

# Wirtualizacja

Systemy operacyjne  
9 listopada 2007



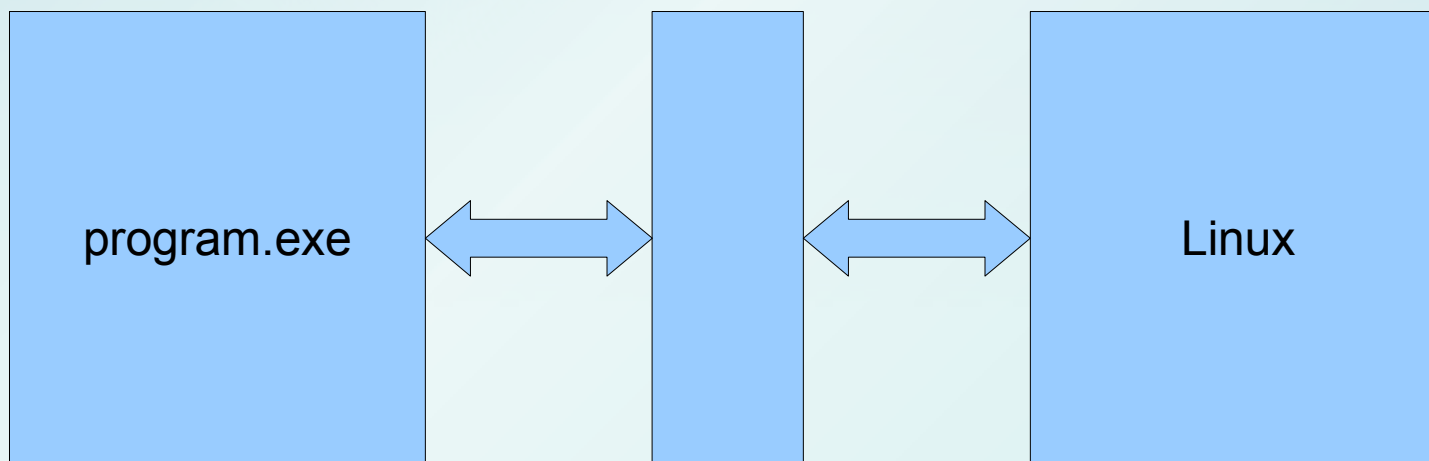
Aleksander Jankowski  
Paweł Matysiak  
Aleksandra Murawska

## Wirtualizacja - teoria

- Wirtualizacja ogólnie i jej główne aspekty
- Pewne problemy z x86
- Nowe rozwiązania
  - Intel VT-x/-d
  - AMD-V
- Zastosowania maszyn wirtualnych

## Wirtualizacja ogólnie

- Tworzenie dodatkowej warstwy pośrednika pomiędzy elementami komputera, które do siebie nie pasują.
- Tworzenie nakładki na element komputera, który chcemy mieć inny (np. powielony).



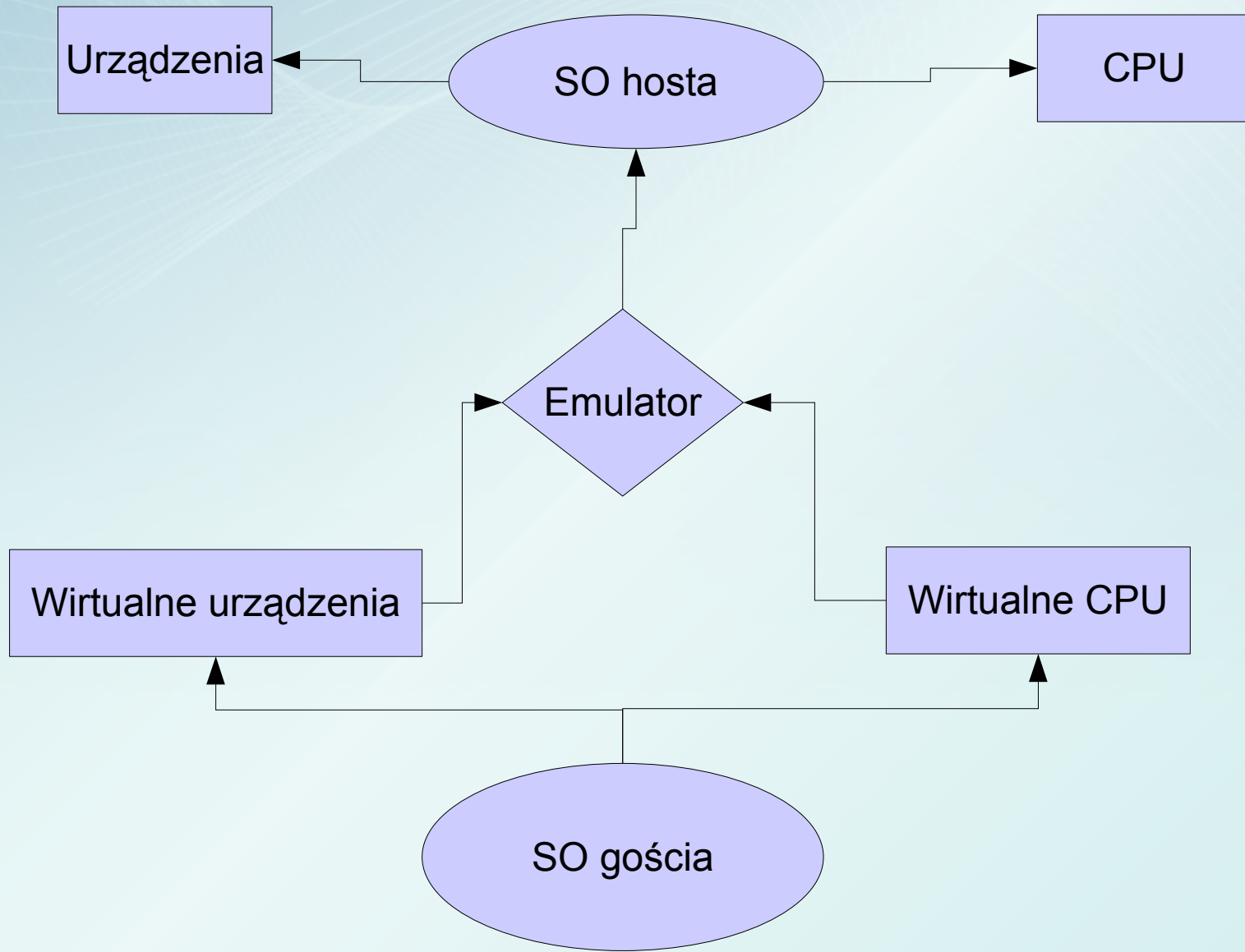
## Podstawowe aspekty wirtualizacji

- O wirtualizacji można mówić na wielu poziomach i w wielu aspektach.
- Najbardziej podstawowe aspekty wirtualizacji to:
  - Emulacja
  - Emulacja API
  - Pełna wirtualizacja

# Emulacja

- Polega na tym, że na konkretnym sprzęcie (z działającym na nim systemem operacyjnym) tworzymy całkowitą iluzję innego sprzętu.
- System operacyjny i aplikacje działające na takim sprzęcie nie mają szans na zorientowanie się, że są oszukiwane.
- Każda instrukcja jest wirtualizowana, przez co emulowany system działa dużo wolniej niż na rzeczywistej maszynie.

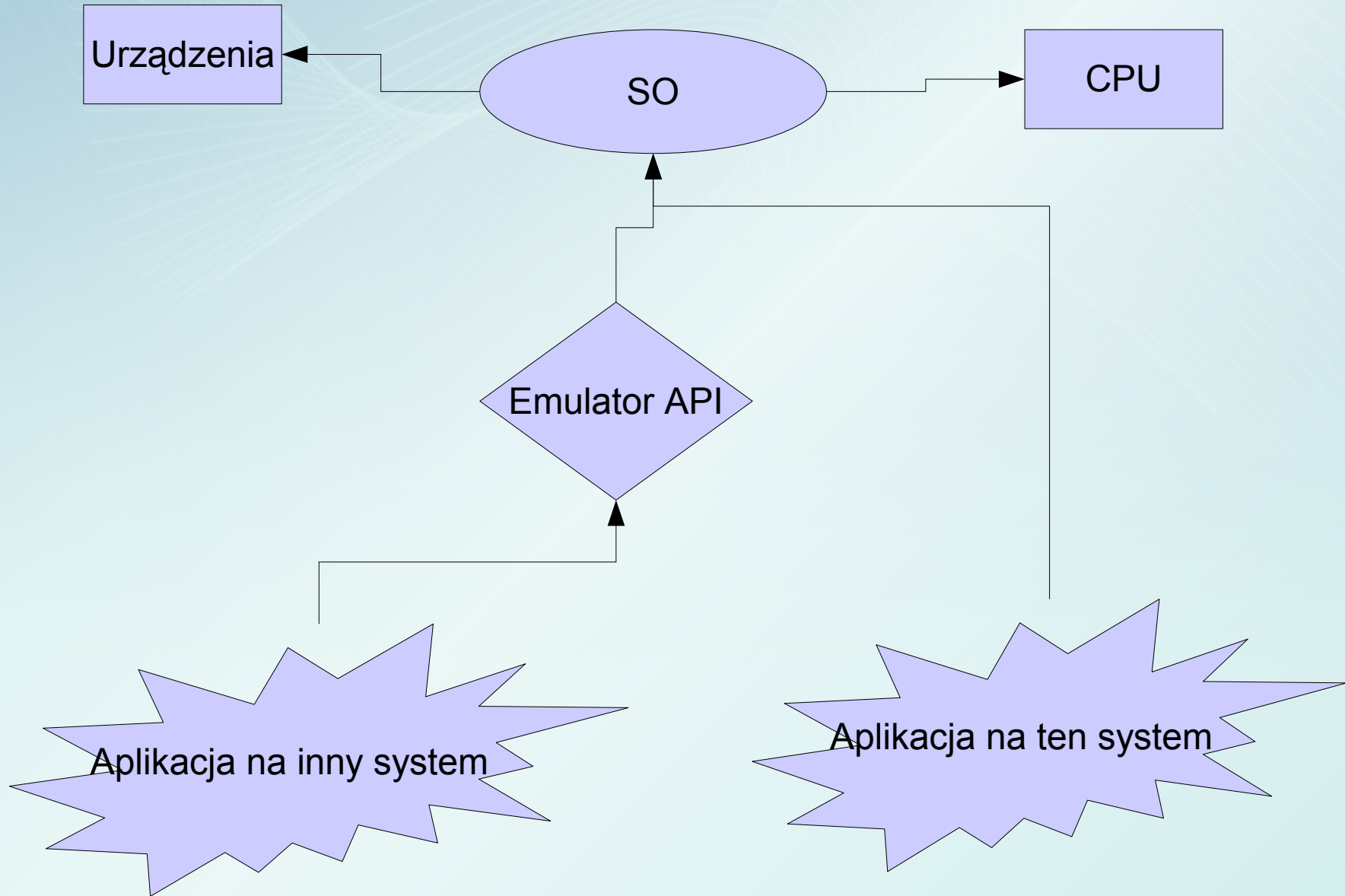
# Emulacja – schemat przepływu



## Emulacja API

- Na ogół aplikacje napisane na konkretny system operacyjny korzystają tylko z określonego zestawu funkcji (API), odpowiedniego dla danego systemu operacyjnego.
- Aby taka aplikacja mogła działać na innym systemie operacyjnym, wystarczy zaimplementować w nim odpowiednie funkcje.

