie uruchamiania usługi:

```
[bash]# systemctl {mask|unmask} usługa.typ
```

System Linux – cz. 3

- Usługi startowe systemu.
- Partycje: `/etc/fstab`.
- Konta i grupy: `/etc/passwd`, `/etc/groups`
- Logi, manuale i dokumentacja
- Konfiguracja sprzętu.
- Dostrajanie systemu, lokalizacja.

Przykładowy zestaw demonów i usług, uruchamianych podczas startu systemu (/etc/rc.d/rc.M w slackware 11.0)

- udev (tworzenie wpisów w katalogu /dev)
- syslogd (tworzenie logów)
- pcmcia (uruchomienie złączy PCMCIA)
- karty sieciowe
- hotplug (automatyczne wykrywanie sprzętu)
- demony sieciowe (/etc/rc.d/rc.inet2)
 - ▶ portmap (/etc/services)
 - ▶ rpc (remote procedure calls)
 - ▶ firewall (/etc/rc.d/rc.firewall)
 - ▶ IPv4 packet forwarding (maskarada) (/etc/rc.d/rc.ip_forward)
 - ▶ inetd (/etc/rc.d/rc.inetd) (ew. xinet)
 - ▶ sshd (/etc/rc.d/rc.sshd)
 - ▶ bind (name server daemon) (/etc/rc.d/rc.bind)
 - ▶ NIS (Network Information Service) (/etc/rc.d/rc.ypp)
 - ▶ NFS (Network Files System) (/etc/rc.d/rc.nfsd)
 - ▶ routed (routing daemon)
 - ▶ rwhod (system status daemon)

- czyszczenie systemu plików (/tmp, /var)
- ldconfig (odświeżenie bazy współdzielonych bibliotek)
- fc-cache (indeksy czcionek)
- drukarka (lpd lub CUPS)
- netatalk (drukowanie w Mac'ach)
- smartd (obsługa S.M.A.R.T.)
- genpowerd (obsługa UPS)
- accton (BSD process accounting)
- crond (harmonogram zadań)
- atd (job scheduler)
- quota (ograniczenia miejsca na dysku)

- SASL authentication for sendmail
- APM lub ACPI (zarządzanie energią)
- ALSA (dźwięk)
- czcionka ekranowa
- mapowanie klawiatury
- mySQL (baza danych)
- Apache (serwer www)
- OpenLDAP (Lightweight Directory Access Protocol)
- Samba (sieć windows)
- GPM (mysz w konsoli tekstowej)
- local

Pakiet e2fsprogs

Zawiera narzędzia do obsługi partycji typu ext2/ext3/ext4

`mke2fs [opcje] urządzenie`

Aliasy to `mkfs.ext2`, `mkfs.ext3` i `mkfs.ext4`. Opcje można zazwyczaj pominąć i ustawione zostaną domyślne wartości.

Pakiet reiserfsprogs

Zawiera narzędzia do obsługi partycji typu reiserfs

`mkfs.reiserfs [opcje] urządzenie`

Podobne pakiety: `jfsutils`, `xfspgms`, `dosfstools`

Dostrajanie partycji ext2/ext3/ext4

```
tune2fs [opcje] [urządzenie]
```

-c max-mount-counts

-j dodaj dziennik do partycji ext2 (tworząc ext3)

-L LABEL ustaw etykietę partycji

Dostrajanie partycji reiserfs

```
tune.reiserfs [opcje] [urządzenie]
```

Opcje służą głównie do zarządzania dziennikiem, przenoszeniem go w inne miejsce, odtwarzaniem etc.

Sprawdzanie i naprawa partycji ext2/ext3/ext4

`e2fsck [opcje] urządzenie`

- D optymalizacja katalogów
- p automatyczna naprawa błędów

Sprawdzanie i naprawa partycji reiserfs

`fsck.reiserfs [opcje] urządzenie`

- rebuild-sb odtwarza superblock
- check sprawdzanie
- fix-fixable naprawa mało poważnych błędów
- rebuild-tree naprawa poważnych błędów – przebudowa systemu plików

Plik `/etc/fstab` definiuje opcje montowania standardowych partycji (polecenia `mount`, `umount`)

<code>/dev/sda5</code>	<code>/</code>	<code>reiserfs</code>	<code>defaults</code>	<code>1 1</code>
<code>/dev/sda6</code>	<code>/home</code>	<code>reiserfs</code>	<code>defaults</code>	<code>0 2</code>
<code>#/dev/sda7</code>	<code>/opt</code>	<code>reiserfs</code>	<code>defaults</code>	<code>0 2</code>
<code>/dev/sda8</code>	<code>swap</code>	<code>swap</code>	<code>defaults</code>	<code>0 0</code>
<code>/dev/sda1</code>	<code>/mnt/windows</code>	<code>ntfs</code>	<code>noauto,owner,ro</code>	<code>0 0</code>
<code>/dev/cdrom</code>	<code>/mnt/cdrom</code>	<code>auto</code>	<code>noauto,user,ro</code>	<code>0 0</code>
<code>/dev/sdb1</code>	<code>/mnt/usb</code>	<code>vfat</code>	<code>noauto,user</code>	<code>0 0</code>
<code>/dev/mmcblk0p1</code>	<code>/mnt/sd</code>	<code>vfat</code>	<code>noauto,user</code>	<code>0 0</code>
<code>devpts</code>	<code>/dev/pts</code>	<code>devpts</code>	<code>gid=5,mode=620</code>	<code>0 0</code>
<code>proc</code>	<code>/proc</code>	<code>proc</code>	<code>defaults</code>	<code>0 0</code>

Struktura wpisów w `/etc/fstab`

[urządzenie] [punkt montowania] [typ partycji] [opcje] [x] [y]

Urządzenia:

`/dev/hda1`, `/dev/hdb1...` typu IDE (dyski, cdrom...)

`/dev/sda1`, `/dev/sdb1...` typu SCSI (dyski SCSI/SATA, cdrom SCSI, pamięci USB)

Typy partycji:

`adfs`, `affs`, `autofs`, `coda`, `coherent`, `cramfs`, `devpts`, `efs`, `ext2`, `ext3`, `ext4`, `hfs`, `hpfs`, `iso9660`, `jfs`, `minix`, `msdos`, `ncpfs`, `nfs`, `ntfs`, `proc`, `qnx4`, `reiserfs`, `romfs`, `smbfs`, `sysv`, `tmpfs`, `udf`, `ufs`, `umsdos`, `vfat`, `xenix`, `xfs...`

Opcje: patrz → `man mount`, `man fstab`

auto	respektuje <code>mount -a</code>
noauto	ignoruje <code>mount -a</code>
owner	tylko właściciel urządzenia (<code>root</code>)
user	również użytkownik może montować
users	każdy może (od)montować; =+noexec,nosuid,nodev
defaults	= rw,suid,dev,exec,auto,nouser,async
dev	respektuje urządzenia blokowe/znakowe
exec	daje prawa do wykonywania dla programów
group	użytkownik może montować jeśli należy do grupy
ro,rw	read-only, read-write
suid	zezwala na bit <code>suid</code>
(a)sync	I/O na systemie plików (a)synchronicznie

Opcje dalsze nadpisują wcześniejsze, więc ma sens: **users,exec,suid**

Dwie ostatnie kolumny w:

[urządzenie] [punkt montowania] [typ partycji] [opcje] [x] [y]

x: flaga 1 jeśli partycja ma być backupowana

y: kolejność sprawdzania partycji przez fsck:

- 1 dla głównego systemu plików
- 2 dla pozostałych
- 0 dla partycji tymczasowych (urządzenia usb), innego typu (windows, procfs...)

Polecenie `mount` korzysta z wpisów w `/etc/fstab`, operuje na `/etc/mtab`. Struktura:

```
mount [urządzenie] [punkt_montowania] [opcje]
```

Przykłady

```
[bash]# mount -a
```

```
[bash]# mount /dev/hdb1
```

```
[bash]# mount /mnt/cdrom
```

```
[bash]# mount /dev/cdrom /mnt/cdrom -t iso9660
```

```
[bash]# mount ./slackware.iso /tmp/slack -t iso9660 -o loop
```

```
[bash]# umount /dev/cdrom
```

```
[bash]# umount /mnt/cdrom
```


Naprawa systemu na dysku /dev/sda1

boot z live-cd

```
[bash]# mount -o dev /dev/sda1 /media/sda1
```

```
[bash]# chroot /media/sda1
```

```
[bash]# mount -a
```

... naprawiamy system ...

```
[bash]# umount -arvf
```

/ostatnia komenda powoduje zapis zmienionych plikow na dysk!/

UWAGA: jeśli urządzenie ma być wykrywane i montowane automatycznie, nie może być jego wpisu w `/etc/fstab`:

- wtedy urządzenie zostanie poprawnie wykryte przez np. KDE Media Manager
- powstanie automatycznie katalog, np. `/media/USB32GB`
- fs zostanie tam zamontowany z odpowiednimi prawami

Jeśli wpis dla danego urządzenia (np. `/dev/sdb1`) istnieje w `/etc/fstab`, będzie on narzucał typ systemu plików, punkt montowania i opcje montowania dla danego urządzenia. W przypadku używania różnych urządzeń na złączu usb, które mogą być tak samo identyfikowane przez system, może to prowadzić do niepoprawnej ich obsługi.

Prace nad automontowaniem partycji (również dysków sieciowych) trwają od dawna. Powstało wiele propozycji i projektów przejściowych.

Automatyczne montowanie urządzeń zewnętrznych – historia

- amd (automount daemon, starszy, bardzo rzadko używany)
- autofs (nowszy, ale też rzadko używany)
- hal + D-Bus + (gnome/kde volume manager, thunar volman)
 - ▶ hal — Hardware Abstraction Layer — ma wbudowaną funkcjonalność hotplug'a; współpracuje z udev aby wykrywać i zlecać tworzenie nowych urządzeń w systemie
 - ▶ D-Bus — Message Bus System — jest pośrednikiem przekazującym informacje pomiędzy jądrem a innymi demonami, lub dowolnymi innymi aplikacjami (komunikacja międzyprocesowa)
 - ▶ menedżery woluminów, korzystając z obu poprzednich mechanizmów, są odpowiedzialne za montowanie (za pomocą hal'a), pokazanie ikony na pulpicie etc.

Obecnie pełną funkcjonalność obsługi sprzętu przejął system udev. Nie instaluje się już hal'a, który został przez udev wchłonięty.

Do zarządzania urządzeniami wymiennymi używa się `udisks` i `udisks2`. Dostarczają one interface'y dla D-Bus'a pozwalające na zarządzanie i manipulowanie urządzeniami typu pamięci masowej.

Przykłady

Wyświetla urządzenia pamięci masowej:

```
[bash]# udisks -enumerate
```

Monitoruje urządzenia pamięci masowej:

```
[bash]# udisks -monitor
```

```
[bash]# udisks -monitor-details
```

Wyłącza próbkowanie urządzenia:

```
[bash]# udisks -inhibit-polling /dev/nazwa
```

Podsumowanie

- Urządzenia obsługuje kernel (moduły)
- Nowe nody, uzupełnienie katalogu `/dev`, klasyfikacja urządzenia i załadowanie jego sterownika (modułu) leżą w gestii systemu `udev`
- Komunikacja między procesami oraz procesami i jądrem jest obsługiwana przez D-Bus
- Interface'y D-Bus pozwalające na manipulowanie pamięcią masową są udostępniane przez demony `udisks`
- Właściwe tworzenie punktów montowania, montowanie i udostępnienie urządzenia użytkownikowi jest realizowane przez KDE Media Manager, Gnome Volume Manager, Thunar Volman (XFCE) lub inne, jeśli się z nich korzysta

Obsługa kont w linuxsie

- standardowy system passwd został zastąpiony dawno temu pakietem shadow
- shadow podmienia następujące aplikacje: su, login, passwd, newgrp, chfn, chsh, id
- shadow dodaje: chage, newusers, dpasswd, gpasswd, useradd, userdel, usermod, groupadd, groupdel, groupmod, groups, pwck, grpck, lastlog, pwconv, pwunconv
- konfiguracja do programu login znajduje się w /etc/login.defs (tam można zdefiniować m.in. metodę hashowania haseł)
- shadow współpracuje z aplikacjami wykorzystującymi hasła użytkowników, w tym z xlock, ftpd, pop3d, xdm, sudo, pppd, które trzeba kompilować z jawną obsługą shadow

Plik `/etc/passwd` (fragment):

```
root:x:0:0:/:root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/log:
lp:x:4:7:lp:/var/spool/lpd:
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:/:
news:x:9:13:news:/usr/lib/news:
uucp:x:10:14:uucp:/var/spool/uucppublic:
operator:x:11:0:operator:/root:/bin/bash
games:x:12:100:games:/usr/games:
ftp:x:14:50::/home/ftp:
marek:x:101:100:,,,:/home/marek:/bin/bash
```

Plik /etc/shadow (fragment):

```
root:$1$23D0zji6$4l4//v4vH5PJ6x40:13655:0:0:0:0:
bin:!:9797:0:0:0:0:
daemon:!:9797:0:0:0:0:
adm:!:9797:0:0:0:0:
lp:!:9797:0:0:0:0:
sync:!:9797:0:0:0:0:
pop:!:9797:0:0:0:0:
nobody:!:9797:0:0:0:0:
marek:$1$BKuitML4$rhf36jaI4Yok3Tjha1:13663:0:99999:7:0:
```

Struktura:

nazwa użytkownika (login) : zakodowane hasło : ostatnia zmiana hasła :
 dni do dozwolonej zmiany : dni przed wymaganą zmianą :
 dni ostrzeżenia o wygaśnięciu : dni przed wyłączeniem konta :
 data ważności konta : zarezerwowane do przyszłego użytku

Plik /etc/group (fragment):

```
root::0:root
bin::1:root,bin,daemon
daemon::2:root,bin,daemon
sys::3:root,bin,adm
adm::4:root,adm,daemon
wheel::10:root
floppy::11:root
mail::12:mail
cdrecorder::60:root,marek
pop::90:pop
nobody::98:nobody
nogroup::99:
users::100:
console::101:
```

Dodanie nowego użytkownika do systemu to:

- wpis w `/etc/passwd`
- wpis w `/etc/group`
- zmiana hasła za pomocą `passwd`
- `mkdir /home/login`
- `cp /etc/skel/* /home/login`
`chown -R login.grupa /home/login`
- `echo "Witaj!" > /var/spool/mail/login`

Jakie numery UID i GID przydzielać?

- system tradycyjny: UID od 101, GID 100 (users)
- system nowy 1: UID od 500, GID od 500 (=login)
- system nowy 2: UID od 1000, GID od 1000 (=login)

W zasadzie dowolność, ale UID poniżej 100 (w nowych konfiguracjach poniżej 1000) są zarezerwowane dla systemu.

```
[bash]# id
```

```
uid=101(marek) gid=100(users) grupy=11(floppy),17(audio),  
18(video),19(cdrom),83(plugdev),100(users)
```

```
[bash]# whoami
```

```
marek
```

```
[bash]# groups
```

```
users floppy audio video cdrom plugdev
```

Dodanie nowego użytkownika:

```
useradd [-c komentarz] [-d katalog_domowy]  
        [-e data_ważności] [-f dni_nieaktywności]  
        [-g grupa_początkowa] [-G grupa [...]]  
        [-m [-k katalog_wzorców]] [-o] [-p hasło]  
        [-s powłoka] [-u uid] login
```

adduser robi to samo w trybie interaktywnym (skrypt w slackware)

Sprawdzenie lub zmiana wartości domyślnych:

```
useradd -D [-g domyślna_grupa] [-b katalog_bazowy]  
          [-f domyślne_dni_nieaktywności]  
          [-e domyślna_data_ważności]  
          [-s domyślna_powłoka]
```

```
[bash]# /usr/sbin/useradd -D
```

```
GROUP=100
```

```
HOME=/home
```

```
INACTIVE=-1
```

```
EXPIRE=
```

```
SHELL=/bin/bash
```

```
SKEL=/etc/skel
```

```
CREATE_MAIL_SPOOL=yes
```

Jest to zawartość pliku /etc/default/useradd

Zmiana ustawień konta:

```
chfn [-f pełna_nazwa] [-r nr_pokoju]
      [-w tel_służb] [-h tel_dom] [-o inne] [użytkownik]
```

```
chsh [-s powłoka] [użytkownik]
```

```
usermod [-c komentarz] [-d katalog_domowy [-m]]
        [-e data_ważności] [-f dni_nieaktywności]
        [-g grupa_początkowa] [-G grupa[,...]]
        [-l login_name] [-s powłoka]
        [-u uid [-o]] login
```

```
userdel [-r] login
```

Polecenie `passwd` służy do zarządzania hasłami kont. Współpracuje z pakietem szyfrującym `shadow`, który przenosi zakodowane hasła z otwartego pliku `/etc/passwd` do niedostępnego pliku `/etc/shadow`

```
passwd [-f|-s] [nazwa]
passwd [-g] [-r|-R] grupa
passwd [-x max] [-n min] [-w ostrzeż] [-i nieakt] login
passwd {-l|-u|-d|-S|-e} login
```

