Podstawy pracy w konsoli

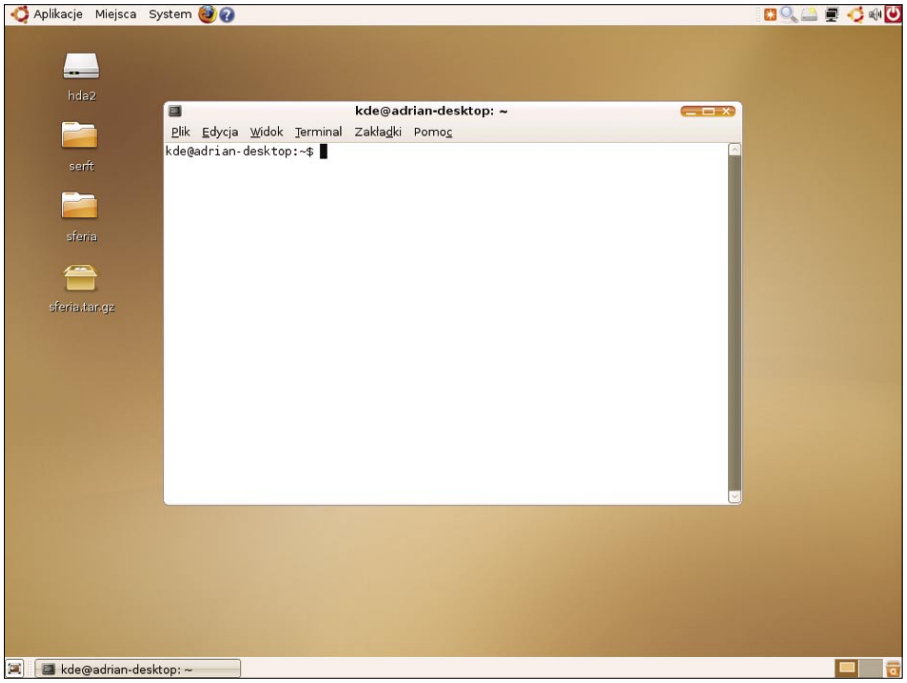
Robert Kałat

Prawdziwą potęgę każdej dystrybucji Linuksa zauważa się dopiero używając konsoli. I choć obecnie prawie wszystko można „wyklikać" w programach graficznych, to zdarzają się sytuacje, gdy sięgnięcie do konsoli może być nie tylko potrzebne, ale wręcz konieczne. Wystarczy wyobrazić sobie sytuację, że pięknie skonfigurowany system z działającą akceleracją 3D, na której to Beryl ukazuje swoją prawdziwą moc odmawia uruchomienia środowiska graficznego. Niektórzy w tym momencie sięgają po płytkę *live-cd* i przeprowadzają ponowną instalację!? Ale po co!? Używając konsoli w ciągu kilku sekund można poprawić 1 wpis tak, by wystartowało środowisko graficzne.

Niestety od czasu do czasu takie scenariusze się zdarzają. Jednak wystarczy posiadać podstawową wiedzę o poleceniach i pracy w konsoli, by wybrnąć z takiej sytuacji bez szwanku. Oczywiście konsola nie służy pomocą wyłącznie w sytuacjach awaryjnych, a wielu użytkowników uważa, że żaden graficzny program nigdy nie dorówna swoim tekstowym odpowiednikom w kwestiach szybkości, czy stopnia kontroli nad systemem. Nawet jeśli jest to bardzo przesadzona opinia warto choć przez chwilę potrzymać byka za rogi!

Jak przełączyć się na konsolę

Typowo do konsoli przełączymy się poprzez kombinację klawiszy [*ALT*]+[*CTRL*]+[*F1...F6*]. Równie dobrze będąc zalogowanym do środowiska graficznego możemy uruchomić *Aplikacje->Akcesoria->Terminal* co przedstawiono na Rysunku 1.



Rysunek 1. Pulpit wraz z uruchomionym terminalem

Konsola czy też terminal, to nic innego jak powłoka systemowa, czyli interpreter poleceń. W powłoce tej wpisujemy polecenia które ma wykonać komputer. Najpopularniejszą i najczęściej stosowaną powłoką w dystrybucjach Linuksa jest Bash (1)(2). Mocną stroną powłoki jest między innymi prostota w pisaniu skryptów, czy też szczególnie przydatne dopełnianie poleceń za pomocą klawisza *[TAB]*.