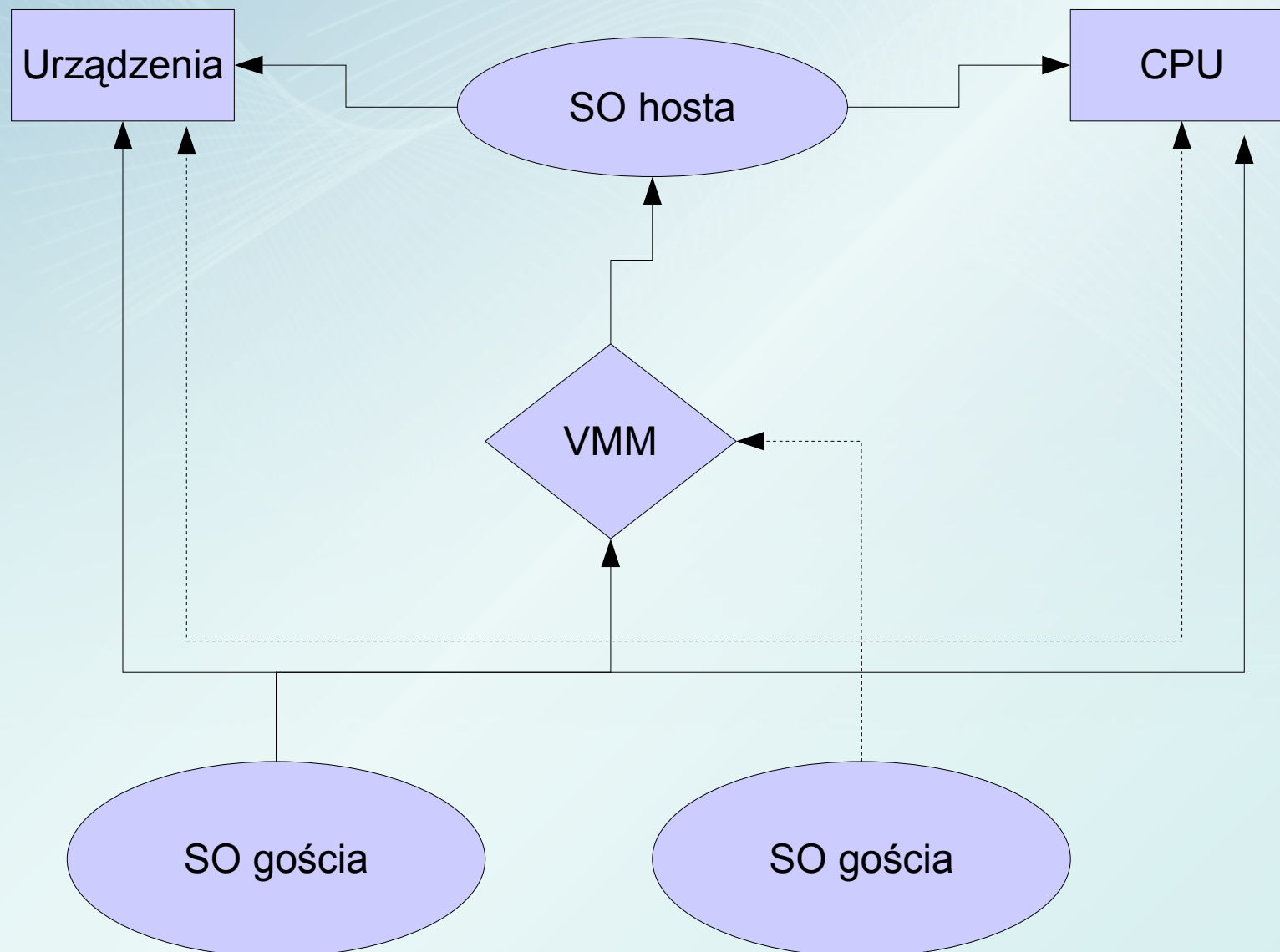
# Emulacja – schemat przepływu



## Pełna wirtualizacja

- Polega na uruchomieniu kilku systemów na jednym sprzęcie (kompatybilnym z tymi systemami) w taki sposób, aby jak najwięcej instrukcji wykonywało się bezpośrednio.
- Maksymalizuje korzystanie z rzeczywistego sprzętu i oprogramowania - maszyna wirtualizuje tylko niebezpieczne instrukcje.
- Najwydajniejszy sposób wirtualizacji.

# Pełna wirtualizacja – schemat przepływu



## Kryteria skuteczności maszyny wirtualnej

- Kryteria te sformułowali w 1974 roku Gerald J. Popek i Robert P. Goldberg.
- Wirtualna maszyna, by być skuteczna, musi spełniać trzy warunki:
  - **Odpowiedniość** – program działający na maszynie wirtualnej musi zachowywać się w dokładnie taki sam sposób, jak na rzeczywistym sprzęcie
  - **Kontrola zasobów** – wirtualna maszyna musi w pełni kontrolować wszystkie zasoby, które są wirtualizowane
  - **Wydajność** – większa część instrukcji musi być wykonywana bez udziału maszyny wirtualnej.

## Wyróżnione rodzaje instrukcji

- **Instrukcje wrażliwe** – mogą zmieniać konfigurację zasobów systemu operacyjnego, bądź też korzystają z tej konfiguracji.
- **Instrukcje uprzywilejowane**
  - Jeśli system jest w trybie użytkownika, to powodują przerwanie lub wywołanie systemowe.
  - Jeśli system jest w trybie jądra, nie powodują przerwania ani wywołania systemowego.

## Twierdzenie Popka-Goldberga

Dla każdego standardowego komputera trzeciej generacji wirtualna maszyna może zostać skonstruowana, jeśli zbiór instrukcji wrażliwych jest podzbiorem zbioru instrukcji uprzywilejowanych.

## Poziomy ochrony

- Nazywane po angielsku *priviledge levels*.
- Na architekturze x86 są 4 poziomy ochrony, z których
  - 0 – przeznaczony dla jądra
  - 3 – przeznaczony dla aplikacji
- Na różnych poziomach ochrony procesy mają dostęp do różnych zasobów, najwięcej praw jest na poziomie 0.
- Ring model – uprawnienia na poziomach 0, ..., n zawierają się w uprawnieniach na poziomach 0, ..., n-1.
- Stosowane skróty: DPL (Descriptor PL), RPL (Requested PL), CPL (Current PL).

## Problemy z wirtualizacją niektórych instrukcji x86

- Niestety nie wszystkie architektury spełniają warunek twierdzenia Popka-Goldberga.
- W architekturze x86 jest aż 17 instrukcji, które są wrażliwe, a nie są chronione.

## Instrukcje korzystające z wrażliwych rejestrów

- SGDT, SIDT, SLDT – te instrukcje odwołują się do rejestrów GDTR, IDTR, LDTR.
  - Wykonywane bezpośrednio, mogłyby przekazać wartości odpowiadające rejestrom systemu operacyjnego hosta.
  - Rejestry są częściowo chronione – tylko do pisania.
- SMSW – korzysta ze słowa stanu procesora
  - Ta instrukcja jest tylko po to, aby zachować kompatybilność z Intel 286 – powinno się używać MOV

## Instrukcje korzystające z wrażliwych rejestrów

- PUSHF, POPF (PUSHFD, POPFD) – korzystają z rejestru EFLAGS
  - PUSHF pozwala sprawdzać zawartość tego rejestru, który podobnie, jak słowo stanu procesora zawiera informacje o trybie pracy oraz stanie procesora
  - POPF pozwala zmieniać zawartość EFLAGS, jednak bez odpowiednich uprawnień dla niektórych flag ani się to nie uda, ani nie wygeneruje wyjątku, dla niektórych się uda, ale to też źle

## Instrukcje odnoszące się do poziomów ochrony systemu

- LAR, LSL, VERR, VERW – potrzebują informacji o obecnym poziomie ochrony
  - Jądro najczęściej zakłada, że jest wykonywane na najwyższym poziomie ochrony, jednak wykonywane na maszynie wirtualnej ma niższy poziom – te instrukcje mogą się źle wykonać
- POP, PUSH – są zależne od aktualnego poziomu ochrony
  - PUSH może zatrzymać proces, gdy jest wykonywany na maszynie wirtualnej

## Instrukcje odnoszące się do poziomów ochrony systemu c.d.

- CALL, JMP, INT n, RET (IRET, IRETD)
  - w instrukcji CALL problem stanowią odwołania do innych poziomów ochrony, JMP – podobnie (jedyną różnicą jest to, że JMP nie zapamiętuje adresu powrotu)
  - RET – przy powrocie rejestry DS, ES, FS i GS są czyszczone, jeśli odnoszą się do segmentów, do których przestajemy mieć dostęp
  - INT dodatkowo korzysta z EFLAGS
- STR – pozwala sprawdzać wartość RPL (Requested Privilege Level)
- MOVE – przy korzystaniu z wrażliwych rejestrów (kontrolnych – CS, SS)

## Technologie wspierające wirtualizację

- Na procesorach x86 dostępne są od pewnego czasu:
  - Intel Virtualization Technology (VT)
  - AMD-Virtualization (AMD-V).
- Są to technologie, które usuwają wiele problemów napotykanym przy tworzeniu VMM dla architektury x86.
- Dla architektury 32-bitowej mamy VT-x (z VT-d)
- Dla architektury 64-bitowej – VT-i, AMD-V
- Pozwala systemom operacyjnym pracującym na VM na pracę na takim poziomie ochrony, do jakiego zostały zaprojektowane.

# Technologia VT-x

- Dwa tryby pracy (VMX root operation, VMX non-root operation)
- Przejścia pomiędzy nimi:
  - VM entry
  - VM exit
- Kontrolowane przez specjalną strukturę danych: VMCS (zawiera guest-state area i host-state area oraz VM-execution control fields)
- Dodatkowe instrukcje:
  - VMPTRLD, VMPTRSD, VMCLEAR, VMREAD, VMWRITE, VMCALL, VMLAUNCH, VMRESUME, VMXOFF, VMXON

## Technologia VT-d (Virtualization for directed I/O)

- Wprowadza technologię IO MMU (Input/Output Memory Management Unit), aby zapobiegać używaniu DMA.
- Zapewnia sposób konfiguracji dostarczania przerw do maszyn wirtualnych.
- IO MMU jest pośrednikiem pomiędzy dyskiem a pamięcią główną, podobnie jak MMU jest pośrednikiem między procesorem a pamięcią główną.

## Technologia AMD-V

- Rozszerzenie dla 64-bitowej architektury x86.
- Korzysta z technologii IO MMU.
- VMCB - odpowiednik VMCS – zawiera informacje o tym, jakie akcje powodują przejście do VMM oraz struktury wskazujące na to, które informacje są widoczne dla systemu gościa.
- VMRUN (entry i exit), VMCALL
- Stan hosta zapamiętywany przy użyciu MSR (Model Specific Register) – składnik VMCB.

# Zastosowania wirtualizacji

- Wirtualizacja często ułatwia życie:
  - Gdy chcemy uruchomić jednocześnie, na jednym komputerze, dwie aplikacje napisane na różne systemy.
  - Gdy nie mamy sprzętu, na którym działa system operacyjny, który chcemy uruchomić (bo na nim z kolei chcemy uruchomić aplikację - ulubioną grę z dzieciństwa - przeznaczoną konkretnie dla tego systemu).
  - Gdy sprzęt, który chcemy oprogramować, jest poza zasięgiem – np. nie istnieje fizycznie.
  - Gdy chcemy udostępnić użytkownikom (np. w sieci lokalnej) serwer, dający możliwość pracy na wirtualnych maszynach, dostępnych z każdego miejsca.

# Testowanie

- Pisząc software często nie mamy dostępu do wszystkich systemów, czy nawet wersji jednego systemu, na którym ten software ma działać.
- Bez sensu jest co chwila restartować komputer, aby uruchomić inny system tylko po to, żeby przetestować na nim jedną funkcję.
- Na szczęście wirtualizacja stwarza możliwość uruchomienia jednocześnie, na jednym komputerze, aplikacji przeznaczonych dla różnych systemów operacyjnych.

## Pisanie niebezpiecznych programów

- Niebezpieczne są np. modyfikacje jądra i sterowniki urządzeń.
- Źle napisane mogą zepsuć coś w systemie tak, że trzeba go będzie ponownie instalować (a to może znacząco wydłużyć czas pracy).
- Uruchomienie takiego oprogramowania na maszynie wirtualnej może nie tylko oszczędzić sporo czasu, ale też ułatwić debugowanie (możemy z zewnątrz popatrzeć, co się stało).



# VirtualBox®

## Welcome to VirtualBox.org!

innotek VirtualBox is a family of powerful x86 [virtualization](#) products for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers, it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL). See "[About VirtualBox](#)" for an introduction; see "[innotek](#)" for more about our company.

Presently, VirtualBox runs on Windows, Linux and Macintosh hosts and supports a large number of [guest operating systems](#) including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista), DOS/Windows 3.x, Linux (2.4 and 2.6), and OpenBSD.

VirtualBox is being actively developed with frequent releases and has an ever growing list of features, supported guest operating systems and platforms it runs on. VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while innotek ensures the product always meets professional quality criteria.

On this site, you can find sources, binaries, documentation and other resources for VirtualBox. If you are interested in VirtualBox (both as a user, or possibly as a contributor), this website is for you.



[About](#)

[Screenshots](#)

[Downloads](#)

[Documentation](#)

[End-user docs](#)

[Technical docs](#)

[Contribute](#)

[Community](#)

### News Flash

- **New Oct 23, 2007**  
**VirtualBox exhibits at SYSTEMS!**  
Make sure to stop by the VirtualBox booth (Hall B2, booth 110 demo point 18)
- **New Oct 18, 2007**  
**VirtualBox 1.5.2 released!**  
innotek has released an update to VirtualBox.
- **New Jul 18, 2007**  
**VirtualBox for OS X updated!**  
innotek has released the second beta of VirtualBox for OS X with USB support.
- **New Apr 10, 2007**  
**VirtualBox gets multilingual!**  
innotek has provided [instructions](#) on how to translate VirtualBox. Please contribute!
- **New Jan 16, 2007**  
**User FAQ.**  
We now have a list of [Frequently Asked Questions \(FAQ\) for end users](#) available.

[More information...](#)

## 2 Wersje VirtualBoxa:

### VirtualBox OSE Edition:

- opublikowany na edycji GPL,
- udostępniony jako kod źródłowy,
- zawiera większość opcji VirtualBox PUEL Edition.

### VirtualBox PUEL Edition:

- darmowy dla zastosowań akademickich, osobistych oraz do ewaluacji produktów,
- udostępniony w postaci binarek/paczek na większość platform (Windows, Mac OS X (intelowskie!), może Linuxów) w architekturach x86/x86-64 (bez IA64),
- w odróżnieniu od OSE wspiera RDP, USB i iSCSI.