Opcje

- f wywołuje chfn
- s wywołuje chsh
- g zmiana hasła grupy
- g -r kasowanie hasła grupy
- g -R blokowanie dostępu do grupy
- x liczba dni ważności hasła
- n min liczba dni między zmianami haseł
- w ile dni system ostrzega o wygaśnięciu hasła
- i po ilu dniach od wygaśnięcia hasła konto blokowane
- e natychmiastowe wygaśnięcie hasła
- d usunięcie hasła – brak autoryzacji!
- l blokowanie konta (nie kasowanie); dodaje ! na początku hasła w /etc/shadow
- u odblokowanie konta
- S stan konta


```
[bash]# passwd -S  
marek P 05/01/2006 0 99999 7 -1
```

Składnia:

- login
- **P** password protected; **L** locked; **NP** no password
- data ostatniej zmiany hasła
- minimalny i maksymalny wiek hasła (dni)
- okres ostrzegania o wygaśnięciu
- okres nieaktywności hasła

Operacje na grupach

- ręczna edycja pliku `/etc/group`
- tworzy nową grupę
`groupadd -g [gid] nazwa`
- usuwa grupę
`groupdel nazwa`
- modyfikuje grupę
`groupmod -g [gid] -n nowa_nazwa nazwa`

Grupy można zabezpieczyć hasłem (plik /etc/gshadow)

```
[bash]# groupadd -p 'openssl passwd -1' gzh
```

Tworzy grupę gzh z haszem hasła wygenerowanym przez openssl metodą BSD-1 (opartą o MD5). Polecenie sg (analog do su odnoszący się do uprawnień grupy) dodaje użytkownika do grupy po podaniu hasła

```
[bash]# id
```

```
uid=101(marek) gid=100(users) groups=100(users)
```

```
[bash]# sg gzh
```

```
Password:
```

```
[bash]# id
```

```
uid=101(marek) gid=1002(gzh) groups=100(users),1002(gzh)
```

Chcąc dodać hasło do istniejącej grupy:

```
[bash]# groupmod -p 'openssl passwd -1' passgroup
```

Można też użyć polecenia gpasswd

Podstawowe bezpieczeństwo systemu (konta)

- Hasła: muszą być szyfrowane (pakiet shadow), zawierać wielkie i małe litery, znaki alfanumeryczne i cyfry; nie mogą to być imiona, nazwiska, nazwy słownikowe, nazwy programów, numery telefonów, kart kredytowych etc. Wymuszać ich zmianę co około 40 dni.
- Grupy i prawa dostępu: restrykcyjne, przydzielać grupy tylko tym, którym to potrzebne, zabierać gdy potrzeba wygaśnie.
- Konta: usuwać konta przedawnione i tymczasowe; samo blokowanie konta nie wystarcza. Zakładać konta na określony czas.
- Powłoki: minimalna liczba powłok dostępnych w systemie (/etc/shells).

Bezpieczeństwo i elastyczność systemu Linux leży w umiejętnym użyciu mechanizmów

- praw dostępu do plików i katalogów (→ urządzeń) – właściciel
- praw dostępu do plików i katalogów (→ urządzeń) – grupa
- przydzieleniu użytkowników do odpowiednich grup głównych
- przydzieleniu użytkowników do odpowiednich grup dodatkowych

Wykorzystanie haseł dla grup (ktoś to robi?) daje możliwość:

- ustanowienia administratorów dla części systemu
- przydzielania zaufanego dostępu do części systemu (dane, urządzenia)

Logi systemowe – przyjaciel administratora

- Unix i Linux mają rozbudowany system komunikacji system-człowiek
- każdy porządny program systemowy, w tym jądro i sterowniki, wysyłają komunikaty podczas pracy
- komunikaty te dzielą się na kategorie
- konfiguracja ich zbierania i tworzenia logów zależna od dystrybucji, ale stosunkowo łatwa do własnej modyfikacji

```
[bash]# ls -l /var/log/
```

```
-rw----- 1 root root  btmp
-rw-r----- 1 root root  cron
-rw-r----- 1 root root  cron.1
-rw-r----- 1 root root  debug
-rw-r--r-- 1 root root  dmesg
-rw-r----- 1 root root  faillog
-rw-r--r-- 1 root root  kdm.log
-rw-r--r-- 1 root root  lastlog
-rw-r----- 1 root root  maillog
-rw-r----- 1 root root  messages
-rw-r--r-- 1 root root  nvidia-installer.log
-rw-r--r-- 1 root root  nvidia-nforce-installer.log
-rw-r----- 1 root root  secure
-rw-r----- 1 root root  spooler
-rw----- 1 root users sulog
-rw-r----- 1 root root  syslog
-rw-rw-r-- 1 root utmp  wtmp
-rw-r--r-- 1 root root  Xorg.0.log
-rw-r--r-- 1 root root  Xorg.0.log.old
```

Przydział logów w slackware:

- btmp faillog lastlog wtmp: binarne zapisy logowań

```
[bash]# last -f /var/log/btmp
```

```
.....
```

```
btmp begins Sun Mar 29 13:56:54 2009
```

```
[bash]# last
```

```
marek pts/1 :0.0 Thu Oct 22 14:31 still logged in
marek pts/0 :0 Thu Oct 22 13:51 still logged in
marek :0 Thu Oct 22 13:50 still logged in
reboot system boot 2.6.27.7 Thu Oct 22 13:50 (03:15)
marek pts/0 :0 Wed Oct 21 10:13-12:16 (02:02)
marek :0 Wed Oct 21 10:13-12:16 (02:02)
reboot system boot 2.6.27.7 Wed Oct 21 10:13 (02:03)
```


- cron: logi schedulera (cron.1 to plik zarchiwizowany)
- debug dmesg messages syslog secure: logi systemu (głównie kernel i demony)
- maillog: logi poczty
- spooler: news, uucp
- Xorg.0.log: X window system

Cały system logowania, nazwy plików i rozdział komunikatów, zależy od konfiguracji i użytego programu zarządzającego logami.

Popularnymi programami są syslog i syslog-ng.

Dodatkowo klogd.

Przykład: [syslog](#)

- Źródła komunikatów: auth, authpriv, cron, daemon, kern, lpr, mail, mark (nieużywane), news, security (=auth, nieużywane), syslog, user, uucp, local0-7
- Priorytety komunikatów: debug, info, notice, warning, warn (=warning, nieużywane), err, error (=err, nieużywane), crit, alert, emerg, panic (=emerg, nieużywane)
- W pliku konfiguracyjnym podaje się typy komunikatów i przypisane im akcje (logowanie do pliku, na zdalnej maszynie, mailem, na konsolę).

```
# /etc/syslog.conf
#kern.* /dev/console

# Log anything 'info' or higher, but lower than 'warn'.
# Exclude authpriv, cron, mail, and news.
*.info;*.!warn;\
authpriv.none;cron.none;mail.none;news.none -/var/log/messages

# Log anything 'warn' or higher.
# Exclude authpriv, cron, mail, and news.
*.warn;\
authpriv.none;cron.none;mail.none;news.none -/var/log/syslog

# Debugging information is logged here.
*.=debug                                     -/var/log/debug

# Private authentication message logging:
authpriv.*                                   -/var/log/secure
```

Cron related logs:

cron.* -/var/log/cron

Mail related logs:

mail.* -/var/log/maillog

Emergency level messages go to all users:

*.emerg *

This log is for news and uucp errors:

uucp,news.crit -/var/log/spooler

Administrowanie logami: [logrotate](#) (oraz /etc/logrotate.d/)

```
# /etc/logrotate.conf
weekly      # rotate log files weekly
rotate 4    # keep 4 weeks worth of backlogs
create      # create new (empty) log files
# compress  # compress logs

include /etc/logrotate.d

# Rotate /var/log/wtmp:
/var/log/wtmp {
    monthly
    create 0664 root utmp
    rotate 1
}
```

Konfiguracja sprzętu

Sprzęt działa w systemie, jeśli jest załadowany do pamięci odpowiedni sterownik. Dodatkowo często potrzebny jest program do wykorzystania możliwości danego urządzenia i komunikowania się z nim przez użytkownika. Zarejestrowany sprzęt powinien pojawić się w

- katalogu `/sys`,
- katalogu `/proc` (opcjonalnie),
- jako komunikat w logach systemowych,
- programach do jego obsługi.

Generalna strategia:

- podłącz urządzenie
- dmesg – wyrzuca na stdout komunikaty kernela:
/var/log/dmesg
/var/log/syslog
/var/log/messages
- udev często radzi sobie sam
- modułów („sterowników”) szukaj w
/lib/modules/<wersja jądra>/kernel/*
- dokumentacja, google
- modprobe, insmod, rmmod, lsmod + dmesg

Skąd wiem jaki to sprzęt?

```
[bash]# lspci
```

```
00:02.0 PCI bridge: nVidia Corporation C51 PCI Express Bridge
00:0b.0 USB Controller: nVidia Corporation MCP51 USB Controller
00:10.1 Audio device: nVidia Corporation MCP51
        High Definition Audio (rev a2)
00:14.0 Bridge: nVidia Corporation MCP51 Ethernet Controller
00:18.0 Host bridge: Advanced Micro Devices [AMD] K8
        [Athlon64/Opteron] HyperTransport Technology
01:00.0 Ethernet controller: Atheros Communications Inc.
        AR242x 802.11abg Wireless PCI Express Adapter (rev 01)
04:00.0 VGA compatible controller: nVidia Corporation G72M
        [Quadro NVS 110M/GeForce Go 7300] (rev a1)
05:01.0 FireWire (IEEE 1394): Ricoh Co Ltd R5C832 IEEE 1394
```

Opcja `lspci -k` dodatkowo wyświetla moduły podłączone dla danego sprzętu.

Dla urządzeń usb

[bash]# lsusb

```
Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate
Matching Hub
Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate
Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 010: ID 413c:2513 Dell Computer Corp. internal USB
Hub of E-Port Replicator
Bus 001 Device 011: ID 413c:2513 Dell Computer Corp. internal USB
Hub of E-Port Replicator
Bus 001 Device 005: ID 413c:8187 Dell Computer Corp. DW375
Bluetooth Module
Bus 001 Device 006: ID 1bcf:2802 Sunplus Innovation Technology Inc.
Bus 002 Device 003: ID 0a5c:5800 Broadcom Corp. BCM5880
Secure Applications Processor
Bus 001 Device 012: ID 046d:c401 Logitech, Inc. TrackMan
Bus 001 Device 013: ID 0482:03c5 Kyocera Corp.
```

Klasycznie katalog `/dev` jest tworzony statycznie przy starcie systemu. Tworzone są wszystkie możliwe urządzenia, które mogą wystąpić w systemie (zależnie od konfiguracji; w Fedora Core 1 `/dev` ma 18000+ wpisów!)

Podmieniono to w linuksie 2.5 na dynamicznie tworzony `/dev` przez system `devfs` (2610 wpisów w Slackware 11.0 bez `udev`).

W tym momencie `devfs` jest zastąpiony przez `udev` (jądra 2.6.15 i nowsze).

Początkowo udev współpracował z zewnętrznym programem hotplug, który pośredniczył między kernelem a udev dostarczając informacji o pojawieniu się lub zniknięciu z systemu urządzenia.

Obecnie udev **ma zaimplementowane funkcje hotplug'a** i nie ma konieczności korzystania z obu naraz. Współpracuje z **sysfs** pobierając z niego dane o sprzęcie w systemie, w szczególności o przypisywanych mu **liczbach major i minor**.

Tworząc poprawny zestaw reguł dla udev, urządzenia będą zawsze występowały w systemie pod tymi samymi nazwami, np. /dev/ZenMicro, /dev/pendrive-Jasia-2GB etc.

Problem niejednolicie przypisywanych nazw dotyczy wszystkich urządzeń hot-plug: usb, ieee 1394 (firewire), CardBus...

udev (2.6) vs devfs (2.4) – cytaty –

- udev jest niezależny od numerów major i minor urządzeń, natomiast devfs jest
- **udev obsługuje nadawanie urządzeniom stałych nazw, które nie zmieniają się po restarcie i nie zależą od kolejności załadowania sterowników, czy podłączenia urządzeń**
- udev jest narzędziem działającym w przestrzeni użytkownika, zaś devfs działa w przestrzeni jądra; powoduje to przeniesienie obsługi nazw urządzeń poza jądro, co jest uważane za zaletę
- udev jest mniejszy i zgodny z wytycznymi co do sposobu nazywania urządzeń zawartego w Linux Standard Base
- niektórzy deweloperzy uważają, że devfs cierpi z powodu braku opieki, zaprojektowany został w sposób nie nadający się do poprawienia i powoduje poważne problemy

Konfiguracja udev znajduje się w `/etc/udev/udev.conf`

```
# udev.conf
# The main config file for udev
#
# This file can be used to override some of udev's default values for where it
# looks for files, and where it places device nodes.

# udev_root - where in the filesystem to place the device nodes
udev_root="/dev/"

# udev_log - The initial syslog(3) priority: "err", "info", "debug" or
# its numerical equivalent. For runtime debugging, the daemons
# internal state can be changed with: udevadm control log_priority=<value>
udev_log="err"
```

Standardowo konfiguracja reguł udev znajduje się w `/etc/udev/udev.rules`, ale patrz też `/etc/udev/rules.d/*.rules`.
Np.:

```
[bash]# ls -R /etc/udev/
```

```
udev:  
rules.d/  udev.conf
```

```
udev/rules.d:  
50-udev.rules           80-libmtp.rules  
60-bluetooth.rules     80-libnjb.rules  
60-pcmcia.rules         80-libpisock.rules  
64-device-mapper.rules  80-libsane.rules  
75-network-devices.rules 90-hal.rules  
75-optical-devices.rules
```

udev zawsze czyta całą konfigurację w poszukiwaniu pasujących do sytuacji reguł. Kilka reguł może się odnosić do tego samego urządzenia i **wszystkie zostaną wykonane**.

Poprawna reguła to co najmniej jedno dopasowanie i jedno podstawienie, np. `KERNEL=="hdb", NAME="zapasowy-dysk"`

Liczby major i minor definiują urządzenia:

```
crw-rw-rw-  1 root audio      14,   3 2007-06-17 11:45 dsp0
crw-rw-rw-  1 root audio      14,  19 2007-06-17 11:45 dsp1
crw-rw-rw-  1 root audio      14,  35 2007-06-17 11:45 dsp2
crw-rw-rw-  1 root audio      14,  51 2007-06-17 11:45 dsp3
```

```
[bash]# mknod dsp4 c 14 67 ; chown root.audio dsp4
```

```
crw-rw-rw-  1 root audio      14,  67 2008-03-06 17:45 dsp4
```

Klucze dopasowań:

BUS typ szyny, np. PCI, USB, SCSI

KERNEL nazwa nadawana przez jądro, np. hdb1, sda2, lp0

ID numer identyfikujący urządzenie (lspci, lsusb)

[bash]# lsusb

```
Bus 002 Device 002: ID 046d:c019 Logitech, Inc.
```

```
Bus 002 Device 001: ID 0000:0000
```

```
Bus 001 Device 002: ID 05e1:0501 Syntek Semiconductor
```

```
Bus 001 Device 001: ID 0000:0000
```

PLACE fizycznie gdzie znajduje się port

SYSFS_plik pliki dające się wyczytać z drzewa sysfs

SYSFS{plik} pliki dające się wyczytać z drzewa sysfs

PROGRAM wywołuje zewnętrzny program lub skrypt...

RESULT ...i dopasowuje wynik jego działania

Klucze podstawień:

NAME	nazwa przypisywana temu urządzeniu
SYMLINK	dodatkowy link symboliczny do tego urządzenia
%k	zastępuje nazwę nadawaną przez kernel (np. hdb)
%n	zastępuje numer nadawany przez kernel
%M, %m	liczba major i minor nadawana przez kernel
%b	identyfikator szyny (bus id)
%c	łańcuch zwracany przez PROGRAM
%2c	liczba wskazuje na drugie słowo w tym łańcuchu
%%	znak %
*	zero lub więcej znaków
?	jeden dowolny znak
[]	jeden z zestawu lub zakresu znaków, np. [abc], [a-c], [!d-z]

Przykłady:

```
KERNEL="[hs]d[a-z]", PROGRAM="name_cdrom.pl %M %m", \  
NAME="%1c", SYMLINK="cdrom"
```

```
# if /sbin/scsi_id returns "OEM 0815" device will  
# be called disk1
```

```
BUS="scsi", PROGRAM="/sbin/scsi_id", \  
RESULT="OEM 0815", NAME="disk1"
```

```
# USB printer to be called lp_color
```

```
BUS="usb", SYSFS_serial="W09090207101241330", \  
NAME="lp_color"
```

SCSI disk with a specific vendor and model number

is to be called boot

```
BUS="scsi", SYSFS_vendor="IBM", \
SYSFS_model="ST336", NAME="boot"
```

sound card with PCI bus id 00:0b.0 to be called dsp

```
BUS="pci", ID="00:0b.0", NAME="dsp"
```

USB mouse at third port of the second hub to

be called mouse1

```
BUS="usb", PLACE="2.3", NAME="mouse1"
```

```
# ttyUSB1 should always be called pda with two
# additional symlinks
KERNEL="ttyUSB1", NAME="pda", \
SYMLINK="palmtop handheld"

# multiple USB webcams with symlinks to be called
# webcam0, webcam1, ...
BUS="usb", SYSFS_model="XV3", NAME="video%n", \
SYMLINK="webcam%n"
```

Sprzęt zazwyczaj jest wstępnie konfigurowany podczas instalacji dystrybucji. Jedne dystrybucje radzą sobie z tym lepiej, inne gorzej...