Jak ułatwić sobie życie pracując w terminalu

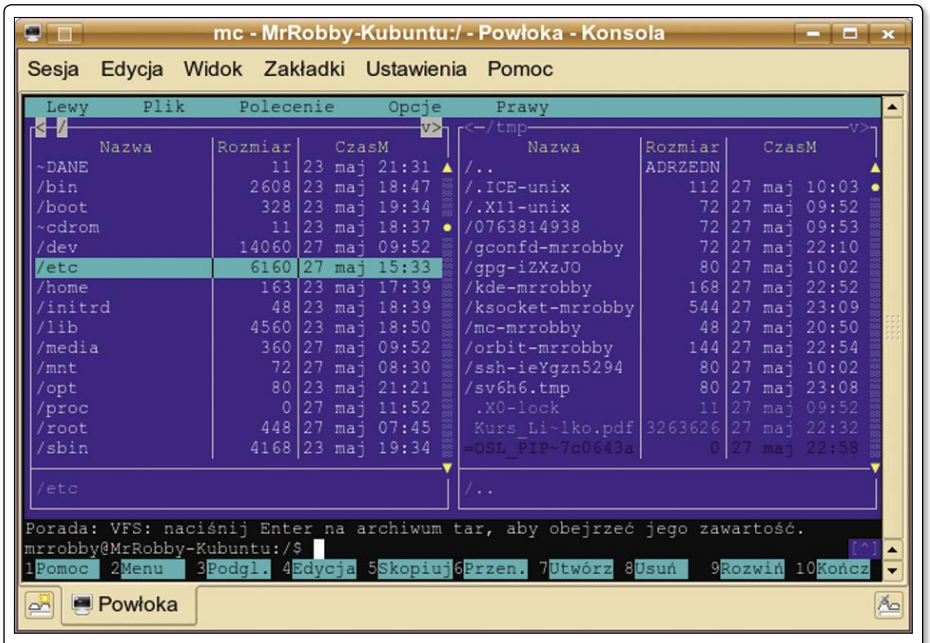
Wiele typowych operacji, pozwoli wykonać nam program mc (*Midnight Commander*) bez znajomości poleceń które musielibyśmy wpisać w konsoli. Swoją funkcjonalnością, rozpoczynającą się od typowych operacji na plikach i katalogach, poprzez zaawansowaną obsługę archiwów i kończąc na funkcjach klienta ftp (jak również smb czy ssh) nie odbiega od komercyjnych menadżerów plików. Okno programu Midnight Commander przedstawiono na Rysunku 2.

Niestety program nie jest domyślnie instalowany z naszym systemem. Rozwiązujemy to w szybki sposób za pomocą `sudo apt-get install mc`. Po zakończonej instalacji nie zostaje nam nic innego jak tylko uruchomić program wpisując `mc [ENTER]`. Jak widać mc jest bardzo prosty i intuicyjny, więc opis z mojej strony będzie tylko zbędnym komentarzem.

Podstawowe polecenia

Przed opisaniem typowych poleceń, chciałbym zwrócić uwagę na klawisz *[TAB]*,

O Autorze
Autor jest studentem informatyki na uczelni WSTI w Katowicach. W społeczności Ubuntu znany jako *MrRobby*.
Kontakt z autorem:
mrrobby@ubuntu.pl
lub jid:
mrrobby2@gmail.com