# VirtualBox w Środowisku WinXP:

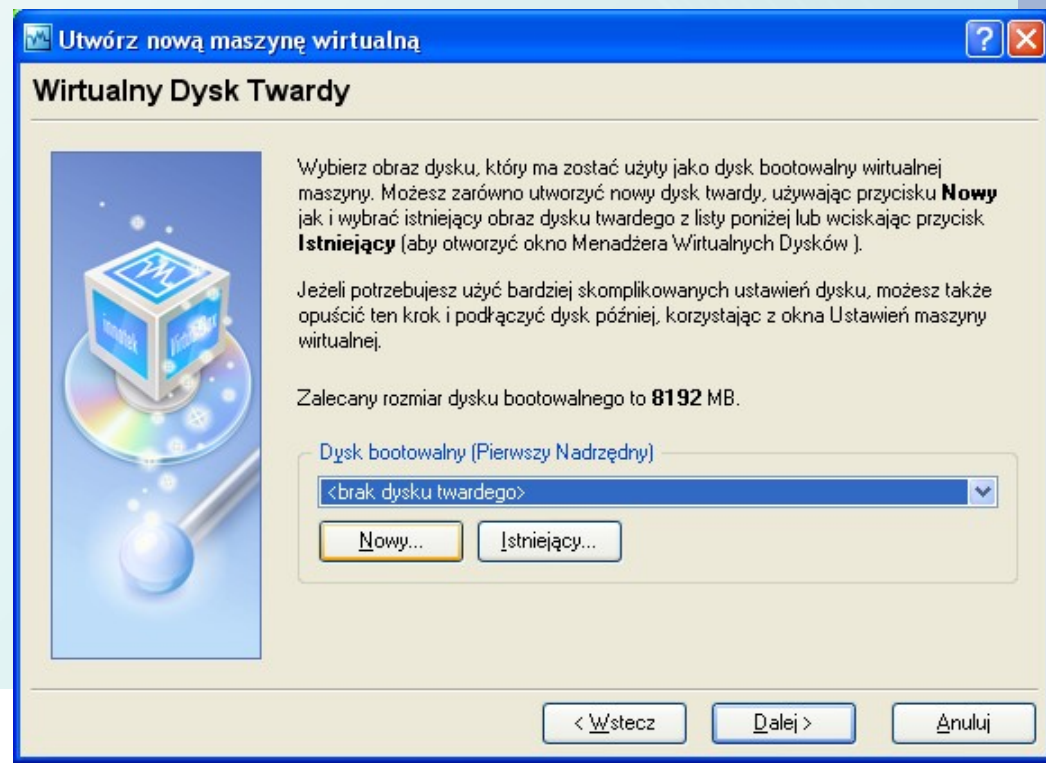
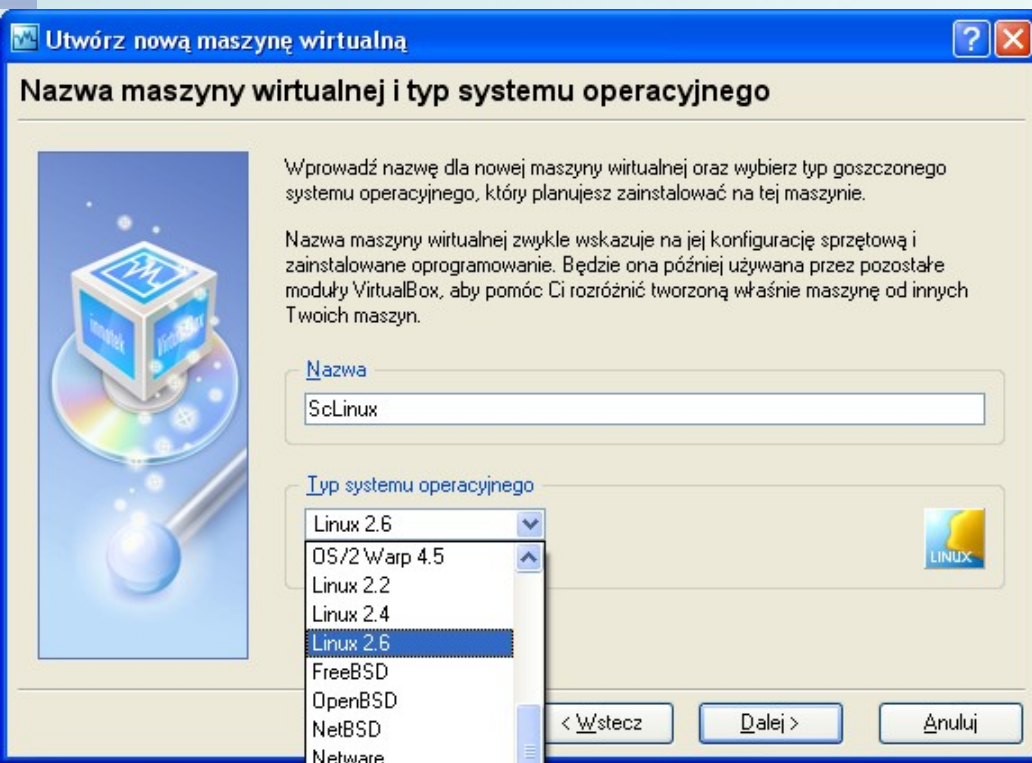
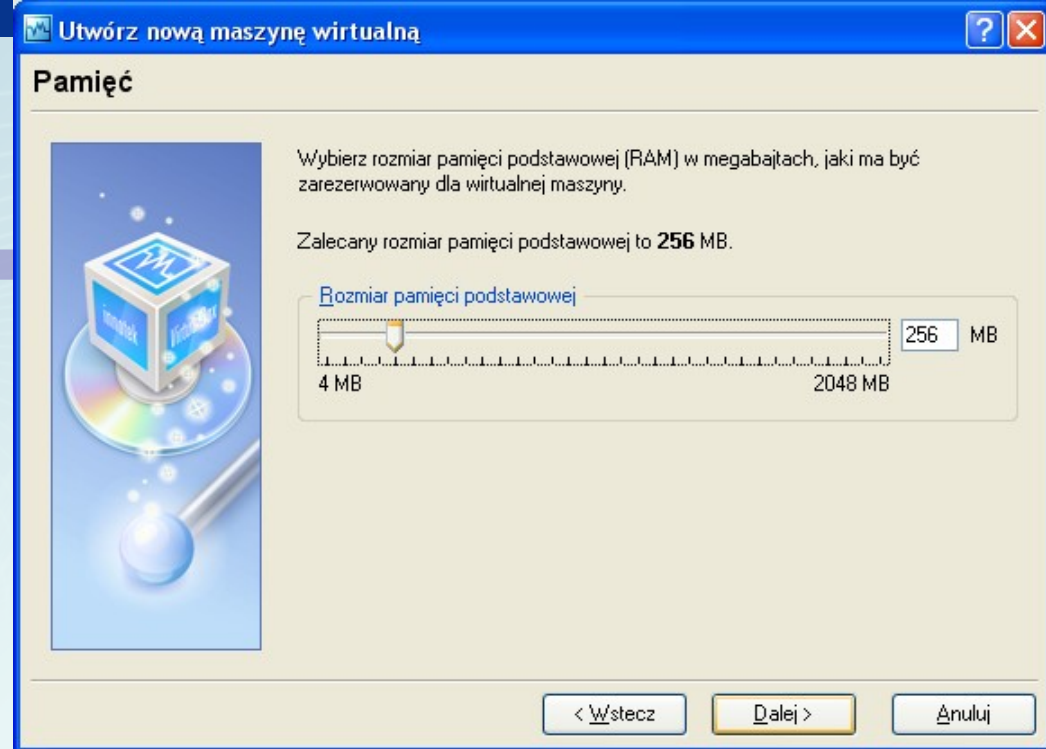
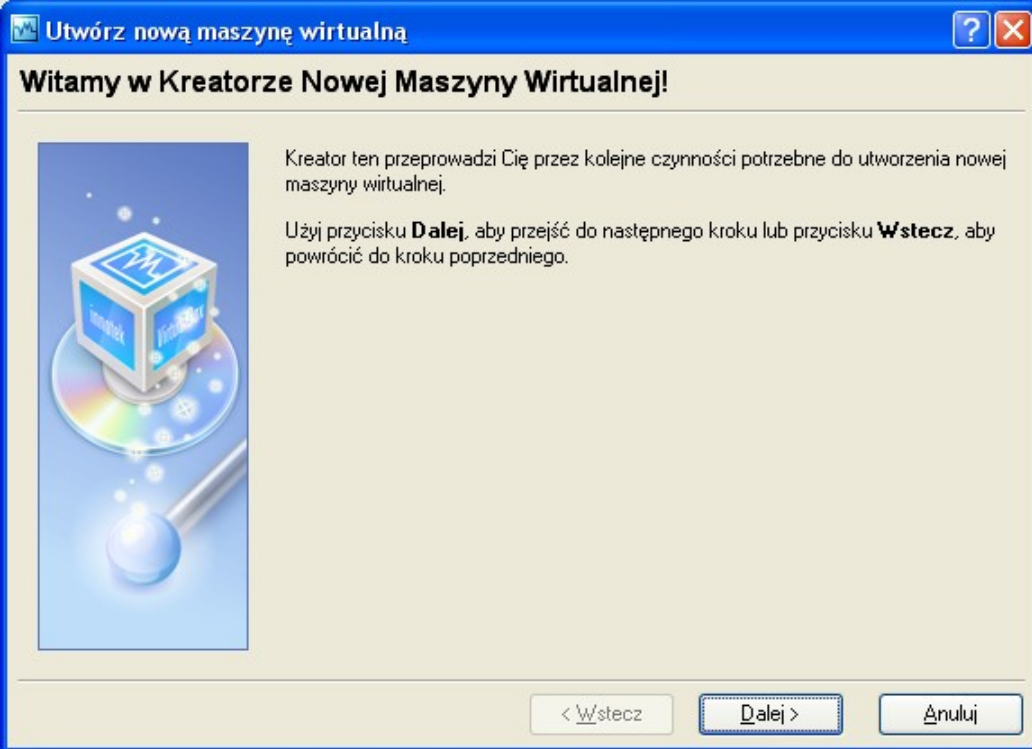
The screenshot displays the VirtualBox configuration window for a virtual machine named 'Kubuntu'. The interface is in Polish. On the left, a list of operating systems is shown, with 'Kubuntu (Ubuntu on Kubuntu on WindowsXP)' selected and its status set to 'Uruchomiona' (Running). Below it are 'Linux 2.6', 'Scientific Linux', and 'Ubuntu', all with status 'Wyłączona' (Stopped).