- grafika – X (karta, monitor, polska klawiatura)
- sieć
- dźwięk – ALSA (alsacfg, alsamixer), OSS
- drukarka – CUPS (localhost:631), lprng, lpd
- skaner – SANE

System graficzny X Window

- XFree86 vs. X.Org
- xfree/xorg jest aplikacją całkowicie oddzieloną od systemu
- można jej nie uruchamiać
- można ją uruchomić w wielu kopiach
- jej awaria (zazwyczaj) nie prowadzi do awarii całego systemu operacyjnego, chociaż czasami dostęp do niego może być niemożliwy
- działa w architekturze klient–serwer, a więc bez interfejsu 1o nie będzie pracować

Słowniczek: **Display Manager**

- pośredniczy między użytkownikiem a systemem i serwerem X
- udostępnia ekran logowania
- wywołuje odpowiednie środowisko graficzne lub menadżer okien
- ma (często) uprawnienia do zarządzania energią (wyłącza, usypia, hibernuje system)

Przykłady:

xdm (X), kdm (KDE), sddm (Plasma), gdm (gnome), lxdm (LXDE), mdm (Mint), SLiM, wdm (Wings)

Słowniczek: **Window Manager**

- aplikacja zarządzająca „oknami”
- pozwala na uruchamianie wielu zadań jednocześnie
- pozwala na zarządzanie obsługiwanyymi procesami
- korzysta z biblioteki graficznej widget’ów i dodatkowych plików
- obsługuje zdarzenia związane z urządzeniami typu mysz i klawiatura
- może obsługiwać zmiany rozdzielczości, współpracę z wieloma urządzeniami wejścia (myszy, klawiatury) i wyjścia (wiele ekranów)

Przykłady:

fvwm, kwin, sawfish, metacity, windowmaker, blackbox, fluxbox...

Słowniczek: **Desktop Environment**

- pulpit, biurko, środowisko graficzne
- zintegrowany pakiet oprogramowania korzystający jednolicie z tych samych bibliotek i mechanizmów
- zawiera zazwyczaj menadżer logowania, menadżer okien, menadżer plików, terminal, program rysujący biurko, panel lub pasek zadań, menu, kosz, edytor tekstu...
- obecnie środowiska DE są tak rozbudowane, że przekrywają swoją konfiguracją konfigurację systemową, umożliwiając pełną pracę „spod nich”

Przykłady:

KDE, gnome, XFCE, Enlightenment, LXDE

Wywołanie „gołego” serwera

```
[bash]# X
```

```
[bash]# Xorg
```

Inicjalizacja serwera

```
[bash]# xinit
```

Szuka pliku `$HOME/.xinitrc` z instrukcjami, jak go nie znajdzie, poszuka pliku globalnego w

```
/var/X11R6/lib/xinit/
```

```
/usr/X11R6/lib/X11/xinit/
```

```
/etc/X11/xinit
```

lub wykona

```
xterm -geometry +1+1 -n login -display :0
```

Jeśli nie podano serwera, szuka pliku `$HOME/.xserverrc` lub użyje

```
X :0
```

```
#!/bin/sh
# $XConsortium: xinitrc.cpp,v 1.4 91/08/22 11:41:34 rws Exp $

userresources=$HOME/.Xresources
usermodmap=$HOME/.Xmodmap
sysresources=/usr/X11R6/lib/X11/xinit/.Xresources
sysmodmap=/usr/X11R6/lib/X11/xinit/.Xmodmap

# merge in defaults and keymaps

if [ -f $sysresources ]; then
    xrbdb -merge $sysresources
fi

if [ -f $sysmodmap ]; then
    xmodmap $sysmodmap
fi
```

```
if [ -f $userresources ]; then
    xrdp -merge $userresources
fi

if [ -f $usermodmap ]; then
    xmodmap $usermodmap
fi

# start some nice programs

twm &
xclock -geometry 50x50-1+1 &
xterm -geometry 80x50+494+51 &
xterm -geometry 80x20+494-0 &
exec xterm -geometry 80x66+0+0 -name login
```

UWAGI i WYJAŚNIENIA

- baza `Xresources` zawiera globalne i lokalne ustawienia dla aplikacji graficznych; obecnie przy powszechnym korzystaniu z DE jest mało wykorzystywana, chyba że ktoś miesza aplikacje
- `startx` działa podobnie jak `xinit` ale pozwala na przekazanie parametrów do serwera graficznego
- `startkde`, `gnome-session` i in. uruchamiają odpowiednie środowiska (można użyć w `.xinitrc` lub zamiast polecenia `startx`)

Konfiguracja X: starsza wersja

- monolityczny plik, np. `/etc/X11/xorg.conf`
- budowa modułarna z podziałem na sekcje: `ServerLayout`, `Screen`, `Device`, `Monitor`, `InputDevice`, `Module`, `Files` i inne
- klawiatura: `pl` lub `pl2`
- monitor: ważne parametry `HorizSync` i `VertRefresh`
- grafika: `vga`, `vesa`, `nv`, `nvidia`, `ati`, `radeon`

Section "Monitor"

```
DisplaySize  331 207
HorizSync     30-62
Identifier    "Monitor[0]"
Option        "DPMS"
VendorName    "CMO"
VertRefresh   43-60
UseModes      "Modes[0]"
```

EndSection

Konfiguracja X: nowsza wersja

- automatyka dzięki evdev
- zalecane jest rozbicie na wiele plików
- główna domyślna konfiguracja w
`/usr/share/X11/xorg.conf.d/*.conf`
- konfiguracja systemowa w
`/etc/X11/xorg.conf.d/*.conf`

```
[bash]# ls -1 /usr/share/X11/xorg.conf.d
```

```
10-evdev.conf
```

```
50-synaptics.conf
```

```
50-vmmouse.conf
```

```
50-wacom.conf
```

```
90-keyboard-layout.conf
```

```
[bash]# cat 10-evdev.conf
```

```
(...)
```

```
Section "InputClass"
```

```
    Identifier "evdev keyboard catchall"
```

```
    MatchIsKeyboard "on"
```

```
    MatchDevicePath "/dev/input/event*"
```

```
    Driver "evdev"
```

```
EndSection
```

```
Section "InputClass"
```

```
    Identifier "evdev touchpad catchall"
```

```
    MatchIsTouchpad "on"
```

```
    MatchDevicePath "/dev/input/event*"
```

```
    Driver "evdev"
```

```
EndSection
```

```
(...)
```



```
[bash]# cat /etc/X11/xorg.conf.d/90-keyboard-layout.conf
```

```
Section "InputClass"
    Identifier "keyboard-all"
    MatchIsKeyboard "on"
    MatchDevicePath "/dev/input/event*"
    Driver "evdev"
    Option "XkbLayout" "pl"
    #Option "XkbVariant" ""
    Option "XkbOptions" "terminate:ctrl_alt_bksp"
EndSection
```

Konfiguracja i obsługa sieci

Interfejsy sieciowe

- muszą być wykryte przez jądro; wpisy w `/proc/net/dev` oraz `/sys/class/net`
- muszą zostać „podniesione” `ifconfig eth0 up`
- zazwyczaj: **eth0** (kabel), **ath0** lub **wifi0** (bezprzewodowa), **lo** (loopback)

```
[bash]# /sbin/ifconfig
```

```
lo          Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:16436  Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

```
ath0 Link encap:Ethernet HWaddr 00:15:AF:12:37:19
    inet addr:192.168.0.109 Bcast:192.168.0.255 Mask:255.255.255.0
    inet6 addr: fe80::215:aff:fe12:3719/64 Scope:Link
    UP BROADCAST NOTRAILERS RUNNING MULTICAST MTU:1500 Metric:1
    RX packets:1065 errors:0 dropped:0 overruns:0 frame:0
    TX packets:1098 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:0
    RX bytes:847616 (827.7 KiB) TX bytes:113068 (110.4 KiB)

wifi0 Link encap:UNSPEC HWaddr 00-15-AF-12-37-19-00-00-00-00-00-00-00-00-00
    UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
    RX packets:458147 errors:0 dropped:0 overruns:0 frame:191407
    TX packets:1953 errors:13 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:199
    RX bytes:48726181 (46.4 MiB) TX bytes:178504 (174.3 KiB)
    Interrupt:20
```

Składnia:

```
ifconfig interface [opcje]
```

Podniesienie interface'u:

```
ifconfig eth0 up
```

Popularne opcje:

address	przypisuje IP
netmask	maska sieciowa
add address/prefixlen	dodaje adres IPv6
del address/prefixlen	usuwa adres IPv6
[-]broadcast adres	ustawia albo kasuje adres rozgłoszeniowy
[-]pointopoint adres	połączenie bezpośrednie dwóch maszyn
hw klasa adres	ustawia adres MAC

Sieć bezprzewodowa jest obsługiwana w identyczny sposób. Dodatkowe aplikacje to:

- `iwconfig` do konfiguracji interface'u sieci bezprzewodowej.

Ustawienie

- ▶ `essid`
 - ▶ `ap`
 - ▶ kanału, częstotliwości, zarządzania energią etc.
- `iwlist` do wyszukiwania sieci bezprzewodowych
- `iwpriv`, `iwspy`, `iwevent`, `iwgetid`
- `wpa_supplicant` i `wpa_gui` do obsługi szyfrowania

Przykład: połączenie szyfrowane z linii komend 1

```
[bash]# wpa_passphrase ESSID HASŁO >> /etc/wpa_supplicant.conf
[bash]# iwconfig ath1 essid ESSID
[bash]# wpa_supplicant -D wext -i ath1 -c /etc/wpa_supplicant.conf
[bash]# dhclient eth1
```

-D sterownik

-i interface

-c plik konfiguracyjny

Plik konfiguracyjny `/etc/wpa_supplicant.conf` zawiera opcje i definicje sieci, np. dla ESSID=SiecBolka i hasła HASŁO

```
ctrl_interface=/var/run/wpa_supplicant
```

```
ctrl_interface_group=root
```

```
network={
```

```
    ssid="SiecBolka"
```

```
    # psk="HASŁO"
```

```
    psk=f6e1c49e15e679bebe385c37648d4141bc5c9297796a (...)
```

```
}
```

Przykład: połączenie szyfrowane z linii komend 2

```
[bash] # wpa_supplicant -B -i wlan0 \  
          -c < (wpa_passphrase ESSID HASŁO)
```

-B praca w tle

-i interface

-c plik konfiguracyjny, tu: tworzony przez wpa_passphrase

W tym przykładzie pomija się tworzenie wpisu w pliku konfiguracyjnym, a więc przetrzymywanie na dysku parametrów sieci (w postaci jawnego tekstu).

W praktyce wygodnie jest korzystać z uprzywilejowanych programów, które zajmują się zestawieniem połączenia.

Wygoda vs. bezpieczeństwo i ingerencja w system:

- menadżery sieci od bardzo prostych do bardzo rozbudowanych
- coraz więcej automatyki, coraz mniej kontroli
 - ▶ automatyczne wyszukiwanie sieci
 - ▶ automatyczne łączenie ze znanymi sieciami
 - ▶ automatyczne nawiązywanie zerwanych połączeń...
 - ▶ ...wykorzystywane przez programy do aktualizacji i podobne
- kłopotliwe, niejednolite nazewnictwo opcji, np. typów szyfrowania
- bardzo silna, często nadmierna, integracja z systemem
- bardzo dużo zależności i powiązań z innymi programami
- duże uprawnienia w systemie – kontrola nad interface'ami sieciowymi
- przechowywanie informacji poufnych – loginy, hasła – często realizowane przez zewnętrzne programy (gnome-keyring, kde wallet...)

Przykładowe menadżery sieci:

- NetworkManager
- WiCD
- Wifi-wiz (python-gtk)
- YaST2 (SuSE)
- getwifi (linia komend)
- wlassistant (KDE)
- wifi-radar (python-pygtk2)
- GTKWifi (Gnome)

Obsługa dźwięku

Sterowniki:

- Open Sound System (OSS) – stare
- Advanced Linux Sound Architecture (ALSA) z emulacją OSS

ALSA obsługuje karty dźwiękowe spełniające standard Intel AC'97 lub Intel HD Audio, w tym produkty Realtek i Creative oraz karty studyjne. Sterowniki ALSA są zintegrowane ze źródłami jądra, ale można zainstalować ich nowszą wersję:

- nowe źródła wgrywane są w strukturę katalogów źródeł jądra
- ALSA jest rejestrowana w konfiguratorach źródeł jądra
- kompilacja tak jak pozostałych sterowników

Urządzenia odpowiedzialne za dźwięk

```
[bash]# ls -l /dev/snd
```

```
crw-rw---- 1 root audio 116,  7 maj  8 11:29 controlC0
crw-rw---- 1 root audio 116, 16 maj  8 11:29 controlC1
crw-rw---- 1 root audio 116,  6 maj  8 11:29 hwC0D0
crw-rw---- 1 root audio 116, 15 maj  8 11:29 hwC1D0
crw-rw---- 1 root audio 116, 14 maj  8 11:29 hwC1D1
crw-rw---- 1 root audio 116, 13 maj  8 11:29 hwC1D2
crw-rw---- 1 root audio 116, 12 maj  8 11:29 hwC1D3
crw-rw---- 1 root audio 116,  5 maj  8 11:29 pcmC0D0c
crw-rw---- 1 root audio 116,  4 maj  8 12:20 pcmC0D0p
crw-rw---- 1 root audio 116,  3 maj  8 11:29 pcmC0D1p
crw-rw---- 1 root audio 116,  2 maj  8 11:29 pcmC0D2c
crw-rw---- 1 root audio 116, 11 maj  8 11:29 pcmC1D3p
crw-rw---- 1 root audio 116, 10 maj  8 11:29 pcmC1D7p
crw-rw---- 1 root audio 116,  9 maj  8 11:29 pcmC1D8p
crw-rw---- 1 root audio 116,  8 maj  8 11:29 pcmC1D9p
crw-rw---- 1 root audio 116,  1 maj  8 2014 seq
crw-rw---- 1 root audio 116, 33 maj  8 11:29 timer
```

Urządzenia audio to: karta dźwiękowa, modem telefoniczny, niektóre karty sieciowe, karta graficzna z wyjściem HDMI itp. Inne urządzenia, np. CD-ROM, DVD-ROM, BD, mikrofon, słuchawki, line-in, podłączone do karty dźwiękowej, są obsługiwane pośrednio przez jej sterownik.

alsacnf

alsamixer

alsactl store

alsactl restore

alsactl names

wykrywa i próbuje skonfigurować karty muzyczne

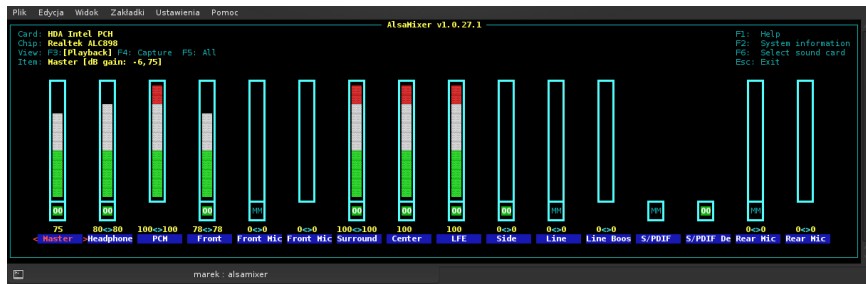
ustawia głośność i inne parametry

zapisuje bieżące ustawienia /etc/asound.state

odtwarza zapisane ustawienia

tworzy listę dostępnych urządzeń
(ctl, pcm, rawmidi, seq, timer,...)

/etc/asound.names



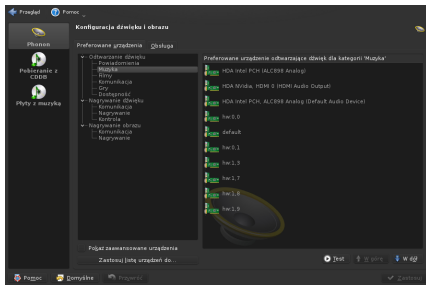
Oprócz samych sterowników, coraz częściej ostatnio używany jest **serwer proxy dla dźwięku**:

- warstwa pośrednia między aplikacjami a sterownikami
- zarządza dźwiękiem z różnych źródeł w jednolity sposób
- miksowanie i filtry
- monitoruje i obsługuje wejścia i wyjścia
- może przekierować dźwięk z dowolnej aplikacji do innej lub na dowolne wyjście
- przesyła dźwięk przez sieć

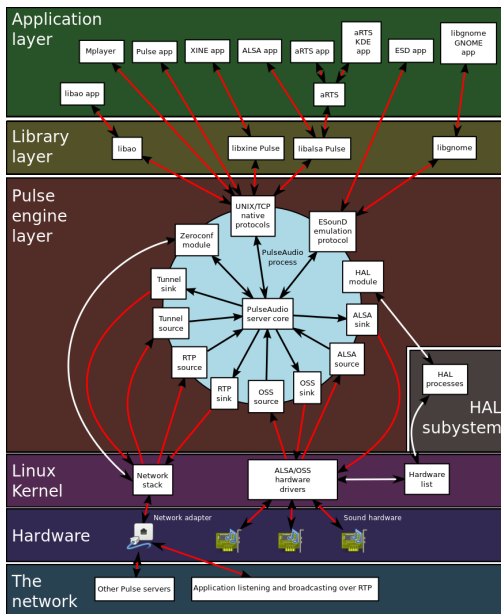
Phonon

- serwer dźwięku zintegrowany z KDE
- aplikacje współpracujące z phononem deklarują typ dźwięku, jaki wysyłają
- phonon obsługuje dźwięk po jego typie: muzyka, film, gra...
- obsługa urządzeń poprzez bibliotekę Solid (część KDE)
- aplikacje napisane dla innych serwerów mogą porozumiewać się z phononem poprzez tzw. backend'y (obsługuje w ten sposób: GStreamer, VLC, MPlayer, Xine, DirectShow9...)

Odpowiednikiem dla Gnome'a jest **GStreamer**.



Innym popularnym serwerem proxy dla dźwięku jest PulseAudio



Drukarki

- istnieje kilka systemów druku: lprng, lpd, cups
- najbardziej obecnie popularny to Common Unix Printing System (cups)
- konfiguracja przez interface w html pod adresem `localhost:631`
- dodatkowe pliki sterowników można znaleźć w repozytoriach, m.in. www.openprinting.org
- wielu producentów wspiera Linuksa, m.in. HP, Samsung, Canon...

Główne pliki znajdują się w `/usr/lib/cups`

- `backend` zawiera interface'y do urządzeń i protokołów różnych typów (ipp w tym ipps, http, httpd i smb, bluetooth, lpd, parallel, serial, snmp, socket, usb)
- `daemon` zawiera pliki wykonywalne aplikacji
- `driver` i `filter` to sterowniki

Własne pliki sterowników wgrywa się do `/etc/cups/ppd/`

Konfiguracja demona w `/etc/cups/*.conf`

Tuning systemu czyli Pimp My Linux

Zmienne środowiskowe i ustawienia powłoki

- /etc/bashrc
- /etc/profile
- /etc/skel

Ustawienia aplikacji graficznych

- Xresources, Xdefaults
`[bash]# cat ~/.Xresources`