Rysunek 2. Midnight Commander

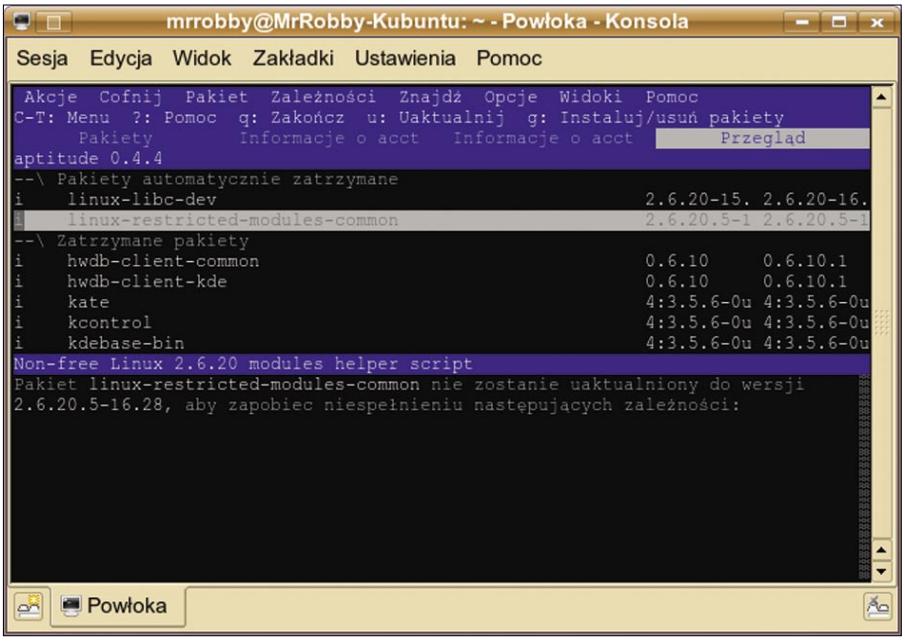
o którym to wspomniałem wcześniej. Dzięki niemu możemy dopełniać polecenia, ścieżki czy też dostawać podpowiedź ze strony powłoki systemowej. Przykładowo praca z poleceniem `cd` (ang. *Change Directory*), które to służy do poruszania się po strukturze katalogów, byłaby bardzo utrudniona, gdyby powłoka nie umożliwiała dopełniania. W takiej sytuacji musielibyśmy znać na pamięć całą strukturę plików (co jest niemożliwe) lub po każdym wejściu do katalogu należałoby listować jego zawartość a następnie przemieszczać się poziom głębiej i tak w kółko. Przykładowo wpisując `cd /e`, po wciśnięciu *[TAB]*, powłoka dokończy polecenie `cd /etc/`. Polecam poćwiczyć samemu, na pewno będzie to lepsze rozwiązanie, niż sucha teoria. Najczęściej używane polecenia systemowe wraz z ich krótkim opisem znajdziemy w ramce „Lista poleceń”.

Jeśli mamy jakikolwiek problem z opcjami danego programu, zawsze możemy użyć przełącznika `help np tar --help` by poznać całą składnię polecenia. Silną stroną systemu jest jego dokumentacja, czyli tzw. *man*, z którego korzystamy w następujący sposób `man tar`. Po wydaniu komendy uruchomi się nam podręcznik programu jeśli tylko takowy istnieje, w naszym przypadku będzie to dokumentacja polecenia `tar`. Bardzo ważną umiejętnością przy pracy w terminalu jest obsługa konsolowego edytora tekstu. Standardowo w systemie jest zainstalowany nano, z którego możemy korzystać w następujący sposób `nano /tmp/Install`. Komenda, którą podałem otwiera do edycji plik *Install* znajdujący się w folderze `/tmp`. Edytor jest całkowicie po polsku. Jeśli zainstalujemy w systemie nakładkę mc, o której wcześniej pisałem, będziemy posiadać dodatkowo edytor *mcedit*. Osobiście używam programu *vim*, który nie jest instalowany domyślnie. W sieci jest bardzo dużo konsolowych edytorów tekstu, a niektóre z nich potrafią wręcz przytłoczyć mnogością swoich funkcji. Jednak niezwykle ważne jest, aby choć jeden opanować w stopniu wystarczającym do nieskrępowanej edycji dowolnego pliku tekstowego.

Instalacja pakietów

Niewątpliwie bardzo mocną stroną Debiana, a co za tym idzie także Ubuntu jest apt (ang. *Advanced Packaging Tool*) który jest systemem zarządzającym pakietami. Apt zarządza pakietami z oprogramowaniem w formacie deb, dzięki czemu, w zdecydowany sposób upraszcza proces instalacji programów. Przykładowo chcąc zainstalować przeglądarkę firefox, wydajemy polecenie: `sudo apt-get install firefox`. Program przeszuka repozytoria w pod kątem żądanego pakietu, sprawdzi zależności, czyli listę pakietów które należy dodatkowo zainstalować, aby żądany program działał poprawnie. Gdy taka lista będzie gotowa zostaniemy poinformowani o tym jakie pakiety muszą zostać zainstalowane oraz ewentualnie usunięte, aby interesujący nas program mógł zostać zainstalowany. Jeśli „zgodzimy się” na to, apt ściągnie za nas wymagane pakiety, zainstaluje i na dodatek wstępnie je skonfiguruje. Warto przy tym zauważyć, że poza listą wymaganych pakietów otrzymujemy dodatkowe podpowiedzi, a dokładniej listę pakietów, które mogą okazać się przydatne, jednak program może działać bez ich instalacji.

Dzięki stosowaniu pakietów *deb* możemy w łatwy sposób zarządzać programami w systemie. Mam tu na myśli instalacje, usuwanie czy też aktualizację do nowych wersji. Na stronach w internecie często spotkacie informacje typu: by zainstalować ten program, zainstaluj to i to.