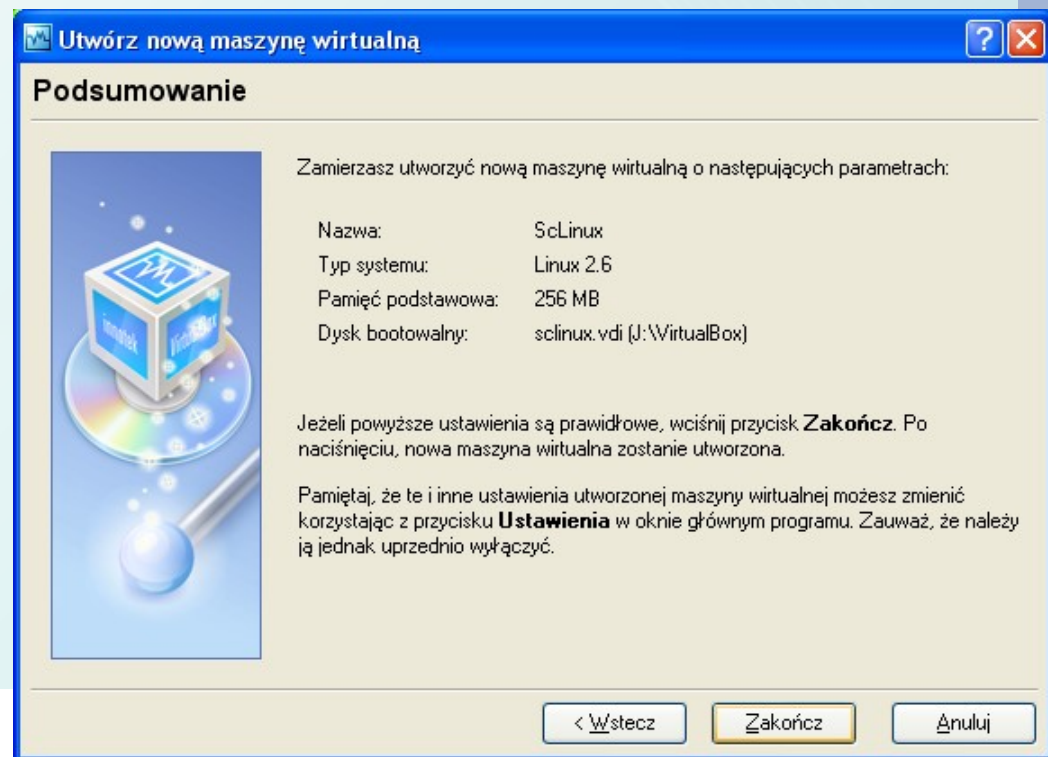
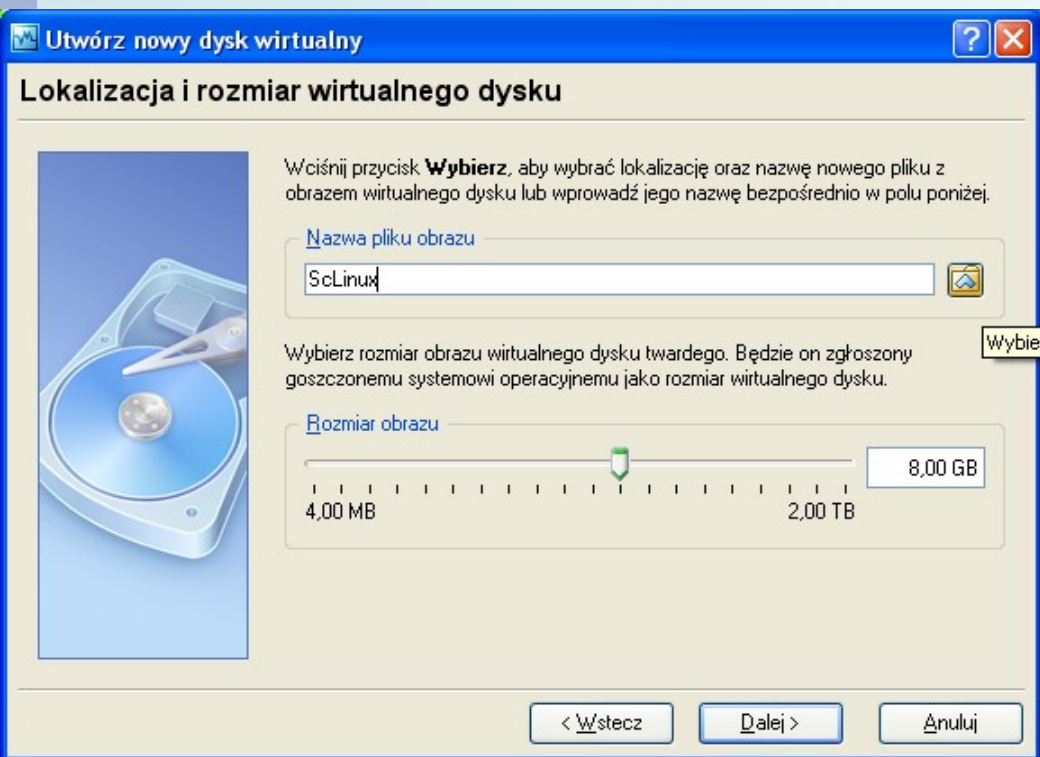
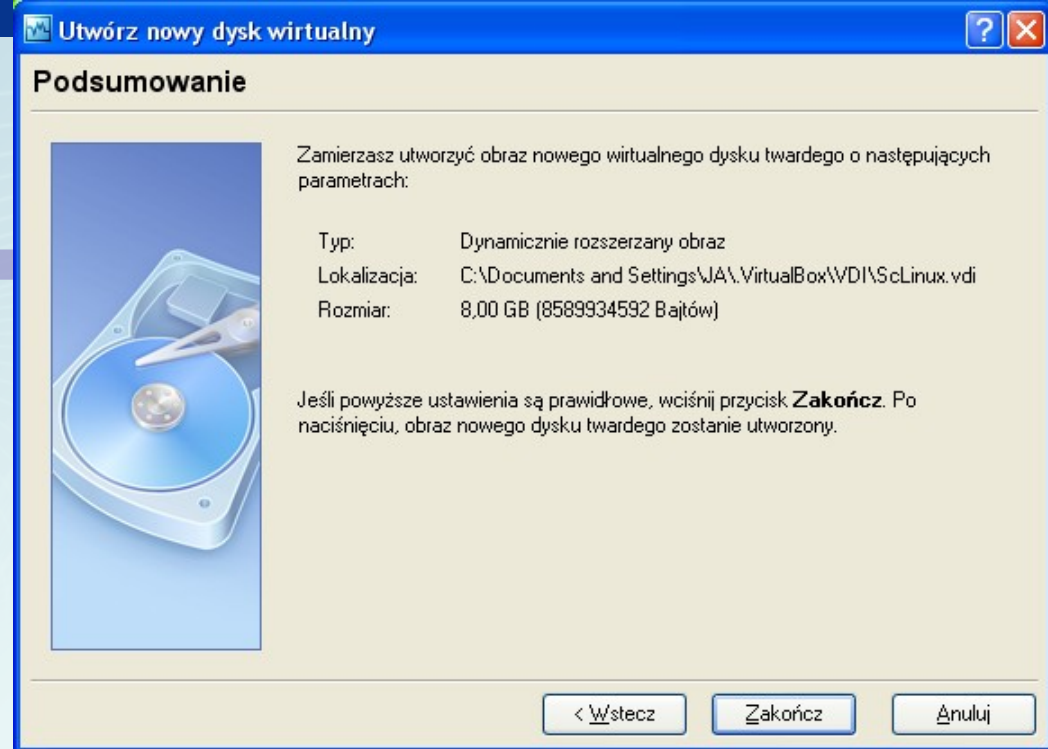
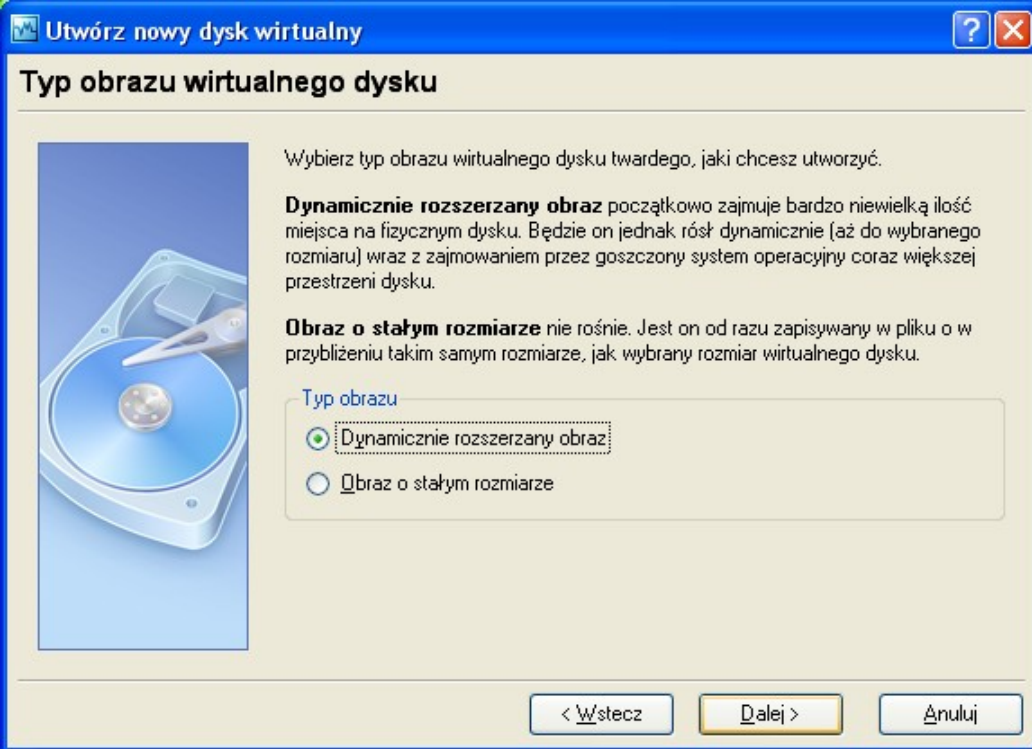
The main area shows the configuration details for the selected VM, organized into sections:

- Ogólne** (General):
  - Nazwa: Kubuntu
  - Typ systemu: Linux 2.6
  - Pamięć podstawowa: 512 MB
  - Pamięć wideo: 16 MB
  - Kolejność bootowania: Dyskietka, CD-DVD-ROM, Dysk twardy
  - ACPI: Włączone
  - ID APIC: Wyłączone
- Dyski twarde** (Hard Disks):
  - Pierwszy Nadrzędny: kubuntu.vdi [Różnicowy, 8,00 GB]
- CD/DVD-ROM**:
  - Obraz: ubuntu-7.10pl-desktop-i386\_ftp.iso
- Dyskietka**:
  - Niezamontowana
- Dźwięk**:
  - Wyłączony
- Sieć**:
  - Karta 0: NAT
- Porty szeregowo**:
  - Wyłączone
- USB**:
  - Wyłączony
- Współdzielone foldery**:
  - Brak
- Zdalny pulpit**:
  - Wyłączony

## Instalacja maszyny wirtualnej

- Jest zautomatyzowana i całkowicie okienkowa
- Sprowadza się do wybrania etykiety VM, typu Systemu Operacyjnego (do wyboru: DOS, Win 3.10/95/98/Me/2k/XP/Server2003/Vista, OS 2 Warp 3/4/4.5, Linux 2.2/2.4/2.6, Free/Open/NetBSD, Netware, Solaris i L4 lub nieznany/inny), zarezerwowania pamięci RAM dla danej maszyny, zlokalizowania dysku twardego dla VM (lub nie) i tyle.
- Następnie możemy dalej skonfigurować VM doprecyzowując ustawienia dotyczące karty sieciowej, współdzielonych folderów, zamontowanych napędów optycznych (maszynowe lub ISO – w odróżnieniu od np. Daemon Toolsa nie można używać zipów etc). itp.







### Przykładowe efekty instalacji:

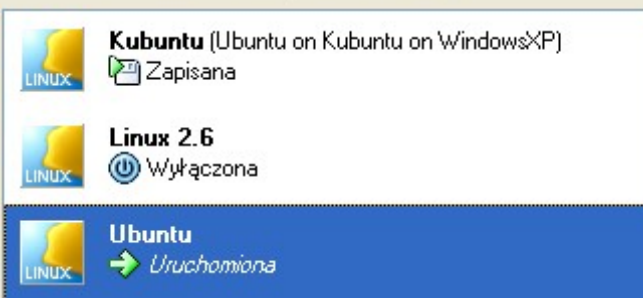
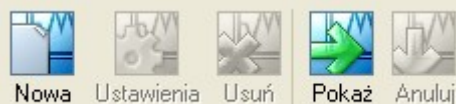
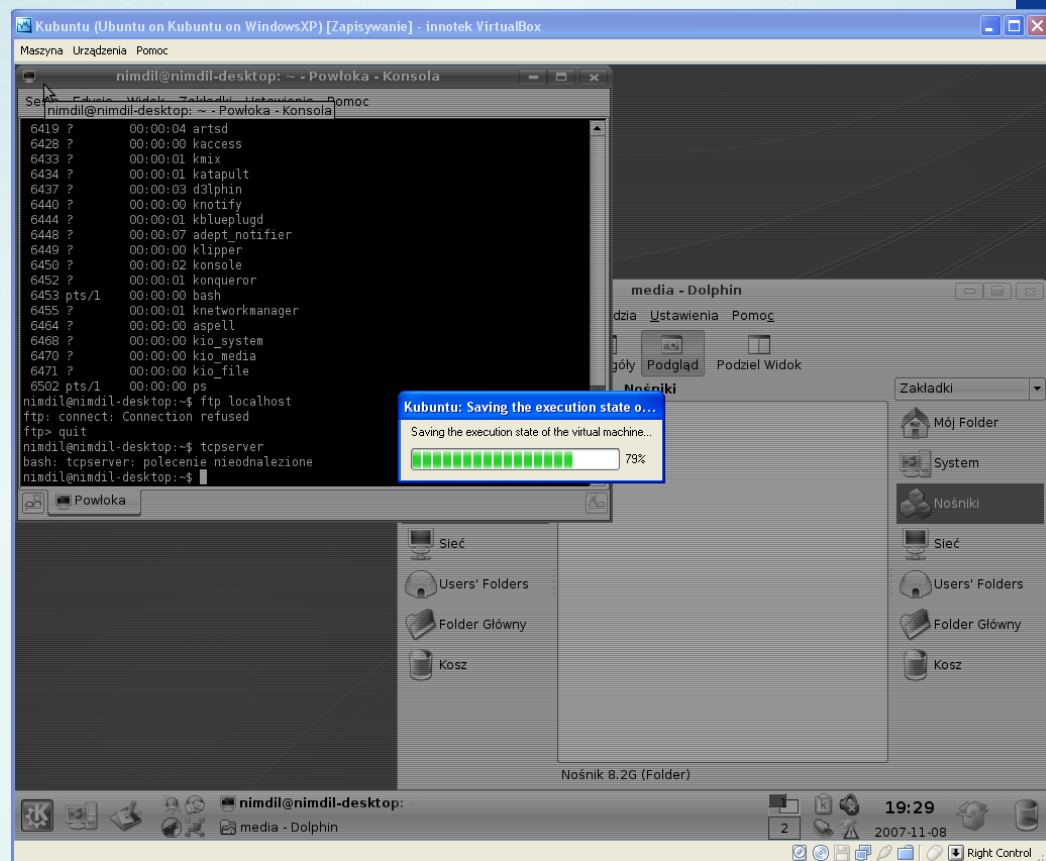
- Debian 4.0 instalowany z płyty C – bp
- Ubuntu 7.10 instalowany z obrazu ISO – instalator nie umiał samodzielnie dobrać partycjonowania
- Kubuntu 7.10 instalowany z obrazu ISO - bp

## Interfejs

- W momencie aktywacji okna VM, ta przechwytuje wszystkie dane przesłane z klawiatury. W momencie kliknięcia na okno VM kursorem myszki, VM przechwytuje również ruchy myszki.
- Uwolnić myszkę można specjalnie ustalonym klawiszem (domyślnie: pr. ctrl) lub skrótem systemowym np. alt-ctrl-del

# Zapisywanie stanu VM

- Maszynę wirtualną można w dowolnym momencie zapisać, a następnie przywrócić. Zapisany stan można też anulować odblokowując normalną inicjację maszyny.
- W czasie działania VM można wykonać tzw. „zdjęcie” czyli zapisanie bieżącego stanu bez wyłączenia.
- Kiedy VM jest wyłączona, „zdjęcie” można załadować jako zapisany stan analogiczny z w/w.
- Stabilność tego systemu – w założeniu mającego ułatwić pracę – jest mocno nierewelacyjna.