```
xterm*background: black
```

```
xterm*foreground: grey
```

```
xterm*font: *-fixed-medium-r-normal-*--120-*--*--iso8859-2
```

```
[bash]# xrdp load /etc/X11/xinit/Xdefaults
```

```
[bash]# xrdp merge ~/.Xresources
```

Niektóre ustawienia ekranu (jasność, rozdzielczość, wiele monitorów...) można regulować z linii komend poprzez polecenie `xrandr`.

Część ustawień serwera X dostępna jest poprzez polecenie `xset`, w tym oszczędzanie energii (wygaszanie monitora), czułość myszy, kontrola diod na klawiaturze, bell i inne.

Obecnie bardzo wiele ustawień administracyjnych można przeprowadzić spod paneli kontrolnych środowisk graficznych

- Centrum Sterowania KDE: `kcontrol` (KDE3), `systemsettings` (KDE4), `systemsettings5` (Plasma)
- Gnome: `gnome-control-center`
- XFCE: `xfce4-settings`

- narzędzie `locale`, zmienne `LANG`, `LC_ALL=pl_PL`

```
[bash]# locale
```

```
LANG=pl_PL.utf8
```

```
LC_CTYPE="pl_PL.utf8"
```

```
LC_NUMERIC="pl_PL.utf8"
```

```
LC_TIME="pl_PL.utf8"
```

```
LC_COLLATE="pl_PL.utf8"
```

```
LC_MONETARY="pl_PL.utf8"
```

```
LC_MESSAGES="pl_PL.utf8"
```

```
LC_PAPER="pl_PL.utf8"
```

```
LC_NAME="pl_PL.utf8"
```

```
LC_ADDRESS="pl_PL.utf8"
```

```
LC_TELEPHONE="pl_PL.utf8"
```

```
LC_MEASUREMENT="pl_PL.utf8"
```

```
LC_IDENTIFICATION="pl_PL.utf8"
```

```
LC_ALL=pl_PL.utf8
```

- czcionki UTF-8 `unicode_start`, `unicode_stop` lub `iso-8859-2` (`Latin2`)
- UWAGA: nowe jądra standardowo obsługują unicode w konsolach. Aby to wyłączyć, należy dodać przy bootowaniu opcję do jądra `vt.default-utf8=0`
- czcionka konsolowa `Latin-2`:
`[bash]# setfont lat2a-16`
Czcionki standardowo znajdują się w `/usr/share/kbd/consolefonts`
- klawiatura w konsoli:
`[bash]# loadkeys pl2`
Mapowania klawiatury standardowo znajdują się w `/usr/share/kbd/keymaps`

UWAGA: ważna jest kolejność wydawania poleceń, tzn. ustawienie trybu, wgrania czcionki i ustawienia klawiatury!

Komenda `setfont` ustawia czcionkę.

Komenda `kbd_mode -u` przełącza klawiaturę w tryb unicode.

Komenda `unicode_start` jest silniejsza niż obie powyżej.

Ustawienie czcionki i klawiatury w trybie UTF8:

```
unicode_start Lat2-Terminus16
```

```
loadkeys pl2
```

Linux – cz. 4

- Pakietologia: rpm, deb, tgz
- Kompilowanie programów ze źródeł
- Kompilowanie jądra
- Jak się zabezpieczać przed nieplanowanym...

Pakiety RPM – Redhat Package Manager to pierwszy wprowadzony w linuxie system zarządzania oprogramowaniem bez konieczności jego kompilowania. Stworzony przez firmę Redhat dla dystrybucji Redhat, później pojawia się w Mandrake/Mandriva, SuSE, PLD i innych.

- Archiwum spakowane cpio, skrypty pomocnicze i opisujące atrybuty plików, wersje etc
- Baza danych o pakietach
- Obsługa zależności
- Upgrade / Downgrade

Z linii komend: **[bash]**# rpm

Istnieje wiele graficznych menadżerów pakietów (frontend'y dla rpm) specyficznych dla danej dystrybucji.

Tryby pracy rpm:

- odpytywanie
- weryfikowanie
- sprawdzanie sygnatury
- instalowanie / aktualizacja / odświeżanie / odinstalowanie
- inicjalizowanie bazy danych
- przebudowa bazy danych
- ponowne podpisanie
- dodanie sygnatury
- ustawienie właścicieli i grup
- pokazanie etykiet zapytań
- pokazanie konfiguracji

INSTALOWANIE I AKTUALIZACJA

Dołożenie nowego pakietu:

```
[bash]# rpm -i|-install [opcje] pakiet.rpm
```

Aktualizacja pakietu (ew. poprzednie wersje pakietu usuwane):

```
[bash]# rpm -U|-upgrade [opcje] pakiet.rpm
```

Odświeżenie pakietu (reinstalacja, poprzednia wersja musi istnieć w systemie):

```
[bash]# rpm -F|-freshen [opcje] pakiet.rpm
```

Użyteczne opcje:

-badreloc	relokuje ścieżki wszystkich plików
-excludepath ŚCIEŻKA	pomija pliki mające pojawić się w ŚCIEŻCE
-excludedocs	pomija dokumentację
-force	= -replacepkg -replacefiles -oldpackage
-ignoresize	nie sprawdza miejsca na dysku docelowym
-justdb	odświeża bazę danych a nie filesystem
-nosignature	pomija sprawdzanie sygnatury
-nodeps	pomija zależności
-nosuggest	nie sugeruje pakietów uzupełniających zależności
-noscripts	pomija skrypty instalacyjne; = -nopre -nopost -nopreun -nopostun

<code>-notriggers</code>	omija skrypty drugiego poziomu; = <code>-nottriggerin -nottriggerun -nottriggerpostun</code>
<code>-oldpackage</code>	zezwala na downgrade
<code>-prefix ŚCIEŻKA</code>	nowa ścieżka przy relokacji pakietu
<code>-relocate STARA=NOWA</code>	zmiana ścieżki instalacji pakietu; korzysta z pliku odpowiedzi dla relokacji (<code>-badreloc</code>)
<code>-replacefiles</code>	zezwala na zastąpienie plików z innych pakietów
<code>-test</code>	symuluje działanie komendy

USUWANIE

```
[bash]# rpm -e|-erase [opcje] pakiet.rpm
```

-allmatches	usuwa wszystkie wersje pakietu
-nodeps	nie sprawdza zależności przy usuwaniu
-noscripts	pomija skrypty
-nopreun	pre-uninstall
-nopostun	post-uninstall
-notriggers	pomija skrypty drugorzędne
-notriggerun	j.w. dla uninstall
-notriggerpostun	j.w. dla post-uninstall
-test	symuluje działanie komendy

ZAPYTANIA

```
[bash]# rpm -q|-query [opcje-wyboru] [opcje-zapytań]
```

Opcje wyboru:

pakiet	odpytuje tylko pakiet
-a -all	odpytuje całą bazę
-whatprovides ...	odpytuje pakiet dostarczający ...
-whatrequires	j.w. wymagający ...

Opcje zapytań:

<code>-c -configfiles</code>	listuje pliki konfiguracyjne
<code>-d -docfiles</code>	listuje dokumentację
<code>-dump</code>	listuje rozszerzone informacje o pliku
<code>-i -info</code>	podstawowe info
<code>-l -list</code>	listuje pliki pakietu
<code>-provides</code>	co pakiet dostarcza?
<code>-R -requires</code>	co pakiet wymaga?
<code>-scripts</code>	listuje skrypty
<code>-s -state</code>	stan pakietu: normalny, niezainstalowany, zastąpiony

WERYFIKACJA

Sprawdza stan faktyczny z informacjami zapisanymi w bazie rpm:

```
[bash]# rpm -V|-verify [opcje-wyboru] [opcje-weryfikacji]
```

PRZEBUDOWA BAZY DANYCH

```
[bash]# rpm -initdb|-rebuilddb [-v] [-dbpath KATALOG] [-root  
KATALOG]
```


RPM posiada wewnętrzną obsługę protokołów **ftp** i **http**. Pakiety mogą być podawane jako adresy URL w/g schematu:

```
ftp://user:pass@host:port/path/path/pakiet.rpm
```

```
http://user:pass@host:port/path/path/pakiet.rpm
```

Jeśli pass jest pominięte to RPM spyta o hasło.

GDZIE TO WSZYSTKO SIEDZI NA DYSKU?

Konfiguracja rpmrc

/usr/lib/rpm/rpmrc

/usr/lib/rpm/redhat/rpmrc

/etc/rpmrc

~/.rpmrc

Konfiguracja makr

/usr/lib/rpm/macros

/usr/lib/rpm/redhat/macros

/etc/rpm/macros

~/.rpmmacros

Baza danych

/var/lib/rpm/Basenames

/var/lib/rpm/Conflictname

/var/lib/rpm/Dirnames

/var/lib/rpm/Filemd5s

/var/lib/rpm/Group

/var/lib/rpm/Installtid

/var/lib/rpm/Name

/var/lib/rpm/Packages

/var/lib/rpm/Providename

/var/lib/rpm/Provideversion

/var/lib/rpm/Pubkeys

/var/lib/rpm/Removed

/var/lib/rpm/Requirename

/var/lib/rpm/Requireversion

/var/lib/rpm/Shalheader

/var/lib/rpm/Sigmd5

/var/lib/rpm/Triggername

Zobacz też pliki/skrypty:

- `rpme`, `rpmi`, `rpmu`, `rpmquery`
- `rpm2cpio`, `rpm2targz`, `rpm2tgz`
- `rpmbuild`
- `rpmdb`
- `rpmverify`, `rpmsign`, `rpmoffset`, `rpmgraph`

Pakiety DEB opracowane dla Debiana z myślą o prowadzeniu zautomatyzowanych repozytoriów sieciowych, zgodnie z filozofią rozwoju tej dystrybucji. Można je znaleźć we wszystkich klonach Debiana, w tym Ubuntu, Knoppix etc.

- Archiwum, wewnątrznie pakowane tar + gz
- Obsługa bazy danych o pakietach
- ...zależnościach, aktualizacjach,
- ...konfigurowanie pakietów
- Repozytoria sieciowe

Natywny menadżer pakietów w Debianie: `dpkg`

`[bash]# dpkg [opcje] akcja`

Możliwe stany pakietów:

- installed
- half-installed
- not-installed
- unpacked
- half-configured
- config-files

INSTALOWANIE

```
[bash]# dpkg -i|-install pakiet.deb
```

- Rozpakowanie plików kontrolnych
- Jeśli inna wersja pakietu istnieje w systemie, uruchamia jej skrypt `prerm`
- Uruchomienie skryptu `preinst`
- Rozpakowanie nowych plików, tworzenie kopii zapasowych poprzednich
- Jeśli inna wersja pakietu istniała w systemie, uruchomienie jej skryptu `postrm`
- Konfigurowanie nowego pakietu

ROZPAKOWANIE

```
[bash]# dpkg -unpack pakiet.deb
```

KONFIGUROWANIE

```
[bash]# dpkg -configure pakiet.deb
```

```
[bash]# dpkg -configure -a|-pending
```

Konfiguruje jeden lub wszystkie rozpakowane pakiety

- Wypakowanie plików konfiguracyjnych, tworzenie kopii zapasowych poprzednich
- Uruchomienie skryptu `postinst`

USUWANIE

```
[bash]# dpkg -r|--remove pakiet.deb
```

```
[bash]# dpkg -r|--remove -a|--pending
```

Usuwa jeden lub wszystkie zaznaczone do usunięcia
(/var/lib/dpkg/status) pakiety, zostawiając pliki konfiguracyjne.

```
[bash]# dpkg -r|--remove -P|--purge pakiet.deb
```

```
[bash]# dpkg -r|--remove -P|--purge -a|--pending
```

J.w. ale usuwa również pliki konfiguracyjne.

- Uruchomienie skryptu prerm
- Usunięcie plików
- Uruchomienie skryptu postrm

Podstawowe opcje:

`-no-act -dry-run -simulate`

`-R|-recursive` ścieżka działa na katalogu z pakietami

Pliki konfiguracyjne:

`/etc/dpkg/dpkg.conf`

`/var/lib/dpkg/available`

`/var/lib/dpkg/status`

Inne programy z tej grupy:

`dpkg-deb, deb, deb-control, dpkg-reconfigure, dpkg-query, dpkg-split`

Zarządzanie pakietami: apt-get

[bash]# apt-get akcja opcje pakiet.deb

Akcje:

update	synchro indeksów pakietów z ich źródłami (/etc/apt/sources.list)
upgrade	
dselect-upgrade	j.w. w połączeniu z aplikacją dselect
dist-upgrade	jak upgrade, ale rozwiązuje problem zależności poświęcając mniej ważne pakiety
install	
remove	
source	pobranie pakietu ze źródłami
build-dep	instalacja/usuwanie pakietów aby umożliwić kompilację pakietu źródłowego
check	diagnostyka
clean	czyści lokalnego cache'a pobranych pakietów (/var/cache/apt/archives, /var/cache/apt/archives/partial)

Niektóre opcje:

`-d|-download-only`

bez instalacji

`-no-download`

tylko lokalne pliki

`-q|-quiet`

bez komunikatów

`-s|-simulate|-dry-run|-no-act`

symulacja

`-y|-yes|-assume-yes`

TAK na wszystko

`-b|-compile|-build`

pobranie i kompilacja

`-no-upgrade`

bez aktualizacji

Podstawowe narzędzie do zarządzania pakietami: dselect

The screenshot shows a terminal window titled 'ubuntu@ubuntu: ~' with a menu bar (File, Edit, View, Terminal, Tabs, Help). The terminal displays the 'Debian `dselect` package handling frontend.' with a list of options: 0. [A]ccess, 1. [U]pdate, 2. [S]elect, * 3. [I]ninstall, 4. [C]onfig, 5. [R]emove, and 6. [Q]uit. Option 3 is highlighted. Below the list, instructions for navigation and confirmation are provided. The version (1.13.11) and copyright information (1994-1996 Ian Jackson, 2000,2001 Wichert Akkerman) are shown, along with a notice about the GNU General Public License. At the bottom, it states 'Read-only access: only preview of selections is available!'. The terminal window is part of a desktop environment with a taskbar at the bottom showing 'ubuntu@ubuntu: ~', 'DEB - File Browser', and 'Starting Take Screenshot'.

```

Applications Places System
ubuntu@ubuntu: ~
File Edit View Terminal Tabs Help
Debian `dselect' package handling frontend.

0. [A]ccess    Choose the access method to use.
1. [U]pdate    Update list of available packages, if possible.
2. [S]elect    Request which packages you want on your system.
* 3. [I]ninstall Install and upgrade wanted packages.
4. [C]onfig    Configure any packages that are unconfigured.
5. [R]emove    Remove unwanted software.
6. [Q]uit      Quit dselect.

Move around with ^P and ^N, cursor keys, initial letters, or digits;
Press <enter> to confirm selection.  ^L redraws screen.

Version 1.13.11 (i386).
Copyright (C) 1994-1996 Ian Jackson.
Copyright (C) 2000,2001 Wichert Akkerman.
This is free software; see the GNU General Public Licence version 2
or later for copying conditions. There is NO warranty.  See
dselect --licence for details.