Rysunek 3. Zarządzanie pakietami za pomocą Aptitude

Lista poleceń

- `grep` – służy do przeszukiwania plików.
- `grep deb-src /etc/apt/sources.list` – wyświetli linijki zawierające wyrażenie deb-src.
- `grep -v deb-src /etc/apt/sources.list` – tym razem wyświetli wszystko prócz wierszy zawierających deb-src.
- Ogólnie program posiada bardzo duże możliwości jeśli chodzi o wyszukiwanie wyrażeń regularnych. Jest to na tyle skomplikowany temat że nadaje się na osobny artykuł. Grep jest bardzo często używany do prostego wyszukiwania przy korzystaniu z innych komend co przedstawię poniżej.
- `cat` – wyświetla zawartość pliku na ekranie bądź zapisuje do innego pliku.
- `cat /etc/apt/sources.list` – wyświetli nam zawartość pliku sources.list.
- `cat /etc/apt/sources.list > /tmp/test.txt` – cała zawartość pliku sources.list zostanie zapisana w *test.txt.*
- `cat /etc/apt/sources.list | grep deb-src` – wylistuje nam zawartość pliku, ale pokaże tylko te wiersze które zawierają wyrażenie deb-src. Często wykorzystuje się listowanie z wykluczeniem jakiegoś wyrazu. W przypadku podawania np. pliku konfiguracyjnego na forum, zbędnym jest pokazywanie komentarzy które zaczynają się od #. Sam znak # jest symbolem specjalnym, dlatego musimy go ująć w '.
- `cat /etc/apt/sources.list | grep -v '#'` – wyświetli zawartość pliku z pominięciem wierszy zawierających #.
- `cd` – poruszanie się po strukturze katalogu.
- `cd /tmp` – wchodzimy do katalogu /tmp.
- `cd /` – przeskakujemy do głównego katalogu.
- `cd ~/` – przechodzimy do katalogu domowego.
- Jeśli chcemy się odwołać do jakiegoś katalogu który jest w naszym domowym, to zamiast pisać `cd /home/uzytkownik/wazny.katalog` możemy użyć polecenia `cd ~/wazny.katalog`.
- `chmod` - zmieniamy uprawnienia do pliku lub katalogu. Po wydaniu polecenia `ls -l` wyświetli się nam informacja o plikach i katalogach. W pierwszej kolumnie znajdują się uprawnienia do pliku/katalogu. Uprawnienia składają się z 3 grup: *rwx rwx rwx*. Pierwsza grupa to właściciel oznaczany *u*, kolejna kolumna oznacza grupę i jest oznaczana *g*. Ostatnia kolumna odpowiada za resztę użytkowników i jest oznaczana *o*. Dodatkowo jest wprowadzony kolejny parametr *a* oznaczający wszystkich. Atrybutami jakie możemy ustawić są: *r* – odczyt, *w* – zapis, *x* – wykonywanie.
- `chmod a-w plik.txt` – zabiera wszystkim użytkownikom uprawnienia do zapis
- `chmod o+rx plik.txt` – dodanie praw dla reszty użytkowników czytanie oraz wykonywanie. Jeśli plik posiada już uprawnienie do zapisu, to uprawnienia się zsumują i będą wyglądać *rwx*. Gdy chcemy by plik miał dokładnie takie uprawnienia jak podajemy wykorzystujemy znak = zamiast +
- `chmod o=rx plik.txt` – ustalenie praw dla reszty użytkowników na czytanie oraz wykonywanie.
- `chmod -R o=rx /tmp/katalog` – ustalenie praw dla reszty użytkowników na czytanie oraz wykonywanie dla katalogu `/tmp/katalog`. Dzięki parametrowi `-R` te same uprawnienia zostaną ustawione na plikach i katalogach znajdujących się wewnątrz `/tmp/katalog`.
- `chown` – zmieniamy właściciela pliku lub katalogu. Listując katalog za pomocą polecenia `ls -l` w 2 i 3 kolumnie mamy odpowiednio pokazanego właściciela pliku lub katalogu oraz grupę. Jeśli jesteś właścicielem pliku to możesz dowolnie ustawiać na nim uprawnienia tzn. kto może go czytać, edytować, uruchamiać. Tego wszystkiego dokonujemy wcześniej opisanym poleceniem `chmod`.
- `chmod janek:users /tmp/pliczek` – polecenie zmienia uprawnienia do pliku *pliczek*. Od tej pory właścicielem jest *janek* oraz grupa *users*.
- `chmod -R janek:users /media/hda7` – nadajemy te same uprawnienia na katalog /`media/hda7` z tym że dzięki parametrowi `-R`, nasz ustawienia odziedziczą również pliki i katalogi znajdujące się wewnątrz `/media/hda7`.
- `cp` - kopiowanie plików i katalogów.
- `cp /tmp/test.txt /tmp/test.txt.kopia` – kopiujemy plik `test.txt` do `test.txt.kopia`. Po prostu tworzymy kopię pliku. Bardzo zalecane przed edytowaniem plików systemowych.
- `cp -r /tmp/1 /tmp/2/` – kopiujemy katalog 1 wraz z całą jego zawartością do katalogu `/tmp/2`.
- `exit` – wylogowanie z konsoli
- `gzip` – kompresuje pliki do archiwum `*.gz`.
- `gzip pliczek` – plik o nazwie pliczek zostaje skompresowany do `pliczek.gz`.
- `gzip -9 pliczek` – program wykona identyczną operację jak wyżej z tym że skompresuje plik najmocniej jak tylko można.
- `gzip -d pliczek.gz` – pliczek.gz zostaje zdekompresowany.
- Niestety tym poleceniem możemy kompresować tylko 1 plik. Co jeśli chcemy skompresować cały katalog!? wystarczy wcześniej użyć polecenia `tar`.
- `tar` – archiwizuje pliki lub katalogi do archiwum `*.tar`.
- `tar -cvf paczka.tar /tmp/plik.txt` – tworzy archiwum `paczka.tar` to której dodaje `plik.txt`.
- `tar -cvf paczka.tar /tmp/` – dodaje do archiwum `paczka.tar` cały katalog `tmp`.
- `tar -zcf paczka.tar.gz /tmp` – robi dokładnie to samo, co komenda powyżej, jednak dodatkowo kompresuje archiwum przy pomocy `gzip`.
- `tar -xvf paczka.tar` – wypakowuje plik `paczka.tar`.
- `shutdown` – zamknięcie systemu, lub restart.
- `shutdown -h now` – zamknięcie systemu.
- `shutdown -h 60` – system zostanie zamknięty dopiero po upływie 60 minut.
- `ls` – listowanie zawartości katalogu.