# Zarys realizacji domyślnych ustawień połączeń sieciowych (NAT)

- Z punktu widzenia systemu operacyjnego hosta, wszelkie połączenia sieciowe realizuje proces VirtualBox, a nie proces maszyny wirtualnej.

# Ustanawianie połączenia SSH host->gość

- Instalujemy serwer SSH na VM gościa,
- Konfigurujemy wybrany OS ale go nie uruchamiamy
- Ustanawiamy przekierowanie:

```
VBoxManage setextradata "Ubuntu" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/guestssh/Protocol" TCP
VBoxManage setextradata "Ubuntu" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/guestssh/GuestPort" 22
VBoxManage setextradata "Ubuntu" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/guestssh/HostPort" 2222
```

- Drugi parametr to etykieta VM
- W trzecim parametrze fraza guestssh jest dowolna
- Czwarty parametr to do wyboru TCP lub UDP a następnie port hosta i port gościa.

# Ustawianie połączenia SSH host->gość, c.d.

- Otrzymujemy reakcję na w/o skrypt:

```
C:\Program Files\innotek VirtualBox>go2.bat

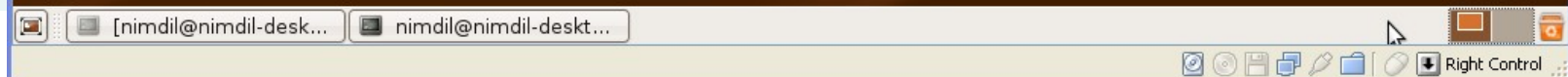
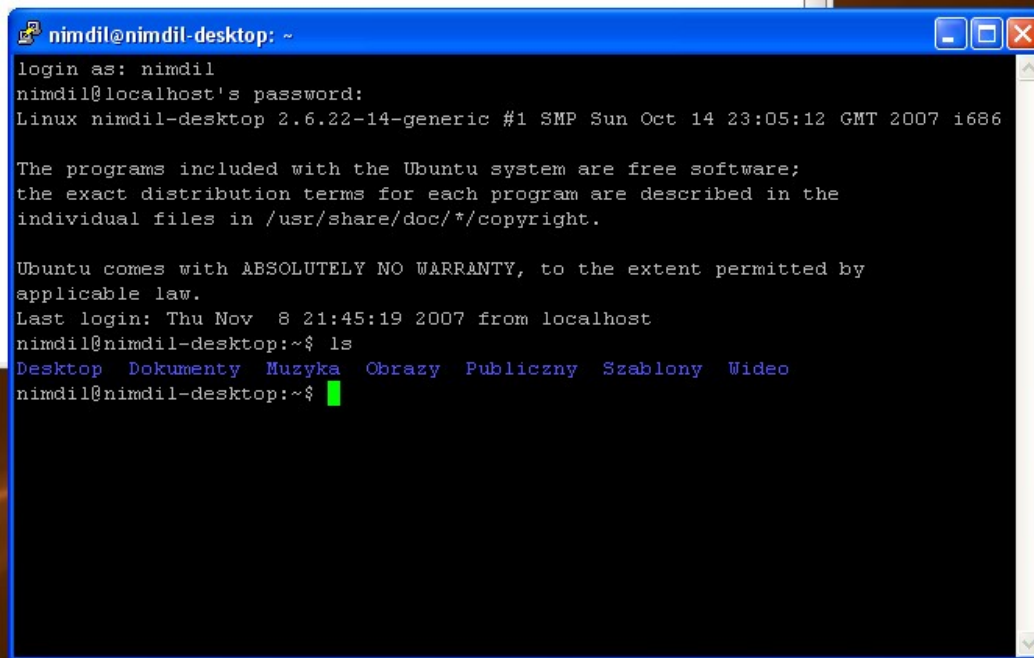
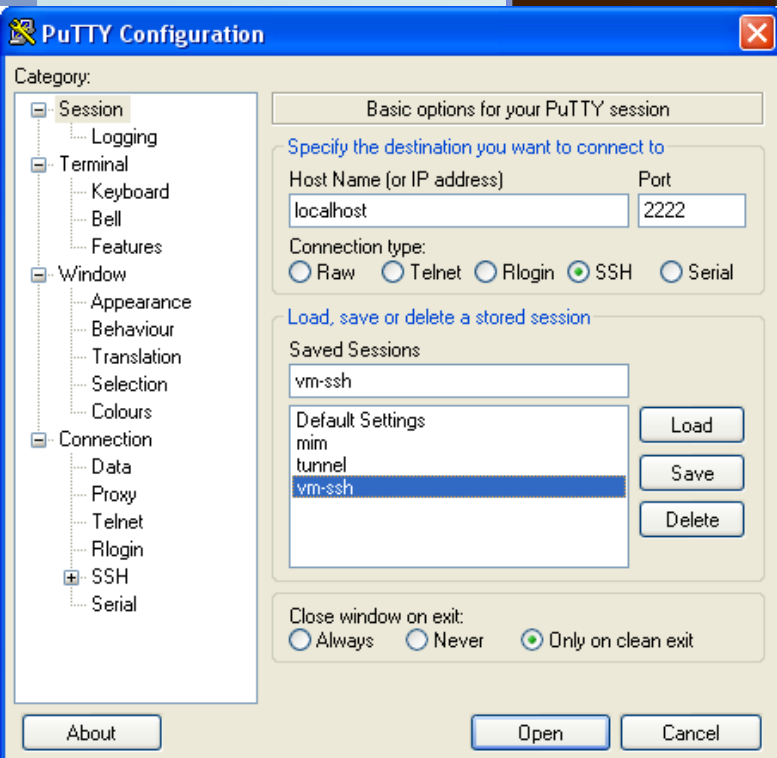
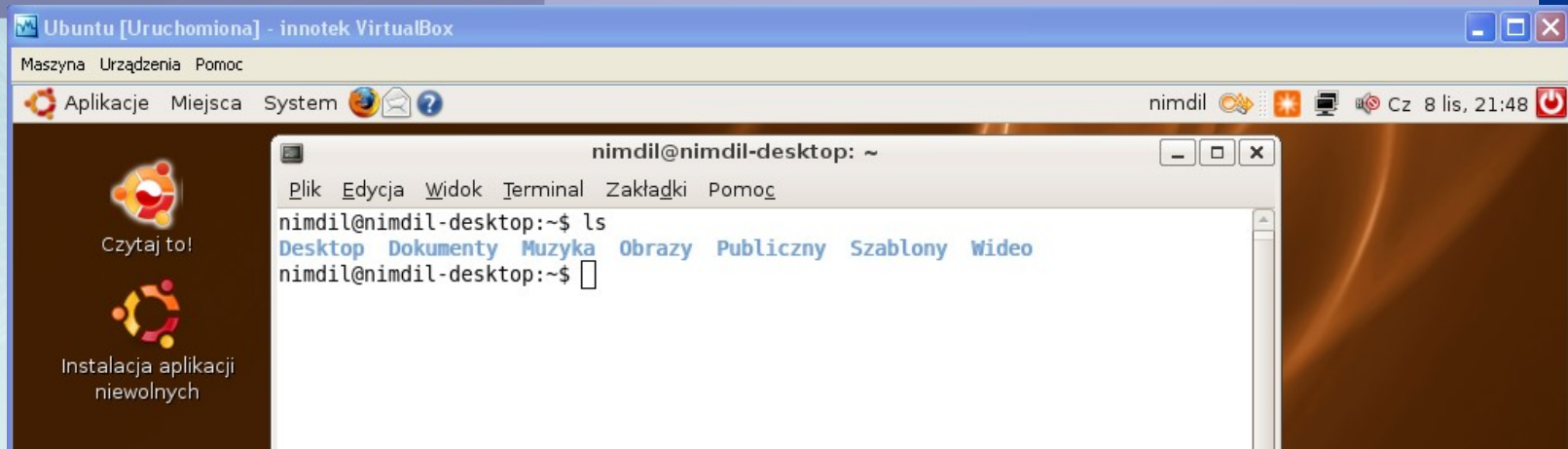
C:\Program Files\innotek VirtualBox>VBoxManage setextradata "Kubuntu" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/guestftp/Protocol" TCP
VirtualBox Command Line Management Interface Version 1.5.2
(C) 2005-2007 innotek GmbH
All rights reserved.

C:\Program Files\innotek VirtualBox>VBoxManage setextradata "Kubuntu" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/guestftp/GuestPort" 21
VirtualBox Command Line Management Interface Version 1.5.2
(C) 2005-2007 innotek GmbH
All rights reserved.

C:\Program Files\innotek VirtualBox>VBoxManage setextradata "Kubuntu" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/guestftp/HostPort" 2121
VirtualBox Command Line Management Interface Version 1.5.2
(C) 2005-2007 innotek GmbH
All rights reserved.
```

- W efekcie VirtualBox wychwyci próby łączenia z portem 2222 i przekieruje połączenie do VM. Żeby się połączyć z komputera hosta, trzeba wybrać adres localhost:2222.

# Efekt



# Połączenie sieciowe: „interfejs gospodarza”

- Realizowany poprzez dodanie nowego interfejsu sieciowego na komputerze hosta

 VirtualBox - Interfejs Gospodarza 1      Sieć LAN lub szybki Internet      Połączono      VirtualBox TAP Adapter

- Po przeładowaniu tak podłączonej VM, będzie ona miała dostęp do np. udostępnionych folderów. Można też taką wirtualną kartę podłączyć do mostka.

## Sieć wewnętrzna

- Systemowo zbliżona do poprzedniego rozwiązania, przy czym
- „widoczność” jest ograniczona do maszyn wirtualnych, ale
- łączność VM jest bezpośrednia, co owocuje poprawą prędkości i bezpieczeństwem.

## Dodatki, tzw. guest additions

- Instalowane w VM,
- poprawa pracy myszki,
- lepsza integracja zegara VM z zegarem hosta
- „Seamless windows” (tylko Windows guest)

# Chciałoby się powiedzieć...



## ...ale:

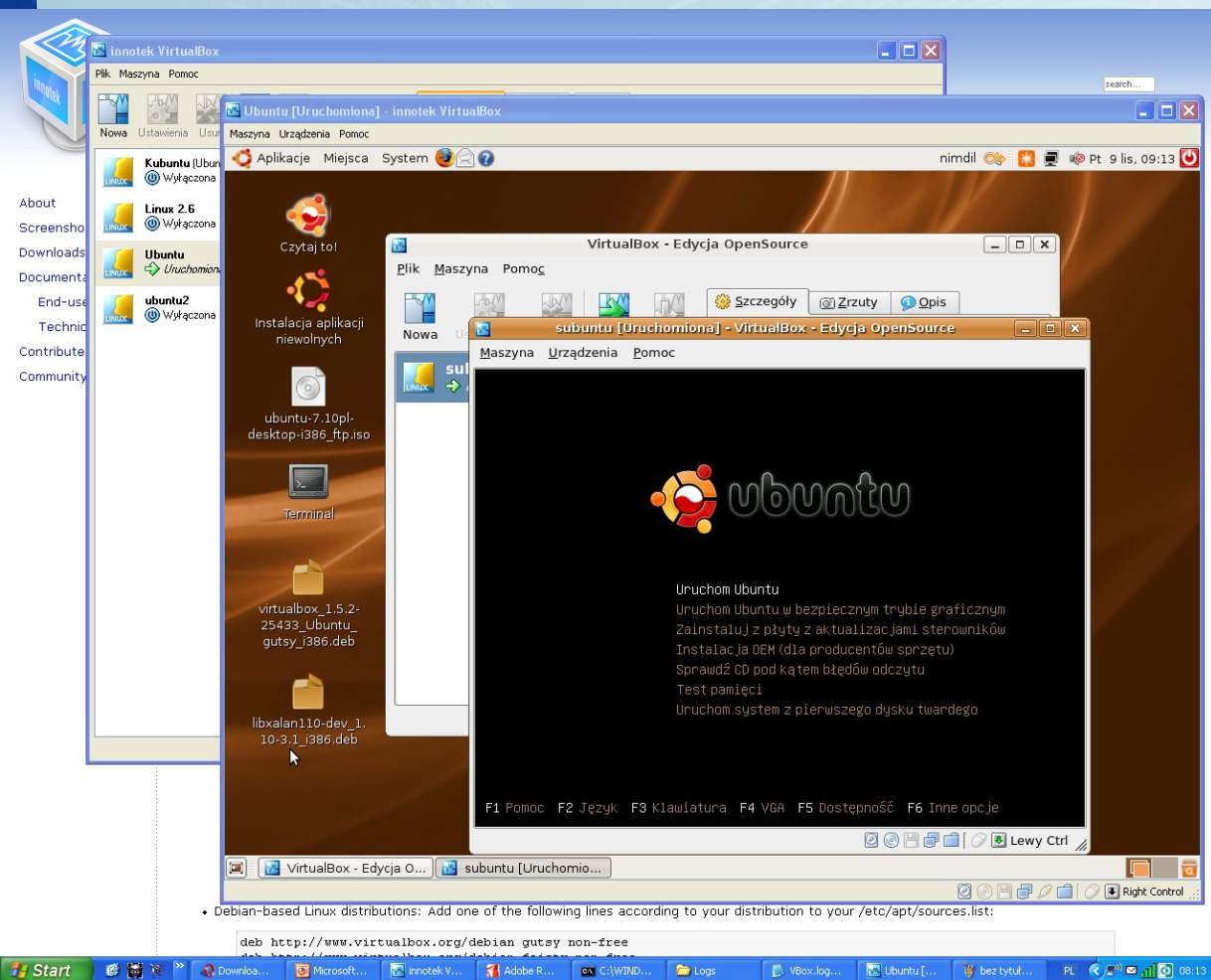
- stabilność systemu pozostawia nieco do życzenia,
- system operacyjny w VM nie zachowuje się identycznie jak w komputerze, co czasami wywołuje błędy,
- system tworzenia kopii i ładowania z nich systemu jest awaryjny,
- serwer FTP,
- systemy operacyjne po jakimś czasie bytowania w VM zachowują się „dziwnie”,
- VirtualBox on VirtualBox...