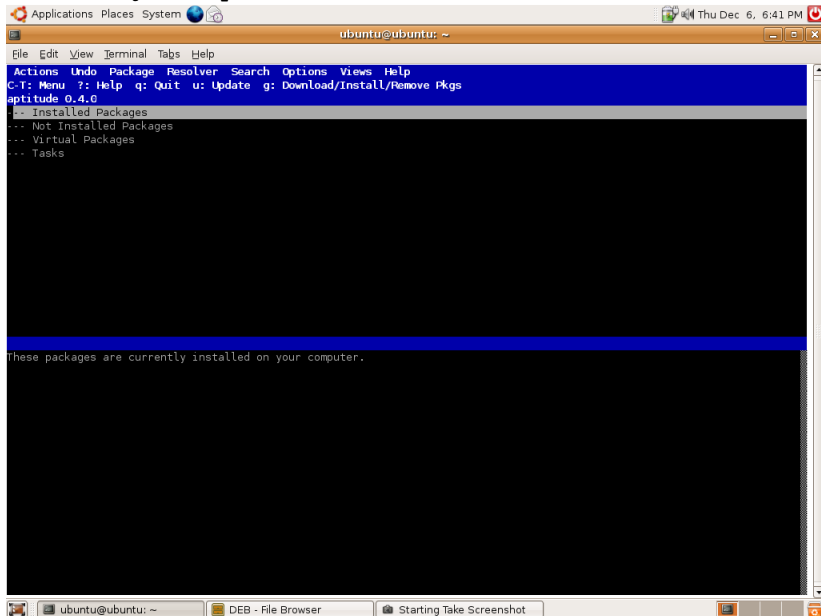
Read-only access: only preview of selections is available!

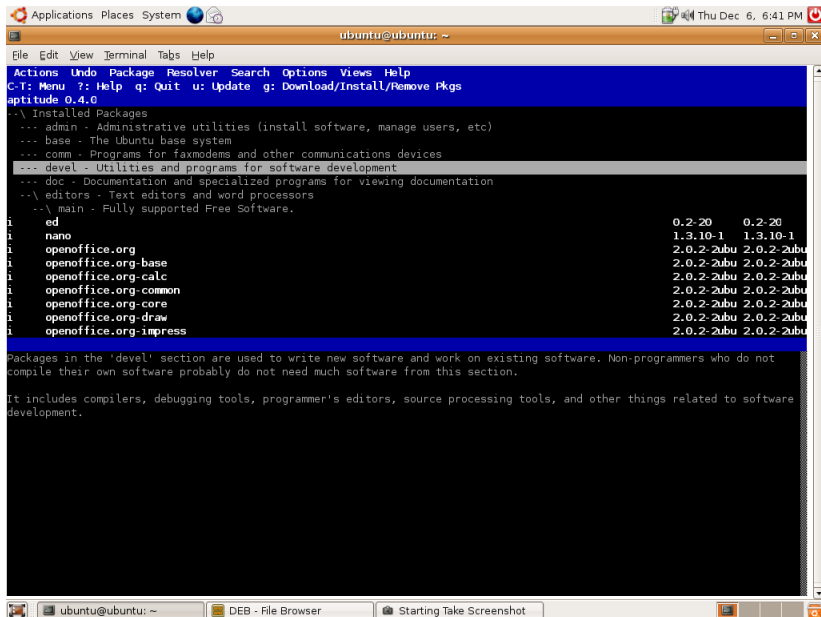
ubuntu@ubuntu: ~  DEB - File Browser  Starting Take Screenshot

```

A screenshot of a Linux terminal window titled "ubuntu@ubuntu: ~". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The terminal shows the command "dselect - inspection of package states (avail., priority)" being executed. Below this, there's a table listing various packages like Xtr otherosf, python-gd, python-pisoc, python2.4-gd, python2.4-py, sound festvox-kall, text foomatic-fil, utils ethtool, inputattach, slocate, x11 ttf-freefont, and unclassified packages. The user has selected "python-gd" and its description is shown: "Provides PNG, JPEG, and XPM creation and manipulation routines through the libgd library. Also allows use of TrueType fonts in images created. This Debian package contains only documentation and always depends on a package built for Debian's default Python version." The bottom status bar shows the terminal icon, the prompt "ubuntu@ubuntu: ~", and two active tabs: "DEB - File Browser" and "Starting Take Screenshot".

Podobne narzędzie: aptitude






```

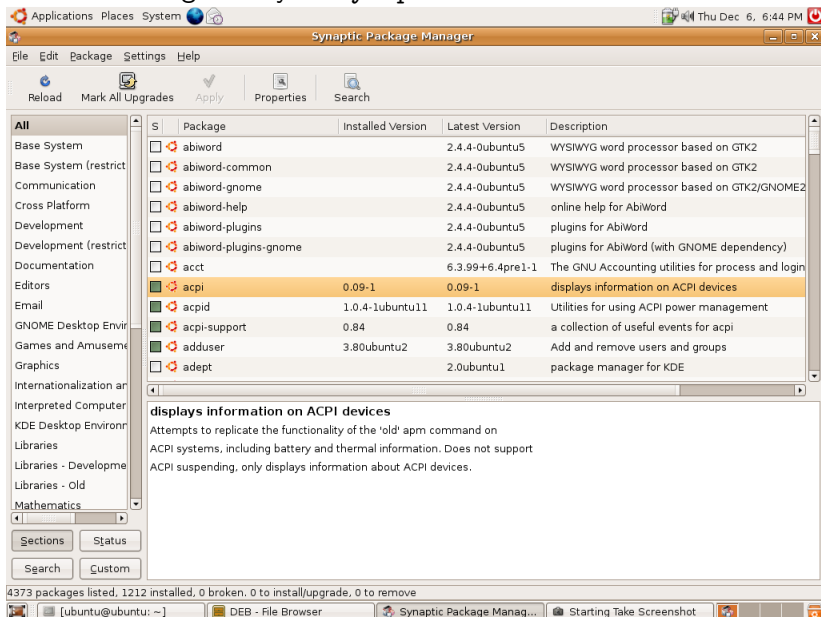
ubuntu@ubuntu: ~
File Edit View Terminal Tabs Help
Actions Undo Package Resolver Search Options Views Help
C-T: Menu ? : Help q: Quit u: Update g: Download/Install/Remove Pkgs
Packages vim info
aptitude 0.4.6
1 --\ vim 1:6.4-006+ 1:6.4-006+
Description: Vi IMproved - enhanced vi editor
Vim is an almost compatible version of the UNIX editor Vi.

Many new features have been added: multi level undo, syntax highlighting, command line history, on-line help, filename
completion, block operations, folding, Unicode support, etc.

This package contain a version of vim compiled with a rather standard set of features. See the other vim-* packages if
you need more (or less).
Priority: important
Section: editors
Maintainer: Debian Vim Maintainers <pkg-vim-maintainers@lists.alioth.debian.org>
Compressed size: 554k
Uncompressed size: 1126k
Source Package: vim
--\ Depends
--- libc6 (>= 2.3.4-1)
--- libgpmgl (>= 1.19.6-1)
--- libncurses5 (>= 5.4-5)
--- vim-common (= 1:6.4-006+2ubuntu6)
--- vim-runtime (= 1:6.4-006+2ubuntu6)
--\ Suggests
--- ctags (UNSATISFIED)
--- vim-doc (UNSATISFIED)
--- vim-scripts (UNAVAILABLE)
--\ Conflicts
--- vim-common (< 1:6.4-001+3)
--- vim-full (< 1:6.4-001+3)
--- vim-gnome (< 1:6.4-001+3)
--- vim-gtk (< 1:6.4-001+3)
--- vim-lesstif (< 1:6.4-001+3)
--- vim-perl (< 1:6.4-001+3)
--- vim-python (< 1:6.4-001+3)
--- vim-rt
Vi IMproved - enhanced vi editor

```

Jedna z nakładek graficznych: synaptic



Najprostszą metodą stworzenia „pakietu” jest spakowanie wraz ze ścieżkami skompilowanego programu. Tak jest w Slackware i jego klonach.

- Pakowane za pomocą `tar` i `gzip`
- Pakowana jest cała ścieżka
- Jeden dodatkowy katalog `install` z plikami `doinstall.sh` i `slack-desc`
- Często dodatkowy plik z sygnaturą MD5 do weryfikacji
- Brak zależności

Schemat nazywania pakietów:

nazwa-wersja-arch-numer_builder.tgz

claws-mail-2.10.0-i486-1mg.tgz

gimp-2.2.17-i486-2_slack12.0.tgz

Od Slackware 13.0 zamiast gzip używany jest inny kompresor: xz. Pakiety zmieniły się na *.txz

Jak przekonwertować na stary format?

```
[bash]# mv pakiet.txz pakiet.tar.xz  
[bash]# xz -d pakiet.tar ; gzip pakiet.tar  
[bash]# mv pakiet.tar.gz pakiet.tgz
```

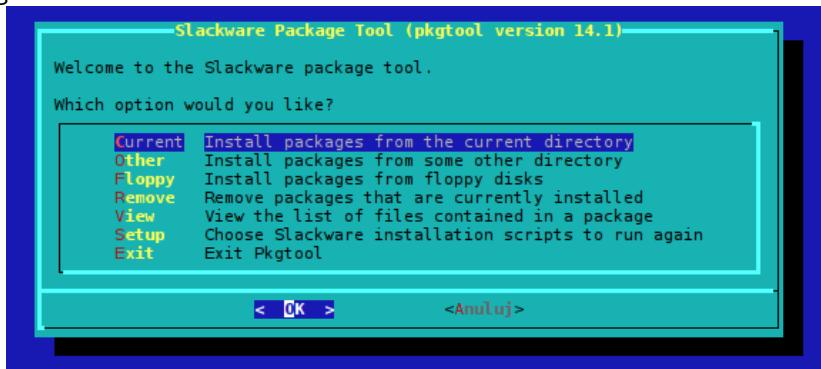
Narzędzia do zarządzania pakietami tgz

- pkgtool
- installpkg, removepkg, upgradepkg
- explodepkg, makepkg

Istnieją programy sprawdzające sieciowe repozytoria i zależności na wzór rpm i deb:

- swaret
- slackpkg

Pkgtool:



Pliki na dysku (slackware):

`/var/log/packages`

`/var/log/removed_packages`

`/var/log/removed_scripts`

`/var/log/scripts`

Wolne oprogramowanie jest rozprowadzane albo jako gotowe binarki (często z instalatorami), np. Adobe Reader, Firefox..., albo jako kody źródłowe do samodzielnej kompilacji, np. Firefox, większość sterowników...

KOMPILACJA ŹRÓDEŁ

Źródła zazwyczaj napisane są w C/C++, ale równie dobrze mogą to być programy w Pythonie, Tcl/Tk, Ruby, Java, Perl,...

- Konieczne jest posiadanie kompilatora, parserów...
- Podstawowy zestaw deweloperski: automake, autoconf, bison, byacc, binutils, db, flac, flex, gcc, pkgconfig, pmake, util-linux...
- Biblioteki wraz z plikami nagłówkowymi (`*/include/*.h`); pakiety `*-dev`

- Rozpakowanie źródeł:
`[bash]# tar xzf sources.tgz`
- CZYTAMY INSTRUKCJĘ, DOKUMENTACJĘ I WSZYSTKO CO SIĘ DA!
- Sprawdzenie zależności:
`[bash]# configure`
- Standardowa procedura:
`[bash]# make ; make install`

Polecenie `configure` ma często wiele opcji

```
[bash]# configure -help  
-prefix=nowa/ścieżka  
-enable-cośtam  
-disable-cośtam  
-with-cośtam
```

Polecenie `make install` zakłada skompilowany program

- musi być wykonane przez roota, chyba że instalacja lokalna
- można zbudować własny pakiet zgodnie z używaną dystrybucją

`[bash]# make install DESTDIR=ścieżka`

MINIMALNE WYMAGANIA DLA KOMPILACJI JĄDRA LINUX 3.x-4.x

GNU C	3.2	# gcc --version
GNU make	3.80	# make --version
binutils	2.12	# ld -v
util-linux	2.10o	# fdformat --version
module-init-tools	0.9.10	# depmod -V
e2fsprogs	1.41.4	# e2fsck -V
jfsutils	1.1.3	# fsck.jfs -V
reiserfsprogs	3.6.3	# reiserfsck -V
xfsprogs	2.6.0	# xfs_db -V
squashfs-tools	4.0	# mksquashfs -version
btrfs-progs	0.18	# btrfsck

MINIMALNE WYMAGANIA DLA KOMPILACJI JĄDRA LINUX 3.x-4.x

pcmciautils	004	# pccardctl -V
quota-tools	3.09	# quota -V
PPP	2.4.0	# pppd --version
isd4k-utils	3.1pre1	# isdnctrl 2>&1 grep version
nfs-utils	1.0.5	# showmount --version
procps	3.2.0	# ps --version
oprofile	0.9	# oprofiled --version
udev	081	# udevd --version
grub	0.93	# grub --version
mcelog	0.6	# mcelog --version
iptables	1.4.2	# iptables -V
openssl, libcrypto	1.0.0	# openssl version
bc	1.06.95	# bc --version

Architektury, na których można skompilować Linux 4.x:

- **x86**, Compaq Alpha AXP, Sun SPARC, Sun UltraSPARC, Motorola 68000, **PowerPC**, **PowerPC64**, **ARM**, Hitachi SuperH, Cell, IBM S/390, MIPS, HP PA-RISC, **Intel IA-64**, DEC VAX, **AMD x86-64**, AXIS CRIS, Xtensa, Tiler TILE, AVR 32, ARC, Renesas M32R

Ogólne wymagania dotyczące architektury:

- stronicowanie pamięci (PMMU – paged memory management unit); można nie respektować tego warunku ograniczając funkcjonalność
- działający port gcc

Linux doczekał się portu na Linuksa. Można uruchomić kernel jako aplikację w przestrzeni użytkownika, tzw. UserMode Linux (UML).

Kompilacja kernela 2.4.x

```
# Czyścimy stare obiekty .o
make mrproper
```

```
# Konfiguracja --> jedna z poniższych metod
make config          # bash
make menuconfig      # dialog/ncurses
make xconfig         # graficzne menu QT
make oldconfig       # ładuje istniejący ./config
```

```
# Zależności
make dep
```

```
# Kompilacja jądra
make bzImage
make modules
make modules_install
```

Kompilacja kernela 2.6.x - 4.x

Ew. nakładamy łaty --> o tym za chwilę

Czyścimy stare obiekty .o

```
make mrproper
```

Konfiguracja --> jedna z poniższych metod

```
make config          # txt
```

```
make nconfig         # txt + kolorki
```

```
make menuconfig      # dialog/ncurses
```

```
make xconfig         # graficzne menu (Qt)
```

```
make gconfig         # graficzne menu (GTK+)
```

```
make oldconfig       # ładuje istniejący ./config
```

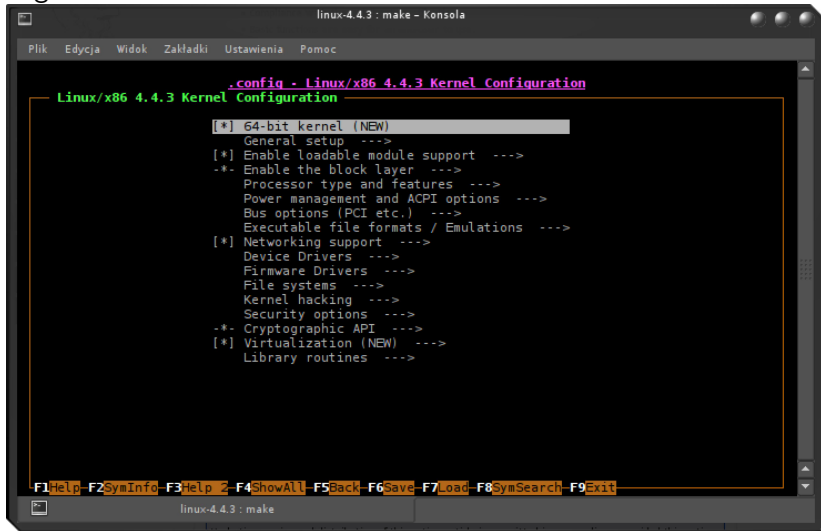
```
make allmodconfig    # wszystko w modułach
```

Kompilacja jądra

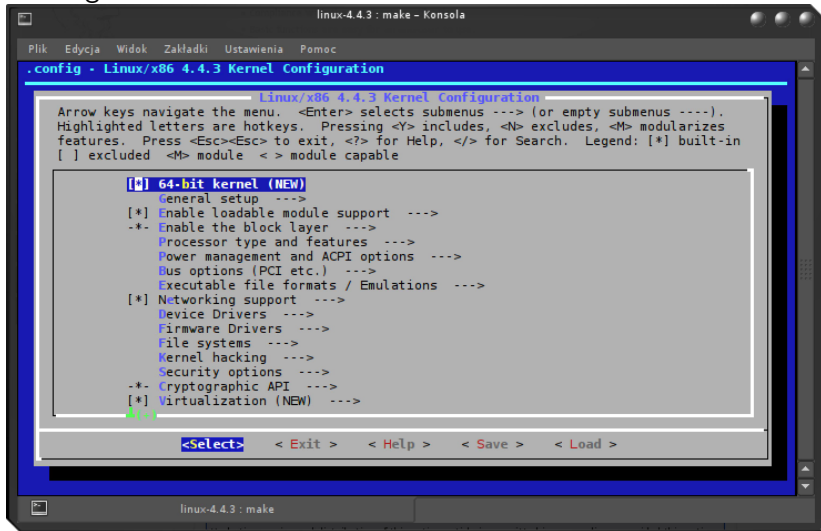
```
make                 # jądra szukaj w arch/x86/boot/bzImage
```