Lista poleceń

- `ls ~/` – zostaną wyświetlone pliki i katalogi.
- `ls -a ~/` – zostaną wyświetlone wszystkie pliki i katalogi łącznie z ukrytymi (nazwa zaczyna się od '.').
- `ls -l ~/` – pliki i katalogi wyświetlone w postaci listy. Wyświetlona jest data, uprawnienia, właściciel oraz wielkość w bajtach. Sama wielkość jest trochę nieczytelna, by temu zaradzić dodajemy kolejną opcję.
- `ls -lh ~/` – zostaną wyświetlone pliki i katalogi jak wyżej. Z tym że program zamiast bajtów będzie używał jednostek *k, M, G*, odpowiednio dla kilobajtów, megabajtów i gigabajtów.
- `ls -lt ~/` – zostaną wyświetlone pliki i katalogi posortowane według daty. Wpierw wyświetlane są pliki nowsze. Gdy zamiast parametru t użyjemy r, wówczas pliki będą wyświetlane od najstarszych.
- `ls -lh ~/ | grep test` – zostaną wyświetlone tylko te pliki i katalogi które będą w swej nazwie zawierały frazę test.
- `man` – pokazuje dokumentację programu.
- `man cp` – wyświetli podręcznik polecenia cp.
- `mkdir` – tworzenie nowego katalogu.
- `mdir /tmp/test` – polecenie stworzy nowy katalog `test` w `/tmp`.
- `mv` – przenosi plik lub katalog w inne miejsce bądź zmienia nazwę.
- `mv /tmp/test /tmp/1/` – przenosi plik lub katalog o nazwie test do katalogu `/tmp/1`.
- `mv /tmp/test /tmp/test2` – zmienia nazwę pliku lub katalogu test na `test2`.
- `passwd` – zmiana hasła.
- `ps` – pokazuje jakie procesy są aktualnie wykonywane
- `ps -aux` – pokazuje procesy (programy) uruchomione przez użytkowników. Każdy proces posiada swój identyfikator *pid*, który jest przydatny w momencie zawieszenia sie programu. Gdy nie uda się go zamknąć, wówczas posługując się poleceniem `kill` wyłączamy program.
- `kill` – zabijanie procesów wyświetlanych za pomocą `ps`. Znając nr *pid* konkretnego procesu możemy go bez problemy zabić (wyłączyć).
- `kill -9 45435` – polecenie zabije proces o nr *pid* 45435.
- `pwd` – pokazuje dokładną ścieżkę do katalogu w którym aktualnie się znajdujemy.
- `reboot` – restart systemu.
- `rm` – kasowanie plików i katalogów.
- `rm /tmp/nowy.txt` – usuwa plik o nazwie `nowy.txt`.
- `rm /tmp/*txt` – usuwa wszystkie pliki które kończą się nazwą `txt`.
- `rm -r /tmp/test` – usuwa katalog o nazwie `test`.
- `who` – pokazuje kto jest aktualnie zalogowany do systemu.
- `ln` – tworzy dowiązania do plików lub katalogów.
- `ln -s /etc/X11/xorg.conf ~/xorg.conf` – utworzy w naszym katalogu domowym dowiązanie symboliczne o nazwie `xorg.conf do pliku /etc/X11/xorg.conf`
- `mount` – montowanie urządzenia czyli włączanie go w strukturę katalogu.
- `mount -t ext3 /dev/hda2 /media/hda2` – montuje dysk `/dev/hda2` z systemem plików `ext3` w katalogu `/media/hda2`.
- `mount -o loop plik.iso /media/iso` – polecenie montuje obraz iso w katalogu /`media/iso`. Jest to bardzo przydatne polecenie. Bez wypalania obrazu na płycie możemy przejrzeć jej zawartość.
- `mount -t smbfs -o username=user,password=XXX //192.168.10.1/filmy /media/filmy/` - montuje katalog udostępniany przez sambe czy też Windowsa (otoczenie sieciowe). Wchodząc do katalogu `/media/filmy` będzie nam się wydawało że cała zawartość znajduje sie na naszym dysku, a w rzeczywistości wszystkie dane znajdują się na komputerze 192.168.10.1. Wspomniałem o sambie. W jednym zdaniu to takie „otoczenie sieciowe" znane wam z Windowsa, z tym że jest bardziej dopracowane, stabilniejsze i dużo szybsze podczas kopiowania danych.
- `sudo mount -t nfs 192.168.10.1:/obrazy /media/obraz` – podobnie jak w przypadku samby która zezwala wymieniać dane między Linuksem a Windowsem, jest coś takiego jak `nfs`. Czyli udostępnianie katalogów w sieci przeznaczone dla systemów linux.