# VirtualBox on VirtualBox


- niemożność poprawnego skonfigurowania napędu optycznego w V2M,
- problemy z „VirtualLockiem” (pomijalny),
- kosmiczne problemy z poprawnym działaniem maszyny wirtualnej wewnątrz maszyny wirtualnej,
- trochę możliwa:



# VirtualBox on VirtualBox, c.d.



**VirtualBox - Guru Meditation**

 Wystąpił błąd krytyczny podczas pracy maszyny wirtualnej i została ona zatrzymana.

Pomoc możesz uzyskać korzystając z forum na stronie <http://www.virtualbox.org> (sekcja Community) lub kontaktując się bezpośrednio z nami. Pamiętaj, aby do opisu problemu dołączyć zawartość pliku z logiem `VBBox_1.log` oraz plik obrazu `VBBox.png`, które znajdziesz w katalogu `C:\Documents and Settings\JA\VirtualBox\Machines\Ubuntu\Logs`, jak również informację o tym co robiłeś bezpośrednio przed wystąpieniem błędu. Dostęp do powyższych plików możesz także uzyskać wybierając opcję **Pokaż Log** w menu **Maszyna** w oknie głównym programu VirtualBox.

Wciśnij **OK** aby wyłączyć maszynę lub **Ignoruj**, jeśli chcesz ją pozostawić nienaruszoną w celu debugowania. **Uwaga:** jeśli nie zamierzasz debugować maszyny wirtualnej, zalecane jest, abyś wcisnął **OK**.

- Ale nie działa?

## W sumie

- Na plus:

- dobrze napisany manual
- system zarówno komercyjny jak i w wersji OSE
- funkcjonalny GUI
- prostota użycia
- znane ciekawe zastosowania np. cluster MySQL

- Na minus:

- niepełny manual
- czasami dziwne zachowanie się VM
- niestabilność systemu zapisywania stanu
- „If you require more information on how to lift this restriction, please contact innotek.”
- konieczność wykonania pewnych czynności z linii poleceń

# VMware

- Oprogramowanie komercyjne opracowane przez firmę VMware, Inc. ([www.vmware.com](http://www.vmware.com)).
- Dostępne są wersje dla systemów Windows i Linux.
- Pozwala na uruchomienie wielu maszyn wirtualnych na jednym komputerze.
- Emuluje wszystkie urządzenia w ramach maszyny wirtualnej, m.in. kartę grafiki, kartę dźwiękową, kartę sieciową oraz dyski twarde.
- Udostępnia także dostęp do fizycznych urządzeń przez porty szeregowy, równoległy i USB.

## VMware – wersje produktu

- Istnieje kilka wersji produktu, m.in.:
  - VMware Workstation
    - Płatny, dostępna 30-dniowa wersja testowa.
    - Umożliwia uruchomienie wielu systemów operacyjnych na jednym hoście.
  - VMware Server
    - Darmowy (!)
    - Pracuje w architekturze klient-serwer.
    - Pozwala na zdalny dostęp do maszyny wirtualnej, za cenę zmniejszonej wydajności operacji graficznych.
  - VMware ESX Server
    - Produkt płatny, rozszerza możliwości VMware Server.
    - Opiera się na własnym jądrze opartym o jądro Linuksa.

## VMware – możliwości

- Pozwala na tworzenie obrazów stanu maszyny wirtualnej (snapshotów) i późniejszy powrót do nich.
- Uruchomiona maszyna wirtualna może być zatrzymana, przeniesiona na inny fizyczny komputer i ponownie uruchomiona, by dokończyć pracę.
  - Komputer, na który maszyna wirtualna jest przeniesiona, może dysponować innym sprzętem – inną kartą graficzną, sieciową, ilością pamięci.
  - Nie można za to przenieść maszyny wirtualnej na komputer o innej architekturze procesora.
  - Aplikacja pomocnicza Vmotion pozwala na działanie maszyny wirtualnej nawet podczas jej migracji do innego komputera.

# VMware – architektura

- Virtual Machine Monitor (VMM)
  - Działa w trybie jądra
  - Pozwala wykonywać niewrażliwe instrukcje pochodzące z maszyny wirtualnej bezpośrednio na procesorze hosta.
  - Przechwytuje wykonywane na maszynie wirtualnej instrukcje wrażliwe i dokonuje ich emulacji.
- VMX Driver
  - Działa w trybie jądra
  - Pozwala na komunikację między systemem operacyjnym hosta i systemem operacyjnym na maszynie wirtualnej.
- VMware Application
  - Działa w trybie użytkownika.
  - Pozwala na sterowanie pracą maszyn wirtualnych.

## User-Mode Linux

- Projekt powstał w 1999 roku, jego twórcą jest Jeff Dike (<http://user-mode-linux.sourceforge.net/>)
- Początkowo był dostępny jako łątka na jądro Linuksa.
- Jest dostępny w jądrze Linuksa jako jedna z architektur od wersji jądra 2.6.
- Oryginalnie został zaimplementowany na procesory rodziny x86, ale został także przeniesiony na inne architektury, m.in. IA-64 oraz PowerPC.

## User-Mode Linux – architektura

- Dla jądra hosta jest zwykłym procesem.
  - Ma postać pliku wykonywalnego ELF.
  - Nie komunikuje się ze sprzętem bezpośrednio, tylko wykorzystuje do tego celu interfejs udostępniany przez jądro hosta.
  - Nie trzeba uprawnień administratora, by z niego korzystać.
  - Można go debugować przy użyciu debuggerów działających w trybie użytkownika.
- Dla uruchomionych w nim procesów jest jądrem.
  - Ma własną przestrzeń adresową, dzięki czemu jest chronione przez MMU (Memory Management Unit) procesora hosta.
  - Można uruchomić UML pod UML.

## User-Mode Linux – instalacja

- Potrzebne nam będą:
  - Źródła jądra Linuksa (na *students* są w katalogu `/usr/local/src/linux-2.6.17.13`).
  - W zależności od wersji jądra – mogą być potrzebne łatki na jądro, aby móc skompilować UML.
  - Obraz systemu plików z wybraną dystrybucją Linuksa (np. `http://uml.nagafix.co.uk/Debian-4.0/Debian-4.0-x86-root\_fs.bz2`).

# User-Mode Linux – instalacja

```
aj219452@students: /home/jsim/aj219452
Plik Edycja Widok Terminal Zakładki Pomoc
Password:
Last login: Thu Nov  8 15:37:18 2007 from green15.mimuw.edu.pl

W razie problemow prosze pisac na root@students.mimuw.edu.pl

NIE zmieniam hasel przez poczte!!!

Administrator

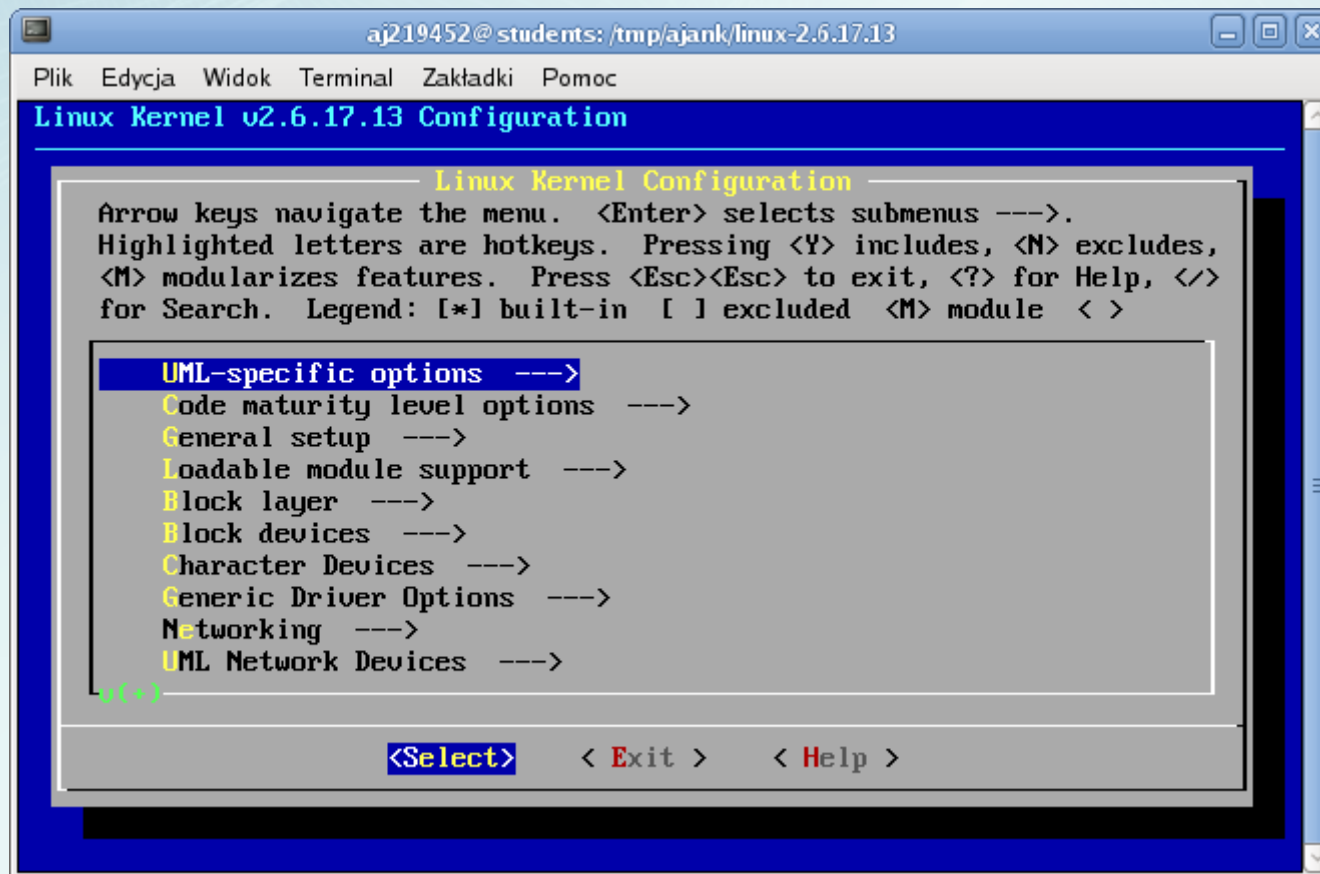
You have old mail in folder /home/jsim/
Disk quotas for user aj219452 (uid 2121):
  Filesystem blocks  quota  limit
  /dev/hdv2  213212  300000  300000
[aj219452@students ~]$
```

```
aj219452@students: /tmp/ajank/linux-2.6.17.13
Plik Edycja Widok Terminal Zakładki Pomoc

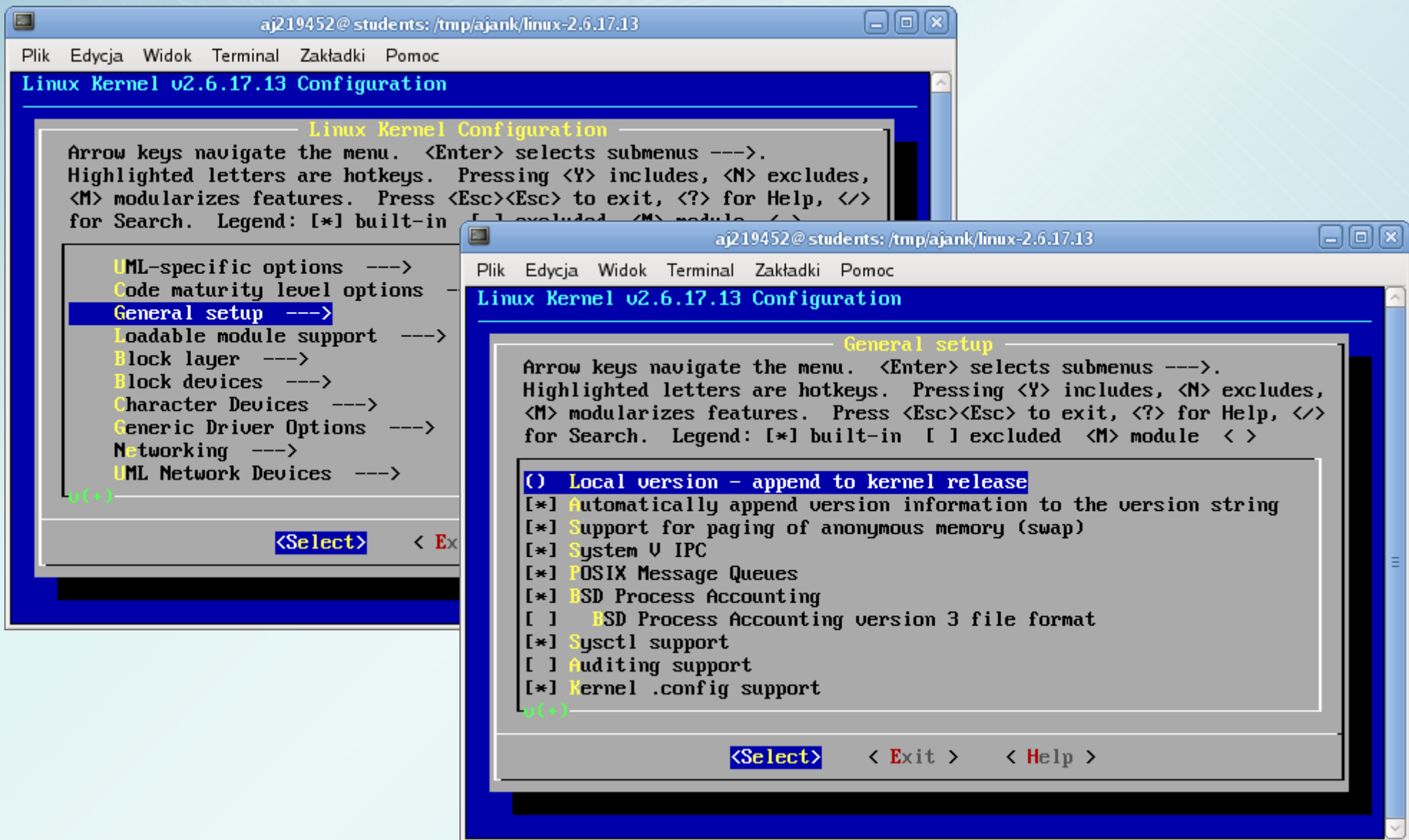
You have old mail in folder /home/jsim/a/aj219452/Mail/Maildir.
Disk quotas for user aj219452 (uid 21213):
  Filesystem blocks  quota  limit  grace  files  quota  limit  grace
  /dev/hdv2  213212  300000  300000          5254  30000  30000
[aj219452@students ~]$ cp -R /usr/local/src/linux-2.6.17.13/ /tmp/ajank
[aj219452@students ~]$ cd /tmp/ajank/linux-2.6.17.13/
[aj219452@students /tmp/ajank/linux-2.6.17.13]$ patch -p1 < ~/so/linux-2.6.17.13
-UML.patch
patching file arch/um/Makefile
patching file arch/um/include/linux/autoconf.h
patching file arch/um/include/linux/config.h
patching file arch/um/include/linux/threads.h
patching file arch/um/os-Linux/aio.c
patching file arch/um/os-Linux/process.c
patching file arch/um/os-Linux/sys-i386/registers.c
patching file arch/um/os-Linux/sys-i386/tls.c
patching file arch/um/os-Linux/sys-x86_64/registers.c
patching file arch/um/os-Linux/tls.c
patching file arch/um/sys-i386/unmap.c
patching file arch/um/sys-i386/user-offsets.c
patching file arch/um/sys-x86_64/unmap.c
[aj219452@students /tmp/ajank/linux-2.6.17.13]$
```