```
make modules_install
```

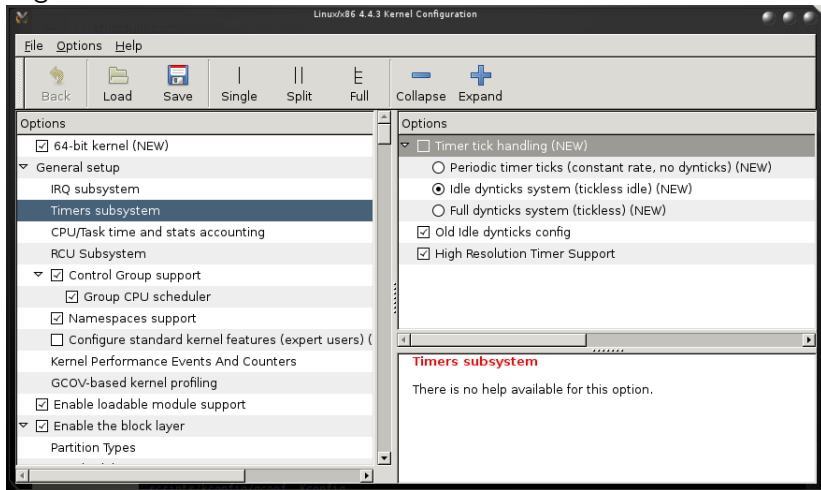
nconfig:



menuconfig:



gconfig:



Łatki:

- zmieniają wersję jądra (zamiast ściągać nowe źródła)
- poprawiają sterowniki (obsługa nowego sprzętu)
- usuwają błędy, zamykają dziury

Opis: plik **README** oraz **Documentation/applying-patches.txt**

Zakładanie łatek (patch) na jądra 2.6.x:

```
[bash]# zcat patch-2.6.xx.gz | patch -p1
```

```
[bash]# bzip2 -cd patch-2.6.xx.bz2 | patch -p1
```

Istnieje też skrypt automatyzujący nakładanie łatek:

```
[bash]# linux/scripts/patch-kernel linux
```

Łaty zmieniające wersję jądra 4.x

- zmiany w obrębie głównych wydań muszą być aplikowane w kolejności, np. 4.1 → 4.2 → 4.3 → 4.4
- zmiany w obrębie podwersji nakładane są licząc od głównego wydania, czyli zmiana 4.4.2 na 4.4.3 wymaga
 - ▶ cofnięcia wersji 4.4.2 → 4.4 (polecenie `patch -R`)
 - ▶ patchowanie 4.4 → 4.4.3
- dla jąder 4.x patche są pakowane w formacie .xz

```
[bash]# xz -cd ../4.x.xz | patch -p1
```

Podczas nakładania łat należy pamiętać o wydaniu komendy z poprawnego katalogu (źródła jądra). Patche powinno się umieścić w katalogu nadrzędnym.

Patche mogą obejmować nie tylko zmianę wersji, ale również scalenie źródeł jądra z nowym kodem, np. nową wersją ALSA, iptables, sterownikiem reiserfs4 etc.

Zmiana katalogu kompilacji

Źródła: **/usr/src/linux-2.6.x**

lokalnie kompilowane w: **/home/marek/kernel**

```
cd /usr/src/linux-2.6.x
```

```
make O=/home/marek/kernel menuconfig
```

```
make O=/home/marek/kernel
```

```
sudo make O=/home/marek/kernel modules_install install
```

Instalacja nowego jądra

Nowe moduły znajdują się (po `make modules_install`) w:
`/lib/modules/4.x.x/`

Nowoskompilowane jądro znajduje się w:

<code>arch/i386/boot/bzImage</code>	wersja 2.6.x
<code>arch/x86/boot/bzImage</code>	wersja > 2.6.x

Należy je umieścić w `/boot` i odpowiednio skonfigurować bootloader (grub, lilo).

Parę uwag o konfiguracji jądra

- konfiguracja jądra jest przechowywana w pliku `.config`, który jest tworzony przez konfigurator
- można konfigurację zapisać też w innym pliku i załadować ją do konfiguratora
- fragment przykładowego pliku (not `set/y/n/m`)

```
#  
# Automatically generated file; DO NOT EDIT.  
# Linux/x86 3.10.17 Kernel Configuration  
#  
# CONFIG_64BIT is not set  
CONFIG_X86_32=y  
CONFIG_X86=y  
CONFIG_INSTRUCTION_DECODER=y  
CONFIG_OUTPUT_FORMAT="elf32-i386"  
CONFIG_ARCH_DEFCONFIG="arch/x86/configs/i386_defconfig"  
CONFIG_LOCKDEP_SUPPORT=y
```

- opcje podzielone są na normalne oraz oznaczone etykietami `development`, `experimental`, `debugging`, `obsolete`; zazwyczaj etykietowanych opcji nie ma potrzeby włączać
- niektóre części kodu można umieścić w modułach – moduły są ładowane przez działające jądro lub przez administratora

Uwaga 1

Sterowniki potrzebne w trakcie startu systemu muszą być wkompiłowane na stałe lub umieszczone w ramdysku. Dotyczy to m.in. sterownika systemu plików głównej partycji, czasami karty sieciowej (przy bootowaniu przez sieć), podstawowego sterownika ekranu etc.

Uwaga 2

Kodu wkompiłowanego nie można unieczynnić, natomiast moduły można wyładować z pamięci w trakcie działania systemu. Ze względów bezpieczeństwa wkompiłowane powinny być m.in. `netfilter`, procedury kryptograficzne itp.

Opcje, które muszą być włączone w typowych zastosowaniach.

- SMP z odpowiednią liczbą obsługiwanych procesorów
- IRQ
- RCU (Read-Copy-Update), aby używane pliki były dostępne dla aplikacji
- IKM (kernel .config support)
- opcja FUTEX (fast userspace mutex) – dba, by dwa wątki nie otrzymały jednocześnie dostępu do zasobu, który jest w stanie obsłużyć tylko jedno zapytanie; musi być uruchomiona, jeśli system opiera się na glibc
- opcja MODULES – uruchamia obsługę modułów
- opcja BLOCK – uruchamia urządzenia blokowe
- opcja X86_MCE (machine check/overheating) – raportowanie o błędach i przegrzaniu
- opcja ELF

- sieć musi być uruchomiona przynajmniej w minimalnym zakresie, tj. loopback, TCP/IP i obsługa gniazd (`Unix domain sockets`) – gniazda używane są do komunikacji międzyprocesowej przez m.in. `X window system`

Można konfigurować system minimalny nie obsługujący sieci, wtedy tylko loopback jest potrzebny.

Przegląd ważniejszych i ciekawszych opcji (Linux 3.9.x).

- cross-compiling – jądro można skompilować dla architektury docelowej innej niż platforma, na której następuje kompilacja
- opcja LOCALVERSION – możliwość dodania do nazwy jądra własnego identyfikatora; pozwala mieć kilka funkcjonalnych kompilacji tej samej wersji jądra
- metoda kompresji kompilatu – obsługiwane są: gzip (domyślne), bzip2, lzma, xz, lzo oraz brak kompresji
- opcja SWAP – jest konieczna, jeśli ma być uruchomiona hibernacja systemu
- zegar (timers subsystem) – wpływa na responsywność systemu poprzez różne modele dzielenia czasu procesora między procesy; bardzo ważne w systemach czasu rzeczywistego i zastosowaniach o niskim opóźnieniu (low latency systems), np. w studyjnych systemach obróbki multimedialnych

- opcja `SYSCTL_SYSCALL` – uruchamia wywołania systemowe poprzez interface `/proc/sys`
- opcja `PRINTK` – pozwala na generowanie i logowanie komunikatów jądra
- opcja `COMPAT_BRK` – losowość sterty; poprawia bezpieczeństwo, ale nie działa z `libc5`

Obecnie używa się `libc6`, która identyfikowana jest jako `glibc 2.x`. Poprzednia wersja `libc5` była przestarzała dla GNU/Linux już 10 lat temu, ale nie wszystkie systemy działają w oparciu o GNU...

- typ procesora – uzgodnić ze sprzętem i resztą systemu (32b/64b); opcje jak `HyperThreading` itp. w zależności od sprzętu
- `DMA` (Direct Memory Access) – część urządzeń z tego korzysta
- opcje `PREEMPT_...` (preemption model) – model zawieszania i wznowiania procesów; trzy wersje rekomendowane dla komputerów typu `server`, `desktop` i `low-latency desktop`

- opcja EFI – tylko dla systemów wyposażonych w EFI/UEFI zamiast tradycyjnego BIOSa; aby działało należy mieć później również odpowiednią wersję bootloadera
- opcja KEXEC – pozwala na zatrzymanie i wznowienie pracy lub podmianę jądra bez zamykania systemu (komendą `kexec` z pakietu `kexec-tools` → www.kernel.org/pub/linux/utils/kernel/kexec)
- ACPI (Advanced Configuration and Power Interface) – bardzo wiele opcji dotyczących oszczędzania energii i sterowania urządzeniami takimi jak ekran, procesor, bateria/zasilacz, wiatraki itd.
- sterowniki – należy pamiętać o obsłudze twardego dysku (obecnie sATA; NVMe Express dla dysków SSD), systemów plików (główny wkompilowany, inne jako moduły)
 - ▶ grafika: konsola (txt/framebuffer) i tryb graficzny (sterowniki X lub sterowniki zewnętrzne nVidia, AMD Catalyst)
 - ▶ dźwięk: ALSA
 - ▶ obsługa USB, MMC/SD, kontrolerów na płycie głównej
 - ▶ karty sieciowe

- podsystem RTC (real-time clock) – obsługa zegara i komunikacji zegara z systemem, korzystanie z serwerów czasu przez NTP (network time protocol)
- podsystem Microsoft Hyper-V client drivers – pozwala linuxowi być klientem maszyny wirtualnej Hyper-V
- grupa Staging drivers – zawiera moduły niestabilne lub odbiegające od standardów jakości Linuksa; obecnie tam znajdują się sterowniki dla Androida, m.in.
 - ▶ Android Binder IPC driver – sterowniki komunikacji międzyprocesowej
 - ▶ Ashmem (anonymous/android shared memory) – pamięć dzielona w przestrzeni użytkownika bazująca na plikach
 - ▶ Android log driver
 - ▶ Android Low Memory Killer
 - ▶ Android alarm driver – cykliczne wybudzanie jądra
- firmware i microcode – uruchamiają obsługę bardzo specyficznych funkcji niektórych urządzeń

- menu `Kernel hacking` – zaawansowane opcje dostrajające zachowanie jądra, m.in.
 - ▶ `timing information in printk` – dokłada dokładny znacznik czasu do logów jądra
 - ▶ `default message log level`
 - ▶ `Magic SysRq key`
 - ▶ `Detect Hard and Soft Lockups` – wykrywa pętle i zawieszenia systemu i może zainicjować odpowiednie działania (np. `reboot`, `panic`, `oops`)
- menu `Security Options` – tu znajdują się opcje i moduły odpowiedzialne za
 - ▶ kryptografię (sieć, system plików): metody weryfikacji i generowania kluczy, algorytmy szyfrujące, wspomagane przez CPU metody weryfikacji kodu wykonywalnego
 - ▶ LSM (Linux Security Modules): `SELinux`, `SMACK`, `TOMOYO`, `AppArmor`, `Yama`

- menu Virtualization

- ▶ opcja Kernel-based Virtual Machine (KVM) support – uruchamia obsługę przez jądro systemów gościnnych; posiada dwie wersje dla CPU Intel i AMD
- ▶ audyt pamięci MMU (memory management unit) dla KVM
- ▶ przyspieszenie sieci systemu gościnnego (Host kernel accelerator for virtio net)

Android działa na zmodyfikowanym jądrze Linuksa.

- sterowniki związane ze sprzętem (telefony, tablety, karty SIM...) nie są włączone do jądra Linuksa
- repozytorium jądra Linuksa dla androida udostępnia google `android.googlesource.com`
- android do wersji 4.4 to Dalvik/Linux; Dalvik to maszyna wirtualna, która hostowała procesy w androidzie; typowo program w Javie był kompilowany bajtowo a następnie zamieniany na kod wykonywalny Dalvika (.dex Dalvik Executable lub .odex Optimized DEX) przy każdym uruchomieniu aplikacji – kompilator Just-In-Time
- android od wersji 5 oparty jest na nowoczesnym ART (Android Runtime), który również działa na plikach .dex, ale kompiluje je raz, po instalacji – kompilator Ahead-Of-Time
- oprócz optyimizacji, ART kończy zależność androida od Javy, a przez to spór patentowy z Oraclem

Orientacyjna tabela wersji (różni dystrybutorzy mogą oferować różne wersje jądra) za Wikipedią

Android Version	API Level	Linux
-----+-----+		
system w Google-TV		2.6.23
1.0		2.6.25
1.5 Cupcake	3	2.6.27
1.6 Donut	4	2.6.29
2.0/1 Eclair	5-7	2.6.29
2.2.x Froyo	8	2.6.32
2.3.x Gingerbread	9, 10	2.6.35
3.x.x Honeycomb	11-13	2.6.36
4.0.x Ice Cream Sandwich	14, 15	3.0.1
4.1.x Jelly Bean	16	3.0.31
4.2.x Jelly Bean	17	3.4.0
4.3 Jelly Bean	18	3.4.39
4.4 Kit Kat	19, 20	3.10
5.x Lollipop	21, 22	3.10 / 3.16.1 (?)
6.0 Marshmallow	23	3.10 / 3.18.10 (?)

Konfigurowanie jądra Linuksa do pracy z androidem

- domyślna konfiguracja inna niż w czystych jądrach – lepiej nie dotykać domyślnie wybranych opcji
- `Embedded system` – android jest systemem wbudowanym
- `cross-compiling` – jądro musi być kompilowane pod architekturę ARM w odpowiedniej wersji; ustawienie zmiennych środowiskowych

```
export ARCH=arm
export SUBARCH=arm
export CROSS_COMPILE=arm-eabi-
```
- wymagane są dodatkowo JDK, Python, g++ arm w odpowiednich wersjach

Linie rozwojowe jądra androida:

- Goldfish kernel – do uruchamiania androida jako systemu-gościa w maszynie wirtualnej
- MSM kernel – dla chipów Qualcomm MSM
- OMAP kernel – dla chipów Texas Instruments OMAP
- Samsung kernel – dla chipów Hummingbird Samsunga
- Exynos kernel – dla chipów Exynos Samsunga
- Tegra kernel – dla chipów NVidia Tegra

Jądra te różnią się zestawem sterowników, dedykowanych konkretnym platformom sprzętowym.

Bezpieczeństwo systemu

Firewall wbudowany w jądro systemu:

`ipfwadm` (Linux 2.0)

`ipchains` (Linux 2.2)

`iptables` (Linux 2.4 i nowsze)

<http://www.netfilter.org>

Można konfigurować z linii komend w czasie rzeczywistym, ale zazwyczaj konfigurację umieszcza się w skryptach startowych, np. `/etc/rc.d/rc.firewall`. Potrafi rozpoznawać stan pakietów sieciowych. Działa na pojedynczych regułach lub zagnieżdżalnych łańcuchach reguł. Większość funkcji uruchamia się podczas kompilacji jądra i albo są one wkompiłowane w kernel, albo tworzone są pod postacią pakietów. Sterowanie częścią funkcji poprzez pseudopliki z katalogu `/proc/sys/net/...`

Przykład skryptu firewall'a:

```
#!/bin/bash
```

```
#Ładowanie modułu  
modprobe ip_tables
```

```
# Weryfikacja adresu źródłowego na poziomie kernela  
echo 1 > /proc/sys/net/ipv4/conf/default/rp_filter
```

```
# Czyszczenie tablic i łańcucha 'firewall'  
iptables -F  
iptables -X firewall
```

Ustawienie łańcucha 'firewall': logowanie i wywalanie

```
iptables -N firewall
```

```
iptables -A firewall -j LOG --log-level warning \  
--log-prefix "FIREWALL:"
```

```
iptables -A firewall -j DROP
```

Akceptuj siebie

```
iptables -A INPUT -i lo -p all -j ACCEPT
```

```
iptables -A OUTPUT -o lo -p all -j ACCEPT
```

Akceptuj DNS i AUTH

```
iptables -A INPUT -p udp --source-port 53 -j ACCEPT
```

```
iptables -A INPUT -p tcp --source-port 113 -j ACCEPT
```

```
iptables -A INPUT -p tcp --destination-port 113 -j ACCEPT
```

Akceptuj ftp w wersji pasywnej

```
iptables -A INPUT -p tcp ! --syn --source-port 20 \  
--destination-port 1024:65535 -j ACCEPT
```

Akceptuj połączenia SSH z zewnątrz