Zazwyczaj będą podane gotowe polecenia które wystarczy wkleić do terminala i uruchomić. Na pewno szybciej i prościej jest wykonać takie polecenie niż odszukać kilkanaście paczek w nakładce graficznej. Bardzo często, oprócz *apt-get* stosuje się *apt-cache*. Dzięki tej opcji możemy wyszukiwać interesujące nas pakiety. Gdy pojawi się problem ze znalezieniem jakiegoś pakietu, lub ilość informacji w terminalu zaczyna przerażać, to zawsze można skorzystać z graficznych nakładek. Jednym z takich programów pracującym w konsoli jest aptitude Rysunek 3.

Podobnie, jak w przypadku programu mc, opis aptitude jest zbędny. Interfejs jest intuicyjny oraz w naszym ojczystym języku.

Instalacja ze źródeł

Instalowanie programów ze źródeł, w przeciwieństwie do paczek deb wymaga od użytkownika znacznie więcej wiedzy i chwili zastanowienia. Niestety czasem kompilacja jest nieunikniona, jeśli w repozytorium brak paczek programu, którego potrzebujemy. Wtedy mamy dwa wyjścia: poprosić kogoś bardziej doświadczonego o stworzenie dla nas paczki lub spróbować skompilować źródła programu samemu. Przed rozpoczęciem kompilacji musimy zaopatrzyć się w podstawowe pakiety, wykonamy to poleceniem `sudo apt-get install build-essential`. W przeciwieństwie do gotowych pakietów, podczas kompilowania bardzo ważnym krokiem jest przeczytanie dokumentacji programu. Na podanym przykładzie przekonacie się, jak przydatne jest jej czytanie. Standardowo program kompiluje sie 3 poleceniami. Wpierw uruchamiamy `./configure`, następnie `make` a na sam koniec `sudo make install`. Ale czy zawsze? Przykładowo w systemie brakuje programu truecrypt który nie ma paczek w repozytorium. Ze strony domowej ściągamy plik ze źródłami *truecrypt-4.3a-source-code.tar.gz* i zapisujemy np. do katalogu `/tmp`. Źródła za-