# User-Mode Linux – instalacja

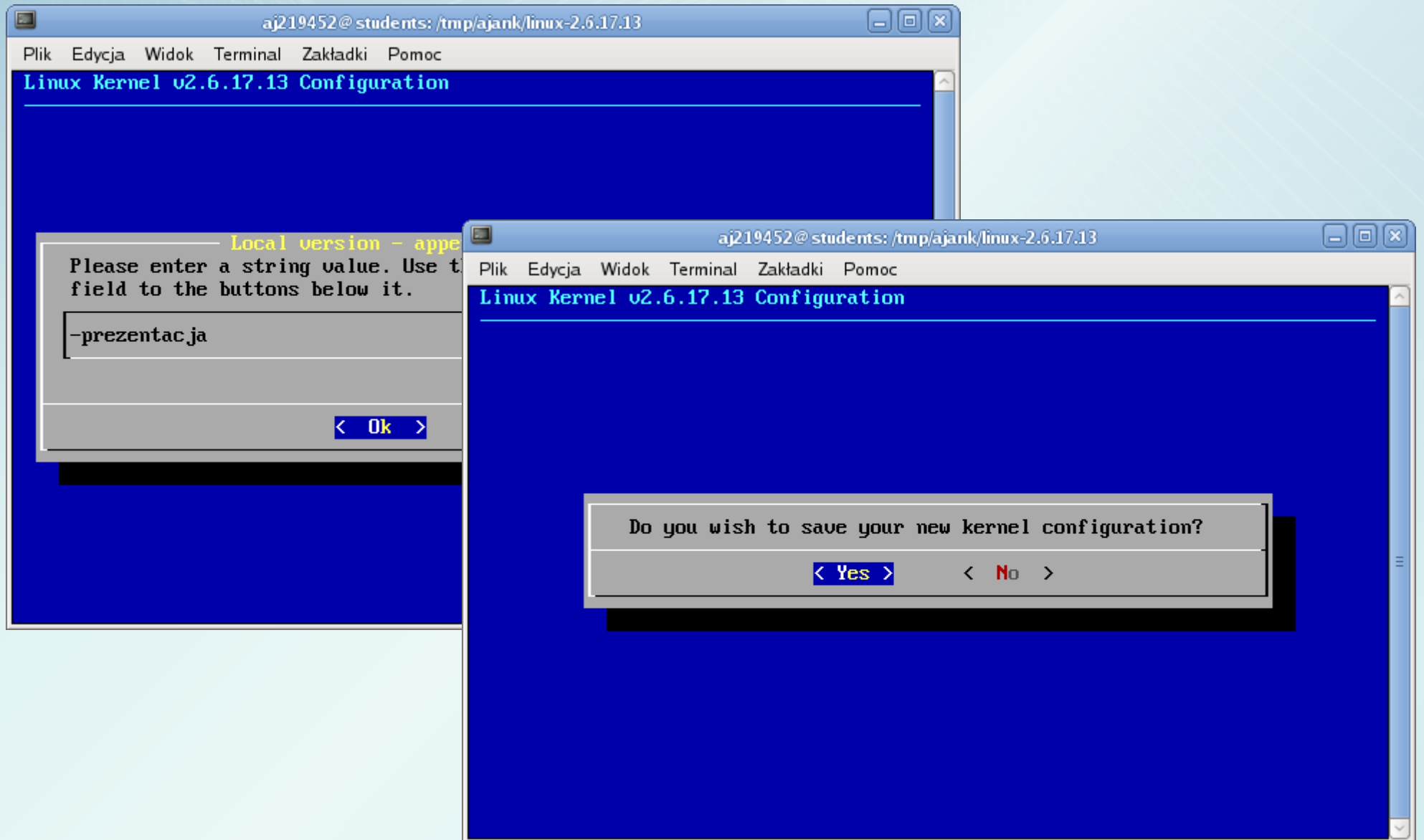
- Konfigurujemy jądro poleceniem  
`make menuconfig ARCH=um`



# User-Mode Linux – instalacja

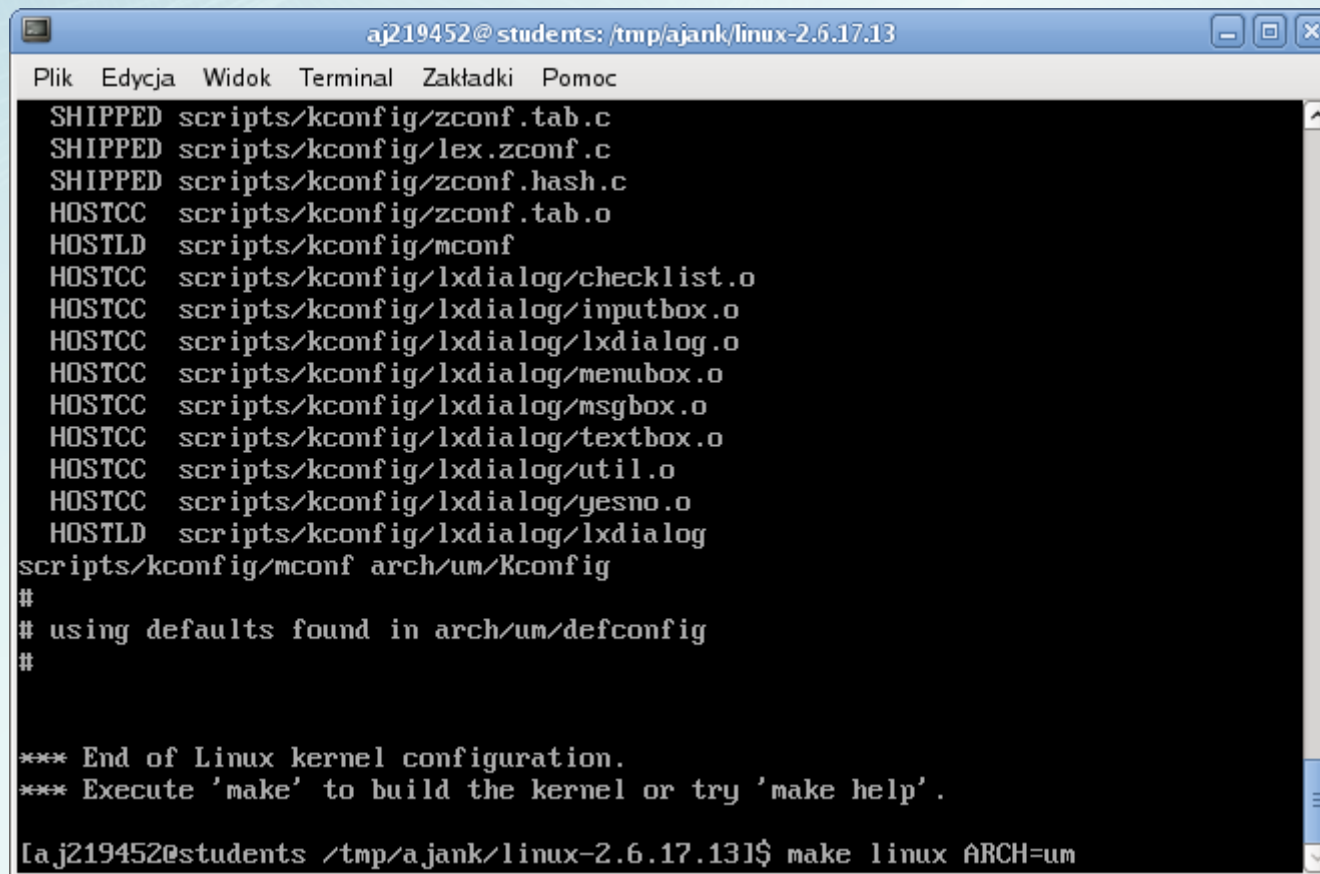


# User-Mode Linux – instalacja



# User-Mode Linux – instalacja

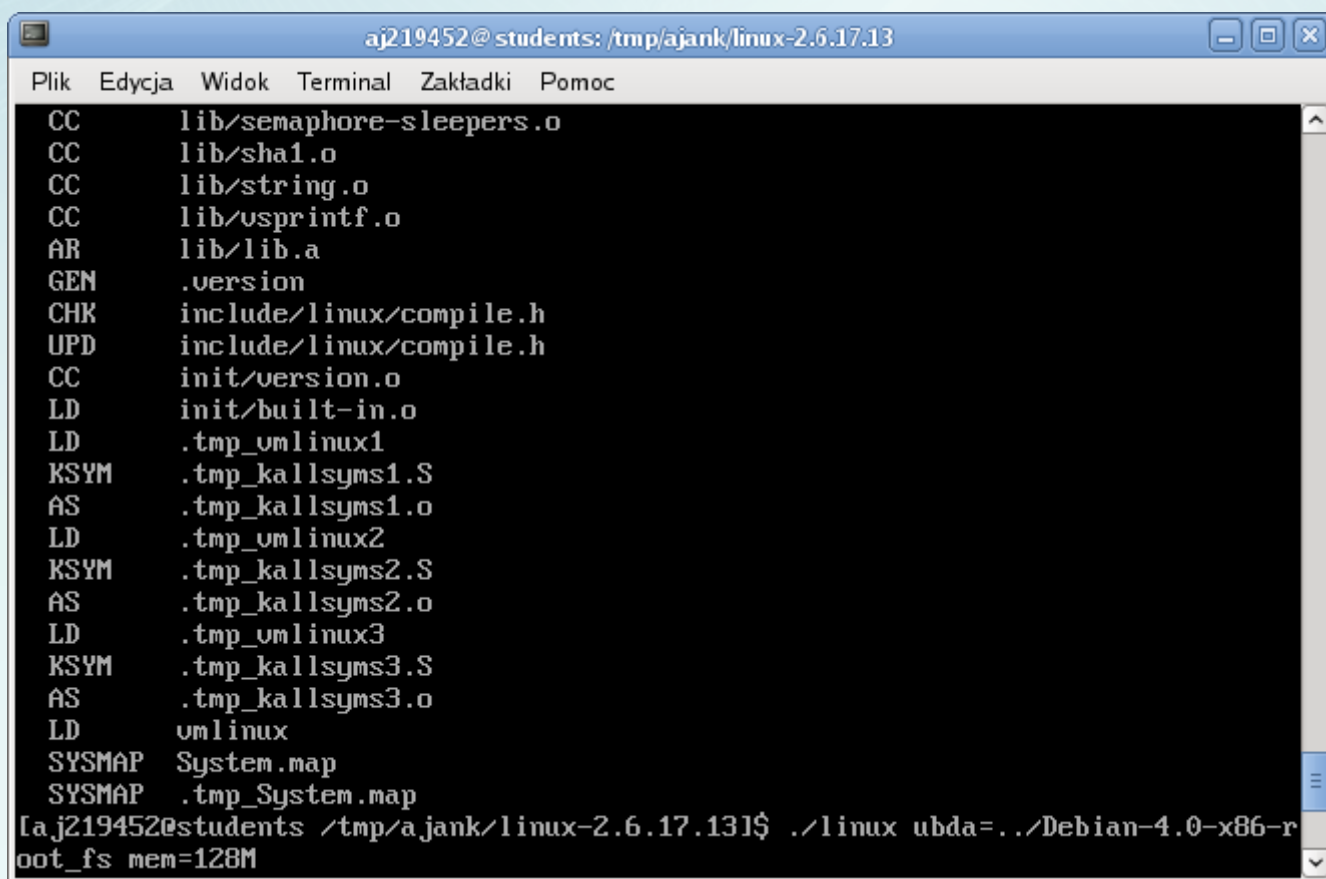
- Kompilujemy jądro poleceniem  
`make linux ARCH=um`



```
aj219452@students: /tmp/ajank/linux-2.6.17.13
Plik  Edycja  Widok  Terminal  Zakładki  Pomoc
SHIPPED scripts/kconfig/zconf.tab.c
SHIPPED scripts/kconfig/lex.zconf.c
SHIPPED scripts/kconfig/zconf.hash.c
HOSTCC  scripts/kconfig/zconf.tab.o
HOSTLD  scripts/kconfig/mconf
HOSTCC  scripts/kconfig/lxdialog/checklist.o
HOSTCC  scripts/kconfig/lxdialog/inputbox.o
HOSTCC  scripts/kconfig/lxdialog/lxdialog.o
HOSTCC  scripts/kconfig/lxdialog/menubox.o
HOSTCC  scripts/kconfig/lxdialog/msgbox.o
HOSTCC  scripts/kconfig/lxdialog/textbox.o
HOSTCC  scripts/kconfig/lxdialog/util.o
HOSTCC  scripts/kconfig/lxdialog/yesno.o
HOSTLD  scripts/kconfig/lxdialog/lxdialog
scripts/kconfig/mconf arch/um/Kconfig
#
# using defaults found in arch/um/defconfig
#
*** End of Linux kernel configuration.
*** Execute 'make' to build the kernel or try 'make help'.
[aj219452@students /tmp/ajank/linux-2.6.17.13]$ make linux ARCH=um
```