```
iptables -A INPUT -p tcp --destination-port 22 -j ACCEPT
```

Uruchom zwrotnego PING'a

```
iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
```

Całą resztę wrzucić do łańcucha 'firewall'

```
iptables -A INPUT -p icmp -j firewall
```

```
iptables -A INPUT -p tcp --syn -j firewall
```

```
iptables -A INPUT -p udp -j firewall
```


Kontrola dostępu na poziomie *TCP-wrappers*

Pliki `/etc/hosts.allow` i `/etc/hosts.deny`. Składnia:

`usługa:hosty:polityka`

Na przykład umieszczenie w pliku `hosts.deny` wpisu `ALL:ALL` spowoduje odrzucanie jakichkolwiek prób połączenia się z naszym komputerem.

Wpis w `/etc/hosts.allow` typu:

```
finger : *.domena.net
```

```
finger : ALL : DENY
```

udostępni fingera tylko dla komputerów z domeny `domena.net`

Kontrola dostępu na poziomie demona `inetd` (`xinetd`)

Superdemon `(x)inetd` nie jest konieczny do działania sieci, ale stanowi kolejny etap kontroli dostępu do serwera. Jest to demon odpowiedzialny za udostępnianie (po spełnieniu odpowiednich warunków) usług sieciowych poprzez wywołanie żądanych demonów (serwerów) i unieczynnianie ich gdy nie są już potrzebne.

Pliki konfiguracyjne to zwyczajowo:

`/etc/inetd.conf` (jeden plik z listą usług)

`/etc/xinetd.d/*` (po jednym wpisie dla każdej usługi)

Składnia inetd:

service type protocol wait user server cmdline

Fragment przykładowej konfiguracji:

```
# inetd.conf This file describes the services that will be
#             available through the INETD TCP/IP super server.
#             To re-configure the running INETD process, edit
#             this file, then send the INETD process
#             a SIGHUP signal.
#
imap    stream  tcp  nowait  root    /usr/sbin/tcpd  imapd
telnet  stream  tcp  nowait  root    /usr/sbin/tcpd  in.telnetd
```

xinetd ma znacznie większe możliwości konfiguracyjne; dla każdej usługi tworzona jest osobna grupa opcji (w osobnym pliku); nad nimi domyślne opcje demona.

Fragment przykładowej konfiguracji:

```
# /etc/xinetd.conf
# Sample configuration file for xinetd
#

defaults
{
    instances            = 25
    log_type              = FILE /var/log/servicelog
    log_on_success        = HOST PID
    log_on_failure        = HOST RECORD
#    only_from            = 128.138.193.0 128.138.204.0
#    only_from            = localhost
    disabled              = tftp
}
```

```
# /etc/xinetd.d/imap
service imap
{
    socket_type      = stream
    protocol         = tcp
    wait             = no
    user             = root
    only_from        = 198.72.5.0 localhost
    banner           = /usr/local/etc/deny_banner
    server           = /usr/local/sbin/imapd
}
```

```
# /etc/xinetd.d/telnet
service telnet
{
    flags                = REUSE
    socket_type          = stream
    wait                = no
    user                 = root
    redirect             = 192.168.1.1 23
    bind                 = 127.0.0.1
    log_on_failure += USERID
}
```

Inne zabezpieczenia: `login.access`

Tablica `/etc/login.access` zawiera reguły logowania poszczególnych użytkowników lub przedstawicieli grup, próbujących uzyskać dostęp do systemu poprzez jakiś terminal lub host.

Składnia:

`zgoda : użytkownicy : skąd`

gdzie

- `zgoda` to `+` (zezwolenie) lub `-` (odmowa)
- użytkowników definiuje lista loginów, grup, ALL lub identyfikator typu `user@host`
- `skąd` to numery konsol `tty`, hosty, domeny, ALL, LOCAL lub EXCEPT, np.
`-:ALL EXCEPT wheel shutdown sync:console`

Inne zabezpieczenia: logowanie na konto root

Plik `/etc/securetty` kontroluje bezpośrednie logowanie roota do systemu. Zawiera listę urządzeń uprawnionych. Domyślnie w slackware:

- uprawnione są konsole lokalne: `console` oraz `tty1-tty6`
- zakomentowane są konsole zdalne: `ttyS0-ttyS3` (serial terminal lines), `ttyp1-ttyp7` (pty, pseudo terminals), `pts/0-pts/7` (pseudo terminal slave)

Ograniczone uprawnienia administracyjne dla użytkowników:

- pliki suid, sgid (luka bezpieczeństwa)
- su (udostępnianie hasła)
- sudo (konfigurowalne su -c komenda)

Pakiet sudo

- pozwala w kontrolowany sposób nadać czasowo użytkownikom prawa administratora
- pozwala logować aktywność użytkowników działających jako root
- współpracuje z systemem uwierzytelniania BSD (/etc/login.conf)
- konfiguracja w /etc/sudoers

Plik `/etc/sudoers`

- jest czytany zawsze w całości, ustawienia są respektowane w kolejności wystąpienia, ostatnie jest wykonywane
- wspiera aliasy

```
User_Alias      FULLTIMERS = millert, mikef, dowdy
Runas_Alias     OP = root, operator
Host_Alias      SPARC = bigtime, eclipse, moet, anchor :\
                SGI = grolsch, dandelion, black
Cmnd_Alias      PRINTING = /usr/sbin/lpc, /usr/bin/lprm
```

- posiada pewne ustawienia domyślne, które można nadpisać

Przykład:

```
# root i grupa wheel mogą wszystko wszędzie
# % oznacza grupę (ustawienie domyślne, + to grupa sieciowa)
root          ALL = (ALL) ALL
%wheel        ALL = (ALL) ALL

# joe może "su operator" na każdej maszynie (musi znać hasło)
joe           ALL = /usr/bin/su operator

# wszyscy mogą montować cdromy na maszynach z grupy CDRom
# bez podawania hasła
ALL           CDRom = NOPASSWD: /sbin/umount /CDROM,\
               /sbin/mount -o nosuid\,nodev /dev/cd0a /CDROM
```

Komenda `chroot` pozwala na zmianę katalogu głównego na inny. Wyjście przez `exit`. Po co to?

- używanie drugiego systemu z konsoli (przy pewnej wprawie da się uruchomić serwer X)
- naprawianie systemu, np. korzystając z LiveCD (lilo, grub, system zhakowany)
- zamknięcie użytkownika (np. gość ftp) w ograniczonej, mniejszej kopii systemu

Pakiety PAM (Pluggable Authentication Modules)

Klasycznie uwierzytelnianie w linuxie polega na podaniu hasła i porównaniu go z zaszyfrowaną wersją w pliku na dysku. PAM udostępnia moduły uwierzytelniające na różne sposoby (inne typy szyfrowania, specjalistyczny hardware...) co powoduje, że programy takie jak `login` nie wymagają przepisywania przy wprowadzaniu nowych metod do użytku. Wystarczy że będą zdolne do wywołania odpowiedniego modułu. Z PAM mogą korzystać m.in.

- OpenPGP
- Bluetooth
- POP3
- SQL
- iptables
- SAMBA
- chroot
- Kerberos
- ssh
- imap
- LDAP
- PHP, Python, Perl
- sudo
- ProFTPD

Łaty na jądro Linuxa: projekt Zen Sources

(github.com/zen-kernel) oraz udostępniający binaria liquorix.net

- Parę lat temu w tym zestawie łat można było znaleźć m.in.:
*aircrack aufs bfq btrfs cko custom-cflags daconfig eeepc
expose-vm split ext4 fbcondecor fujitsu-laptop-extras gspca hz-kconfig
l7filter linux-uv localversion mactel madwifi march-native master
mm-patches phc-acpi-cpufreq reiser4 squashfs strip-debug
swap-prefetch toshiba_acpi tp_smapi tuxonice wireless-testing
zen-logo*
- Obecnie wiele z tych poprawek to osobno rozwijane projekty (np. tuxonice, reiserfs4) lub zostało zarzuconych (np. madwifi na korzyść ath5k i ath9k).
- Nowe wydania Zen wzbogacają jądro o wiele funkcji eksperymentalnych, przede wszystkim mechanizm Zen Interactive Tuning, który w locie pozwala na zmianę responsywności jądra kosztem obciążenia i poboru mocy. Jest wydawany nieco później niż kernel Torvaldsa.

Łatki bezpieczeństwa w linuxie: **SELinux**.

www.nsa.gov/selinux

Security Enhanced Linux

National Information Assurance Research Laboratory of the
National Security Agency

Implementacja rozwijanego przez NSA mechanizmu bezpieczeństwa Flask
(Flux Advanced Security Kernel) w systemie Linux. Licencja GPL.

Model bezpieczeństwa polega na dwuetapowym działaniu:

- Koncepcja **DAC** (Discretionary Access Control – uznaniowa kontrola dostępu) stosuje politykę bezpieczeństwa biorąc pod uwagę tożsamość obiektów i ich uprawnienia; użytkownik przekazuje swoje uprawnienia uruchamianym procesom, co jest szczególnie niebezpieczne w przypadku administratora lub plików z bitem `suid`.
- Koncepcja **MAC** (Mandatory Access Control – obowiązkowa kontrola dostępu) wprowadza reguły **po** regułach DAC i żaden z użytkowników nie może ich obejść.

Inne łaty bezpieczeństwa:

- Openwall, wraz z Openwall GNU/*/Linux
- grsecurity (od IX.2015 wersje stabilne są płatne)
- AppArmor – ochrona modułowa LSM (Linux Security Modules)

Projekt MinGW, czyli *Minimalist GNU for Windows*

- dostarcza środowisko programistyczne z narzędziami GNU, działające w windows
- zawiera m.in. port gcc, zestaw narzędzi GNU binutils, wzorowaną na bash powłokę msys, mogącą zastąpić cmd.exe, wybór innych narzędzi

Nie dostarcza pełnej implementacji POSIX dla windows, więc nie pozwala na uruchamianie wszystkich aplikacji. Przeznaczony do tworzenia aplikacji pod windows z wykorzystaniem narzędzi GNU, więc ułatwia portowanie aplikacji linuksowych.

Projekt cygwin [www.cygwin.com] to

- zbiór narzędzi GNU i open source możliwych do uruchomienia w środowisku windows (m.in. gcc, bash, ssh, X...)
- biblioteka `cygwin1.dll` dostarczająca API zgodnego z POSIX

Cygwin jest środowiskiem wykonawczym windows pozwalającym przede wszystkim kompilować źródła programów linuxowych.

Tworzy warstwę pośrednią między aplikacją a systemem, tłumaczącą wywołania systemowe Linuksa na ich odpowiedniki w windows. Wymaga wkompilowania swojego kodu w aplikację.

Prowadzone jest repozytorium

<https://sourceware.org/cygwinports/>

<https://sourceforge.net/projects/cygwin-ports/>

zawierające porty wielu programów i bibliotek, w tym pełnych środowisk graficznych gnome, kde, lxde, xfce, mate, rox...

Zobacz też: Linux Magazine 08 (126), sierpień 2014, wydanie PL

Projekt wine

- darmowa, otwarta implementacja API windows
- pozwala uruchomić stabilnie wiele wymagających aplikacji, w tym Microsoft Office, programy CAD, Adobe Photoshop i wiele gier (np. Wiedźmin)



Wine jest emulatorem, w którym uruchamiane są pod linuxem aplikacje pisane dla windows. Dostarcza emulację API windows, więc nie zapewnia 100% kompatybilności. Najlepszym testem są aplikacje korzystające z DirectX.

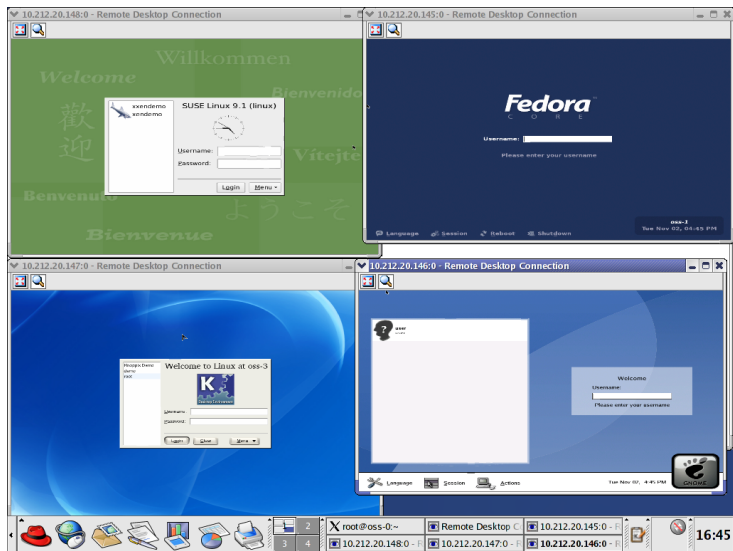
Z wine wydzielił się osobny projekt WineX, wyspecjalizowany do obsługi gier wymagających DirectX. W 2004 projekt zmienił nazwę na Cedega, a obecnie jest to TransGaming/GameTree. Te projekty są komercyjne. Filozofia rozwoju tego projektu była taka, że funkcjonalność API windows „zgadywano” (ze względu na zamknięty kod windows), próbując uruchamiać bezbłędnie kolejne gry. Metodą kolejnych przybliżeń dopracowywane jest działanie cedegi pod konkretne tytuły gier.



Maszyny wirtualne:

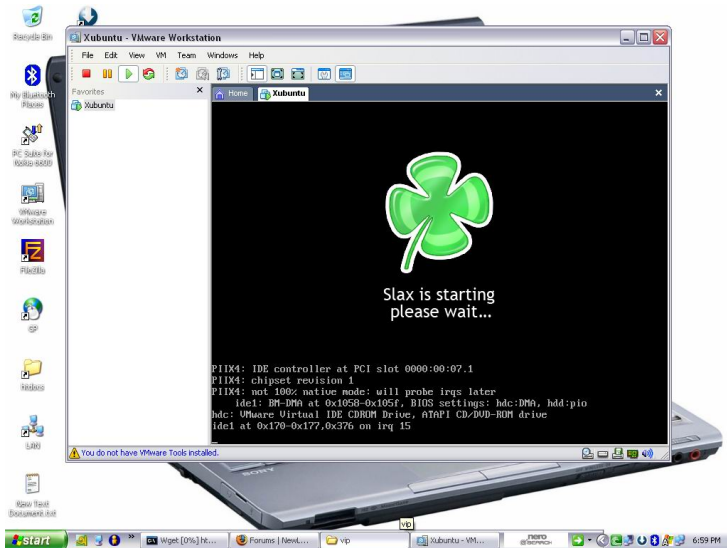
- projekt open source ponad 20 udziałowców, w tym AMD, Cisco, Dell, HP, IBM, Intel, Mellanox, Network Appliance, Novell, Red Hat, SGI, Sun, Unisys, Veritas, Voltaire i Citrix, wydany na licencji GPL
- niewielki: około 50000 linii kodu
- dostarcza „nadzorcę” (XEN hypervisor), który oddziela fizyczny sprzęt od systemu operacyjnego; pozwala to na korzystanie ze sprzętu przez wiele serwerów jednocześnie
- technologia *parawirtualizacji* – tworzenie warstwy abstrakcji reprezentującej sprzęt, wykorzystanie metod wirtualizacji zaszytych w procesorach AMD i Intel

- wykorzystanie działających w systemie sterowników
- dostępny jako moduły, od Linux 2.6.23 włączony do kodu jądra (przynajmniej w części)
- składa się z trzech części: hipernadzorcy (czyli właściwego XENa), i dwóch domen – Dom0 udostępnia wirtualny sprzęt dla DomU
- można na nim uruchomić systemy Linux, Windows, Solaris i BSD w różnych odmianach



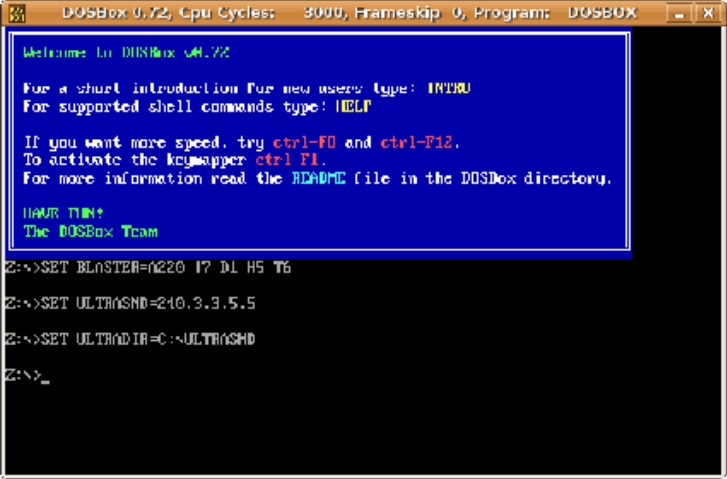
VMWare Workstation

- projekt komercyjny, wersje demo do pobrania (30 dni)
- tworzy wirtualny komputer wewnątrz działającego OS
 - ▶ CPU: Pentium Pro
 - ▶ RAM: wg przydziału (max. 8 GB)
 - ▶ Grafika: S3 z możliwością uruchomienia wsparcia 3D i wielu monitorów
 - ▶ HDD: wg przydziału
 - ▶ sieć pomostowa, NAT, wbudowany serwer DHCP, max. 10 wirtualnych switchy
 - ▶ CD, DVD, audio: wg konfiguracji
 - ▶ wsparcie dla USB 2.0, wirtualnego SMP



Inne emulatory

- DOSBox – emulator DOSa napisany z myślą o uruchamianiu starych gier; obsługa dysku, CD, wielu kart dźwiękowych (ADLib, Gravis Ultrasound, SB16/Pro...), wielu kart graficznych (Hercules, CGA, VGA, S3 Trio 64); działa w trybie tekstowym (ale są frontendy)



The screenshot shows a DOSBox window titled "DOSBox 0.72, Cpu Cycles: 3000, Frameskip 0, Program: DOSBOX". The window has a blue title bar and standard Windows window controls. The main area is a black terminal window with green text. A white rectangular box highlights the initial welcome message. Below the box, several configuration commands are entered at the prompt "Z:\>".