Listing 1. Kompilacja przykładowego programu

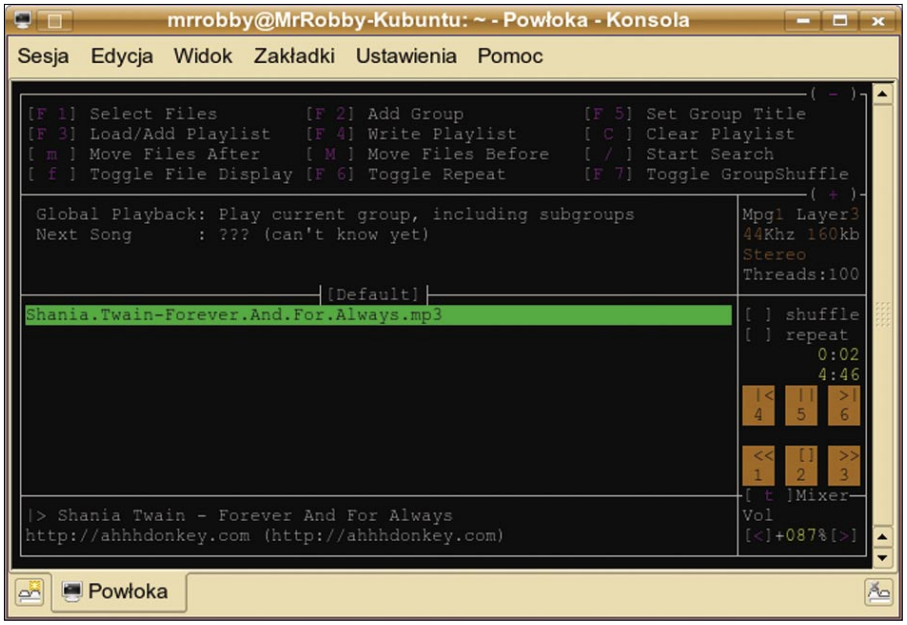
```
mrrobby@Ubuntu:/tmp/truecrypt-4.3a-source-code/Linux$ sudo ./install.sh
   Checking installation requirements...
   /dev/mapper/control not found - create? [Y/n]: Y
   Checking build requirements...
   Building kernel module... Done.
   Building truecrypt... Done.
   Testing truecrypt... Done.
   Install binaries to [/usr/bin]:
   Install man page to [/usr/share/man]:
   Install user guide and kernel module to [/usr/share/truecrypt]:
   Installing kernel module... Done.
   Installing truecrypt to /usr/bin... Done.
   Installing man page to /usr/share/man/man1... Done.
   Installing user guide to /usr/share/truecrypt/doc... Done.
   Installing backup kernel module to /usr/share/truecrypt/kernel... Done.
```

zwyczaj są skompresowane za pomocą `tar.gz`. Będąc w katalogu */tmp* wydajemy `tar -zxvf truecrypt-4.3a-source-code.tar.gz` celem wypakowania plików z archiwum. Jak przeczytamy w dokumentacji programu *Readme.txt*, przed kompilacją potrzebujemy zainstalować źródła kernela. Poleceniem `uname -r` sprawdzamy wersję, a następnie instalujemy `sudo apt-get install linux-source-2.6.20` gdzie 2-6.20 to numer jaki wyświetlił się nam po wcześniejszym poleceniu. Domyślnie źródła są skompresowane. Przechodzimy do katalogu `/usr/src`, rozpakowujemy źródła i wykonujemy *symlink* (dowiązanie do pliku – na potrzeby tego tekstu możemy przyjąć, że jest to odpowiednik skrótu w systemie MS Windows) `sudo ln -s linux-source-2.6.20 linux`. Od tej pory nasz system jest gotowy do kompilowania pakietu. Wracamy z powrotem do katalogu, w którym to mamy wypakowany nasz program. Według wskazówek z dokumentacji, wchodzimy do katalogu `cd linux` a następnie uruchamiamy proces kompilacji `sudo ./install.sh`. Jeśli wszystko pójdzie bez żadnych problemów, to powinniśmy otrzymać informacje podobne do tych z Listingu 1.