# User-Mode Linux – instalacja

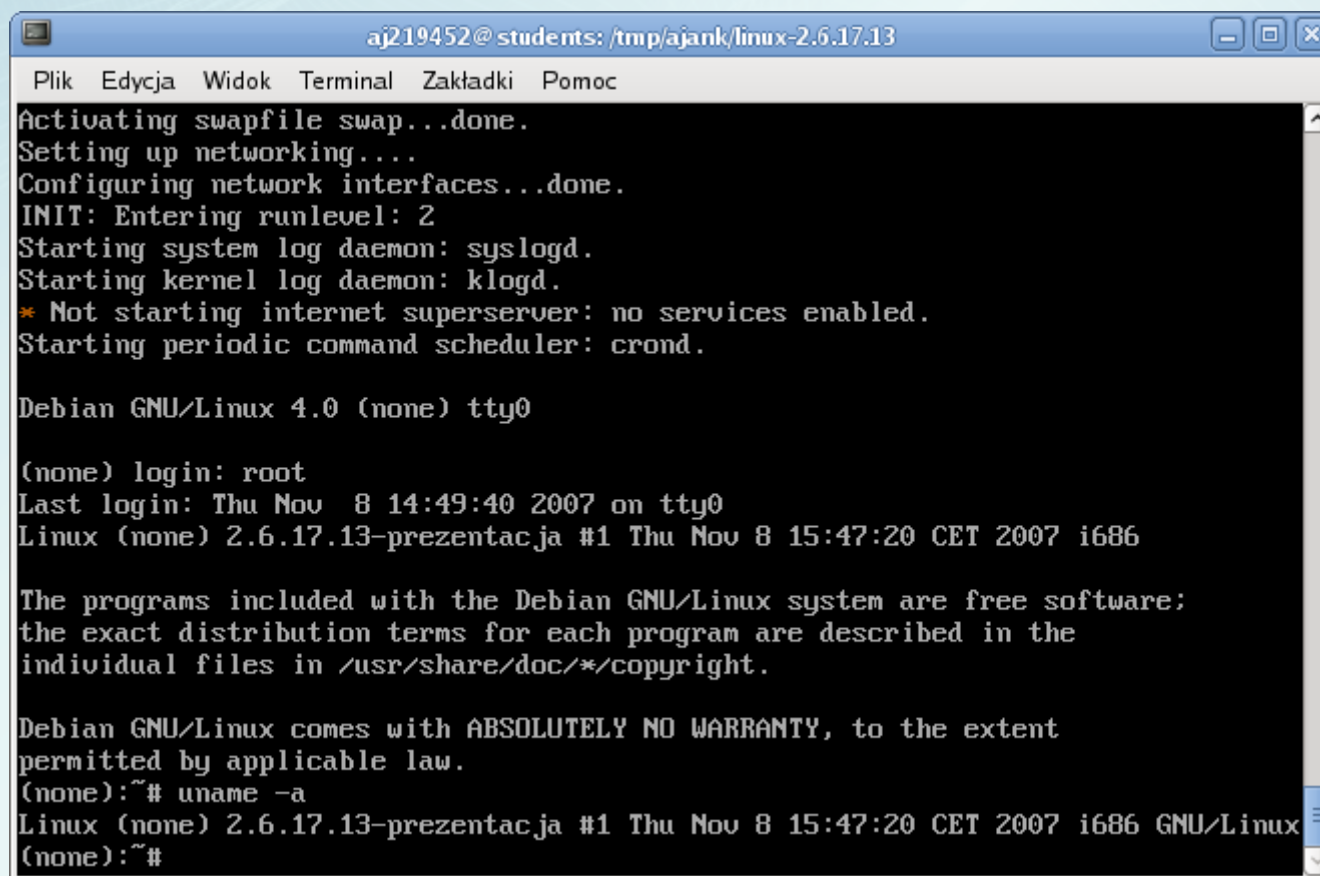
- Uruchamiamy skompilowane jądro poleceniem  
`./linux ubda=../Debian-4.0-x86-root_fs  
mem=128M`



```
aj219452@students: /tmp/ajank/linux-2.6.17.13
Plik  Edycja  Widok  Terminal  Zakładki  Pomoc
CC    lib/semaphore-sleepers.o
CC    lib/sha1.o
CC    lib/string.o
CC    lib/usprintf.o
AR    lib/lib.a
GEN   .version
CHK   include/linux/compile.h
UPD   include/linux/compile.h
CC    init/version.o
LD    init/built-in.o
LD    .tmp_umlinux1
KSYM  .tmp_kallsyms1.S
AS    .tmp_kallsyms1.o
LD    .tmp_umlinux2
KSYM  .tmp_kallsyms2.S
AS    .tmp_kallsyms2.o
LD    .tmp_umlinux3
KSYM  .tmp_kallsyms3.S
AS    .tmp_kallsyms3.o
LD    umlinux
SYMAP System.map
SYMAP .tmp_System.map
[aj219452@students /tmp/ajank/linux-2.6.17.13]$ ./linux ubda=../Debian-4.0-x86-r
oot_fs mem=128M
```

# User-Mode Linux – instalacja

- Cieszymy się działającym systemem. :)



```
aj219452@students: /tmp/ajank/linux-2.6.17.13
Plik  Edycja  Widok  Terminal  Zakładki  Pomoc
Activating swapfile swap...done.
Setting up networking....
Configuring network interfaces...done.
INIT: Entering runlevel: 2
Starting system log daemon: syslogd.
Starting kernel log daemon: klogd.
* Not starting internet superserver: no services enabled.
Starting periodic command scheduler: crond.

Debian GNU/Linux 4.0 (none) tty0

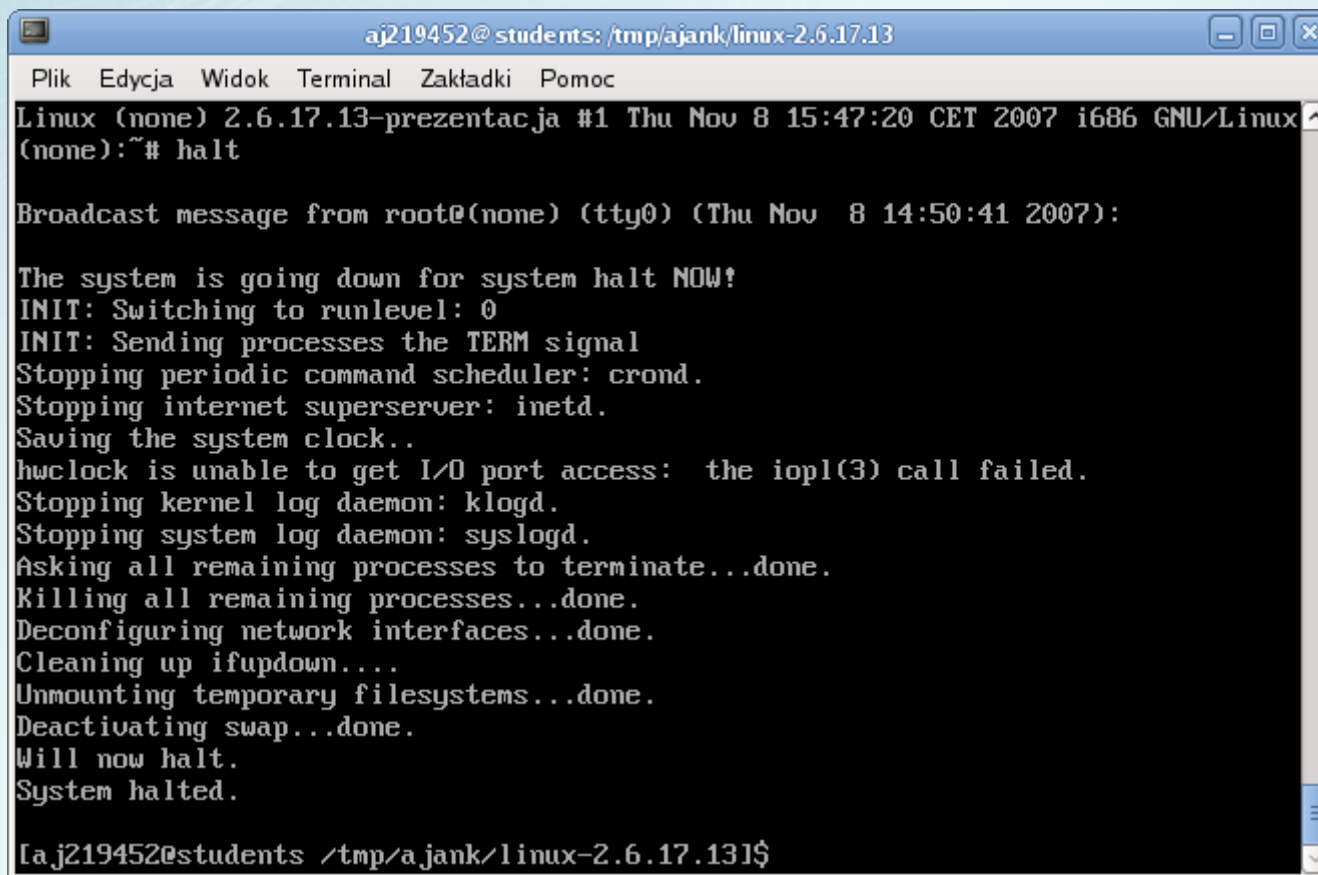
(none) login: root
Last login: Thu Nov  8 14:49:40 2007 on tty0
Linux (none) 2.6.17.13-prezentacja #1 Thu Nov  8 15:47:20 CET 2007 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
(none):~# uname -a
Linux (none) 2.6.17.13-prezentacja #1 Thu Nov  8 15:47:20 CET 2007 i686 GNU/Linux
(none):~#
```

# User-Mode Linux – instalacja

- Gdy już skończymy pracę, możemy wrócić do powłoki hosta.



```
aj219452@students: /tmp/ajank/linux-2.6.17.13
Plik Edycja Widok Terminal Zakładki Pomoc
Linux (none) 2.6.17.13-prezentacja #1 Thu Nov 8 15:47:20 CET 2007 i686 GNU/Linux
(none):~# halt

Broadcast message from root@(none) (tty0) (Thu Nov  8 14:50:41 2007):

The system is going down for system halt NOW!
INIT: Switching to runlevel: 0
INIT: Sending processes the TERM signal
Stopping periodic command scheduler: crond.
Stopping internet superserver: inetd.
Saving the system clock..
hwclock is unable to get I/O port access:  the iopl(3) call failed.
Stopping kernel log daemon: klogd.
Stopping system log daemon: syslogd.
Asking all remaining processes to terminate...done.
Killing all remaining processes...done.
Deconfiguring network interfaces...done.
Cleaning up ifupdown...
Unmounting temporary filesystems...done.
Deactivating swap...done.
Will now halt.
System halted.

[aj219452@students /tmp/ajank/linux-2.6.17.13]$
```