```

Welcome to DOSBox 0.72

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

If you want more speed, try ctrl-F9 and ctrl-F12.
To activate the keymapper ctrl F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team

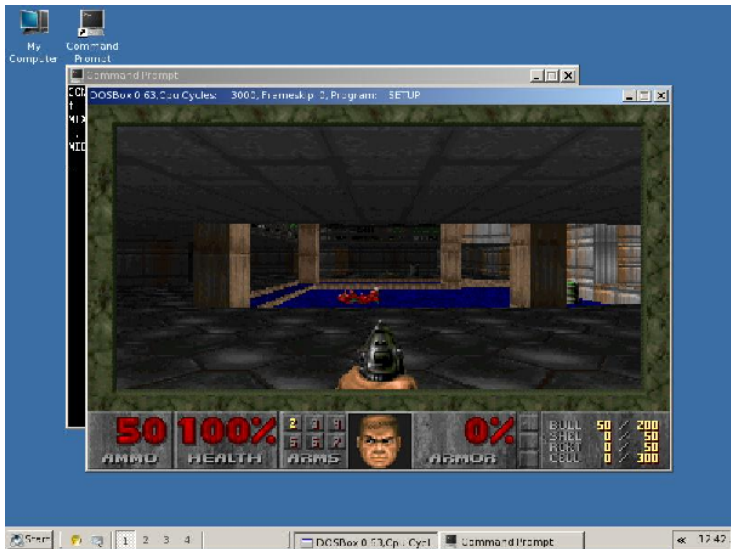
Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>SET ULTRASND=240,3,3,5,5

Z:\>SET ULTRADIR=C:\ULTRASND

Z:\>_

```



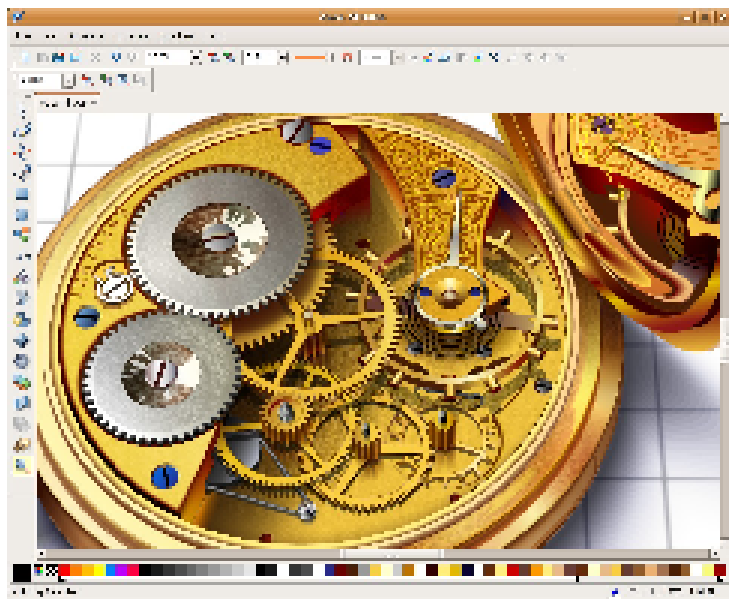
- Win4Lin – komercyjna (30–60\$) aplikacja pozwalająca uruchamiać binaria windowsowe jako natywne w linuxie
- MAME – Multiple Arcade Machine Emulator, pozwala uruchomić gry pisane na automaty (zgrane kości ROM)
- Fake NES – Nintendo Entertainment System
- Virtual Game Boy

Dokumenty

- pakiety biurowe: OpenOffice.org, KOffice, Gnome Office...
- system składu tekstu T_EX
- konwertery (ps2pdf...) i przeglądarki (gv...)

Grafika

- wykresy: gnuplot, grace...
- diagramy, rysunek techniczny: xfig...
- grafika rastrowa: Gimp, ImageMagick, Corel PhotoPaint 9...
- grafika wektorowa: sodipodi, xfig, Xara Extreme...
- grafika 3D: Povray, Blender, Maya...



Wideo

- odtwarzacze: mplayer, xine...
- mnóstwo kodeków (divx, xvid, libavc...) i narzędzi konwertujących (mencoder...)

Muzyka

- tworzenie (sequencer): rosegarden, ecasound...
- tworzenie (tracker): soundtracker, cheesetracker...
- tworzenie (efekty): audacity, glame...
- odtwarzanie: xmms, mplayer, realplayer...
- CD (zgrywanie): grip, cdda2wav, cdparanoia, k3b...

```
# Skrypt konwertujący pliki audio cd na mp3
# startujemy w pustym katalogu

# ripuje ścieżki audio do track01.wav...
# (można też użyć cdda2wav)
cdparanoia --batch

# konwersja na track01.mp3...
for plik in `ls` do
lame -h $plik ${plik%.*}.mp3
done
```

Pakiet ImageMagick – obróbka plików graficznych na poziomie linii komend

```
convert plik.bmp plik.jpg
```

Rozpoznaje prawie wszystkie formaty graficzne i potrafi konwertować jedno na drugie. Opcje (ponad 100!) pozwalają na modyfikację plików, w tym: dodanie tekstu, dodanie efektów (transformacje), zmiana próbkowania, rozdzielczości, deformacje palety kolorów, generowanie miniatur, zmiana tła, przezroczystość, filtry...

W skład pakietu wchodzi: convert, identfy, composite, montage, compare, display, animate, import, conjure (wywołuje skrypty Magick Scripting Language).

Mencoder – koder wideo z pakietu mplayer.

```
# Koduje rozdz. 10-14 z 2. tytułu na płycie dvd ze skalowaniem
mencoder dvd://2 -chapter 10-14 -vf scale=640:480 -o title2.avi
    -oac copy -ovc lavc -lavcopts vcodec=mpeg4
```

```
# Składa dwa pliki w jeden film
cat part1.avi part2.avi |
    mencoder -oac mp3lame -ovc xvid -o full.avi -
```

```
# Podobnie z archiwum rar
rar p test-SVCD.rar |
    mencoder -ovc lavc -lavcopts vcodec=mpeg4:vbitrate=800
    -ofps 24 -
```

```
# Składa pliki jpg w animację
mencoder "mf://*.jpg" -mf fps=25 -o output.avi
      -ovc lavc -lavcopts vcodec=mpeg4

# Zgrywa program z tunera TV
mencoder -tv driver=v4l:width=640:height=480
      tv:// -o tv.avi -ovc raw
```

Programowanie

- kompilatory: gcc...
- edytory, buildery: emacs, vim, CodeForge, Anjuta...

Sieć

- poczta: Thunderbird, Sylpheed, pine, mutt, emacs...
- www: Firefox, Konqueror, emacs, lynx, links...
- serwery: ProFTPD, OpenSSH, Apache, sendmail...
- filtry: SpamAssassin, Procmail

Bazy danych

- PostgreSQL
- MySQL
- DB1...DB4

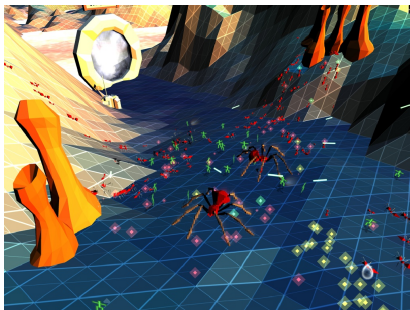
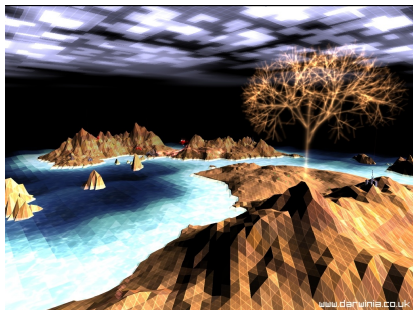
<http://www.linuxrsp.ru/win-lin-soft/table-eng.html#42>

Gry

- gry natywne (różne biblioteki – często SDL: image, mixer, net, ttf lub OpenGL: Mesa, libGLcore)
- gry przeniesione z windowsów (Loki), np.
 - ▶ Rune
 - ▶ Heavy Gear
 - ▶ Unreal Tournament i inne
- przepisane instalatory (wymagają często instalacji na partycji windowsowej lub oryginalnych płyt instalacyjnych), np.
 - ▶ Return to Castle Wolfenstein
 - ▶ Neverwinter Nights i inne
- wykorzystane uwolnione silniki graficzne, m.in. doom, heretic, quake III, np. open arena, gldoom i inne

Darwinia

Gra w świecie komputerowym, wektorowa grafika, sterowanie za pomocą gestów myszki.



Wormux

Klon Wormsów, które są klonem Scorched Earth.



Free Civilization

"Because civilizations should be free!"



Frozen Bubble

Oryginalna gra od ekipy
Mandrake Linux



Supertux

Klon Super Mario Bros.



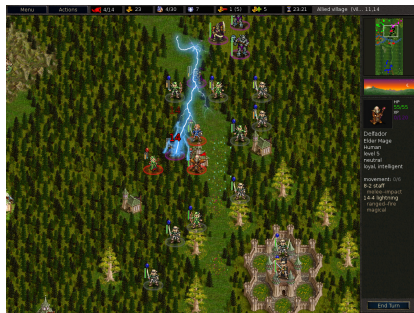
Tux Racer

Zjadanie śledzi na czas
w ślizgu na brzuchu.



Battle for Wesnoth

Strategia turowa podobna do
Warcrafta.



Kilka mniej lub bardziej przydatnych sztuczek i ciekawostek

Konwersja kodowania

```
iconv -f utf8 -t iso-8859-2 plik.utf8 > plik.iso2
```

Swap

komendy `swapon` i `swapoff` pozwalają na włączenie i wyłączenie korzystania z partycji typu Linux swap

Odtworzenie MBR

- start z live-cd
- grub
- `find /boot/grub/stage1` wynikiem będzie np. `(hd0,x)`
- `root (hd0,x)`
- `setup`

Pendrive się nie chce odmontować, „urządzenie zajęte”

```
fuser -v -m /media/pendrive
```

```
fuser -v -n tcp 22
```

potem kill odpowiedni proces

bash w trybie emacs'a

CTRL-r szuka rekursywnie wstecz w historii poleceń

konsola

SHIFT-PgUP, SHIFT-PgDn przewija konsolę

sprawdzanie i naprawa dysków S.M.A.R.T.

```
smartctl -test=long /dev/sda
```

```
mke2fs -c /dev/sda szuka uszkodzonych bloków
```

Tuning X

program xset pozwala na ustawienie wygaszania ekranu, czułości myszy, obsługi klawiatury etc. na działającym serwerze, np.

```
xset m 3 10
```

```
xset dpms 300 600 900 (standby, suspend, off [sek])
```

Zasoby sieci Windows w linuxsie

Demon SAMBA

```
smbmount "\\\\.\\WINDOWS\\C" -c 'mount /mnt/wind'
```

```
-l 192.168.0.3
```

```
mount -t smbfs //server-ip-or-name/share /mnt/location
```

```
-o username=user,password=pass,domain=DOMAIN
```

Niechący usunięto rpm

rpm2cpio lub rpm2tgz przekształca pakiet na wersję użyteczną;
ostatecznie zostaje kompilacja ze źródeł

Wygaszanie konsoli tekstowej

```
setterm -blank 0 (wyłącza)
```

SysRq

Jeśli wkompiłowane w jądro (`CONFIG_MAGIC_SYSRQ`) i uruchomione
`echo "1" > /proc/sys/kernel/sysrq`

- Alt-SysRq-r przestawia klawiaturę przemapowaną przez X
- Alt-SysRq-e zamyka wszystkie procesy (`SIGTERM`) oprócz `init`
- Alt-SysRq-i zabija wszystkie procesy (`SIGKILL`) oprócz `init`
- Alt-SysRq-s synchronizuje systemy plików
- Alt-SysRq-u montuje wszystkie systemy plików `read-only`
- Alt-SysRq-b restartuje system bez odmontowania partycji

`reisub` = bezpieczny restart

Reboot Even If System Utterly Broken

Data i czas

`date -s 19990520` ustawia na 20 maja 1999

`date -s 15:56` ustawia godzinę 15:56

`date -d 19990520` sprawdzi poprawność wpisania formatu daty

`[bash]# date -d '13 JAN 2009 14:34'`

wto, 13 sty 2009, 14:34:00 CET

Kopia MBR

`dd if=/dev/sda of=/root/mbr.bck bs=512 count=1`

Grafika przez ssh

`ssh -X bolekk@komputer.net 'nedit'`

Tunelowanie połączeń przez ssh: SMTP

Wysyłanie poczty (SMTP, port 25) przez szyfrowany tunel ssh (port np. 8025)

```
ssh mastah@poczta.pl -L 8025:localhost:25
```

UWAGA: -L = nasłuchiwanie, localhost liczony względem adresu docelowego (poczta.pl)

Tunelowanie połączeń przez ssh: SMTP + POP3

Wysyłanie poczty (SMTP, port 25) przez szyfrowany tunel ssh (port np. 8025), odbieranie przez POP3 (port 110, przekierowany na szyfrowany 8110)

```
ssh mastah@poczta.pl -L 8025:localhost:25  
-L 8110:localhost:110
```

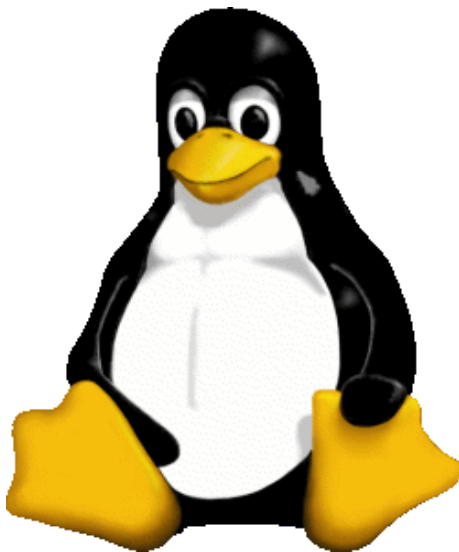
UWAGA: konfiguracja klienta poczty – wysyłanie localhost:8025, odbieranie localhost:8110; sesja ssh musi być cały czas otwarta!

Opcje bootowania jądra, zmieniają konfigurację jądra przy starcie systemu.
Pełna lista np. pod adresem

www.mjmwired.net/kernel/Documentation/kernel-parameters.txt

Kilka bardziej przydatnych:

<code>root=/dev/sda2</code>	partycja z głównym systemem plików
<code>rootdelay=4</code>	czeka 4 sek zanim zamontuje gł. partycję
<code>rootwait</code>	czeka aż urządzenie root się pojawi (usb)
<code>ro, rw</code>	read-only, read-write
<code>vga=ask</code>	rozdzielczość bufora ramki (normal, 773...)
<code>acpi=off</code>	gdy laptop wariuje
<code>init=/bin/sh</code>	co uruchomić zamiast <code>/sbin/init</code>
<code>initrd=, noinitrd</code>	ramdisk
<code>logo, nologo</code>	pingwin on/off
<code>resume=, noresume</code>	hibernacja
<code>nousb</code>	wyłącza system USB
<code>quiet</code>	wyłącza większość logów
<code>S</code>	wprowadza init w stan „single user”
<code>vt.default_utf8=0</code>	wyłącza utf8 w konsolach



Przeciw linuxowi przemawia:

- sprzęt dedykowany dla windowsów – smutne ale prawdziwe (np. winmodemy, laptopy...)
- niepełne sterowniki lub ich brak, szczególnie w przypadku najnowszego sprzętu
- wciąż trudniejsza obsługa (więcej możliwości kusi żeby ich spróbować, a to wymaga nauki!)
- powszechne użycie innych standardów, przeforsowane przez Sami-Wiecie-Kogo – brak office'a, outlooka etc. (nieuczciwe blokowanie w aplikacjach MS współpracy z innym oprogramowaniem)



Za linuxem przemawia:

- cena, stosunek jakość/cena, kompletność dystrybucji i dodatkowego oprogramowania
- można zrobić z nim (prawie) wszystko – potężne możliwości
- nieporównywalna stabilność i bezpieczeństwo – platformy serwerowe, klastry
- nieporównywalna wydajność, szczególnie przy wielu użytkownikach i mocnym obciążeniu
- doskonała okazja żeby się czegoś nauczyć :)