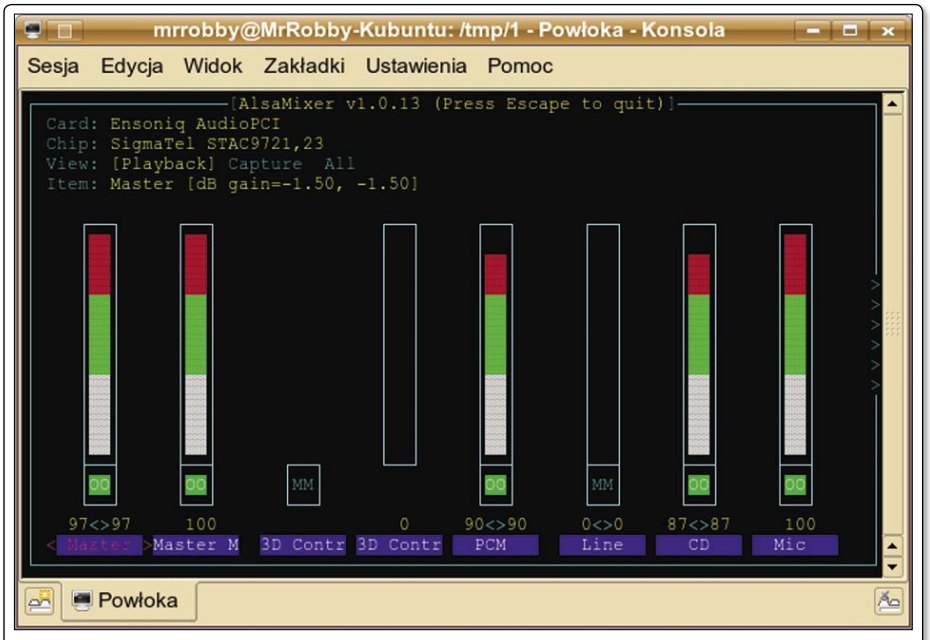
To był przykład nietypowej kompilacji ukazujący, jak ważne jest czytanie dokumentacji. Przy standardowej kompilacji wpierw uruchamiamy skrypt `./configure`, który to sprawdzi za nas, czy w systemie znajdują się wszystkie programy i biblioteki potrzebne do przeprowadzenia poprawnej kompilacji. Uruchomienie `./configure --help` wyświetli nam możliwe parametry, z jakimi możemy instalować program m.in. parametr odpowiadający za katalog, w którym to ma znajdować się program. Gdy uruchomimy `./configure` bez parametrów, to skrypt przyjmie wartości domyślne, ustawione przez autora programu. Jeśli skrypt przejdzie bez żadnych błędów, to możemy przejść do kolejnego kroku, którym jest kompilacja, uruchamiana poleceniem `make`. Jeśli wszystko przebiegło bez błędów możemy instalować program w systemie `sudo make install`.

Multimedia w konsoli

Konsola to nie tylko poruszanie się po strukturze katalogów, czy edycja plików. Wbrew powszechnemu przekonaniu równie dobrze bez udziału środowiska graficz-



Rysunek 4. Mp3blaster w akcji



Rysunek 5. Alsamixer

nego możemy korzystać z niektórych dobrodziejstw multimedialnych. Myślę tu o słuchaniu muzyki czy też oglądaniu filmów. Oglądanie filmów w konsoli? Tak, jest to możliwe i nie ma z tym najmniejszego problemu. Do odsłuchania jednego pliku muzycznego mp3 polecam program *mpg123*. Zgodnie z przypuszczeniami `sudo apt-get install mpg123` zainstaluje go dla nas w systemie. Podobnie ma się sytuacja w sprawie równie popularnego formatu *ogg*, który odsłuchujemy programem *ogg123*. Wywołanie programu w obu przypadkach jest identyczne. W przypadku chęci odsłuchania większej ilości muzyki, polecam bardzo przyjemny odtwarzacz *mp3blaster* pokazany na Rysunku 4, który odtwarza oba wymienione formaty.

Pakiet alsamixer pokazany na Rysunku 5, pozwoli nam ustawić głośność naszej karty muzycznej.

W przypadku chęci oglądania filmu w konsoli, musimy zainstalować odtwarzacz `sudo apt-get install mplayer` oraz dodatkowo skonfigurować *framebuffer* (w skrócie fb). Niestety konfiguracja fb, to temat na osobny artykuł. Chciałem tylko napomnieć, że konsola posiada możliwość wyświetlania filmu, który w przypadku działającego fb uruchamiamy jednym, prostym poleceniem `mplayer -vo fbdev film.avi`.

Mam nadzieję że po tym krótkim artykule przekonacie się do korzystania z konsoli, lub przynajmniej nauczycie się jej podstaw, celem naprawy uszkodzonego zazwyczaj z winy użytkownika systemu. ■

W Sieci

- Bash:
  http://pl.wikipedia.org/wiki/Bash
  http://wazniak.mimuw.edu.pl/index.php?title=%C5%9Arodowisko_programisty/Wprowadzenie_do_Basha
- Midnight Commander:
  http://www.ibiblio.org/mc/
- Apt:
  http://pl.wikipedia.org/wiki/APT
- Aptitude:
  http://pl.wikipedia.org/wiki/Aptitude
- Mp3blaster:
  http://mp3blaster.sourceforge.net/
- Mplayer:
  http://www.mplayerhq.hu