Fedora 12

Руководство по виртуализации

Справочное руководство по виртуализации в Fedora

Кристофер Каррен

Fedora 12 Руководство по виртуализации
Справочное руководство по виртуализации в Fedora
Редакция 1

Автор Кристофер Каррен ccurran@redhat.com

Руководство по виртуализации Fedora 12 предоставляет информацию по установке, настройке, администрированию, диагностике и решению проблем технологий виртуализации в Fedora 12.

Введение

Руководство по виртуализации Fedora 12 описывает все аспекты использования и управления компонентами виртуализации, входящих в состав Fedora 12.

1. Об этой книге

Книга состоит из 7 частей:

- Системные требования
- Installation
- Configuration
- Administration
- Предметный указатель
- Tips and Tricks
- Troubleshooting

2. Соглашения документа

В этом руководстве используются различные стили для выделения текста.

В PDF- и бумажной версиях руководства используются шрифты семейства *Liberation*[1]. Эти же шрифты будут использоваться для отображения HTML-версии, если они установлены в вашей системе. Замечание: Red Hat Enterprise Linux 5 и более поздние версии включают в свой состав комплект Liberation по умолчанию.

2.1. Типографические соглашения

Для выделения текста используются четыре стиля, которые будут перечислены далее.

`Моноширинный жирный шрифт`

Используется для выделения системного ввода, включая команды оболочки, а также имена файлов, пути и комбинации ключей. Пример:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key cap, all presented in Mono-spaced Bold and all distinguishable thanks to context.

Key-combinations can be distinguished from key caps by the hyphen connecting each part of a key-combination. For example:

> Press **`Enter`** to execute the command.

[1] https://fedorahosted.org/liberation-fonts/

> Press **`Ctrl`**+**`Alt`**+**`F1`** to switch to the first virtual terminal. Press **`Ctrl`**+**`Alt`**+**`F7`** to return to your X-Windows session.

The first sentence highlights the particular key cap to press. The second highlights two sets of three key caps, each set pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`Mono-spaced Bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialogue box text; labelled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System > Preferences > Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).

> To insert a special character into a **gedit** file, choose **Applications > Accessories > Character Map** from the main menu bar. Next, choose **Search > Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit > Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in Proportional Bold and all distinguishable by context.

Note the **>** shorthand used to indicate traversal through a menu and its sub-menus. This is to avoid the difficult-to-follow 'Select **Mouse** from the **Preferences** sub-menu in the **System** menu of the main menu bar' approach.

`Mono-spaced Bold Italic` or ***`Proportional Bold Italic`***

Whether Mono-spaced Bold or Proportional Bold, the addition of Italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **`ssh`** ***`username@domain.name`*** at a shell prompt. If the remote machine is **`example.com`** and your username on that machine is john, type **`ssh john@example.com`**.

> The **`mount -o remount`** ***`file-system`*** command remounts the named file system. For example, to remount the **`/home`** file system, the command is **`mount -o remount /home`**.

> To see the version of a currently installed package, use the **`rpm -q`** ***`package`*** command. It will return a result as follows: ***`package-version-release`***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server.

2.2. Соглашения по выделению текста

Данные разных типов, содержащие несколько строк, будут отделены от окружающего текста.

Текст, отображаемый на экране, будет отображаться моноширинным шрифтом:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Для отображения содержимого исходного кода используется моноширинный шрифт:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }

}
```

2.3. Замечания и предупреждения

Наконец, чтобы привлечь внимание читателя к важной информации, используются три стиля.

Замечание

Замечание обычно содержит дополнительную информацию. Если вы его проигнорируете, это не критично, но вы можете пропустить совет, который, возможно, поможет сэкономить время при выполнении заданий в дальнейшем.

Важно

На информацию, отмеченную как важную, следует обратить особое внимание. Она может включать изменения настроек, которые применимы к текущей сессии, или, например, перечень служб, которые нужно запустить, прежде чем обновления вступят в силу. Ознакомление с важной информацией значительно облегчит вашу работу.

Предупреждение

Не стоит игнорировать предупреждения, так как они содержат важную информацию, которая позволит избежать потери данных.

3. Нам нужны Ваши отзывы!

Если Вы нашли опечатку в этом руководстве, или у Вас есть идеи, как его улучшить, мы будем рады выслушать Вас! Пожалуйста, оставьте сообщение в Bugzilla по адресу *http://bugzilla.redhat.com/bugzilla/*, выбрав продукт **Fedora Documentation.**

Убедитесь, что в сообщении об ошибке Вы указали идентификатор данного руководства: *Virtualization_Guide*.

Если Вы хотите предложить улучшения руководства, постарайтесь описать предложения максимально подробно. Если Вы нашли ошибку, пожалуйста, укажите номер раздела и текст вокруг, чтобы мы могли её быстро найти.

Часть I. Installation

Установка виртуализации

В последующих главах будет рассмотрено, как настроить размещающую систему и установить в ней виртуальные машины с Fedora. Рекомендуется внимательно ознакомиться с приведенным материалом, так как полученные знания позволят выполнить успешную установку.

Установка пакетов виртуализации

1.1. Установка KVM в процессе установки Fedora

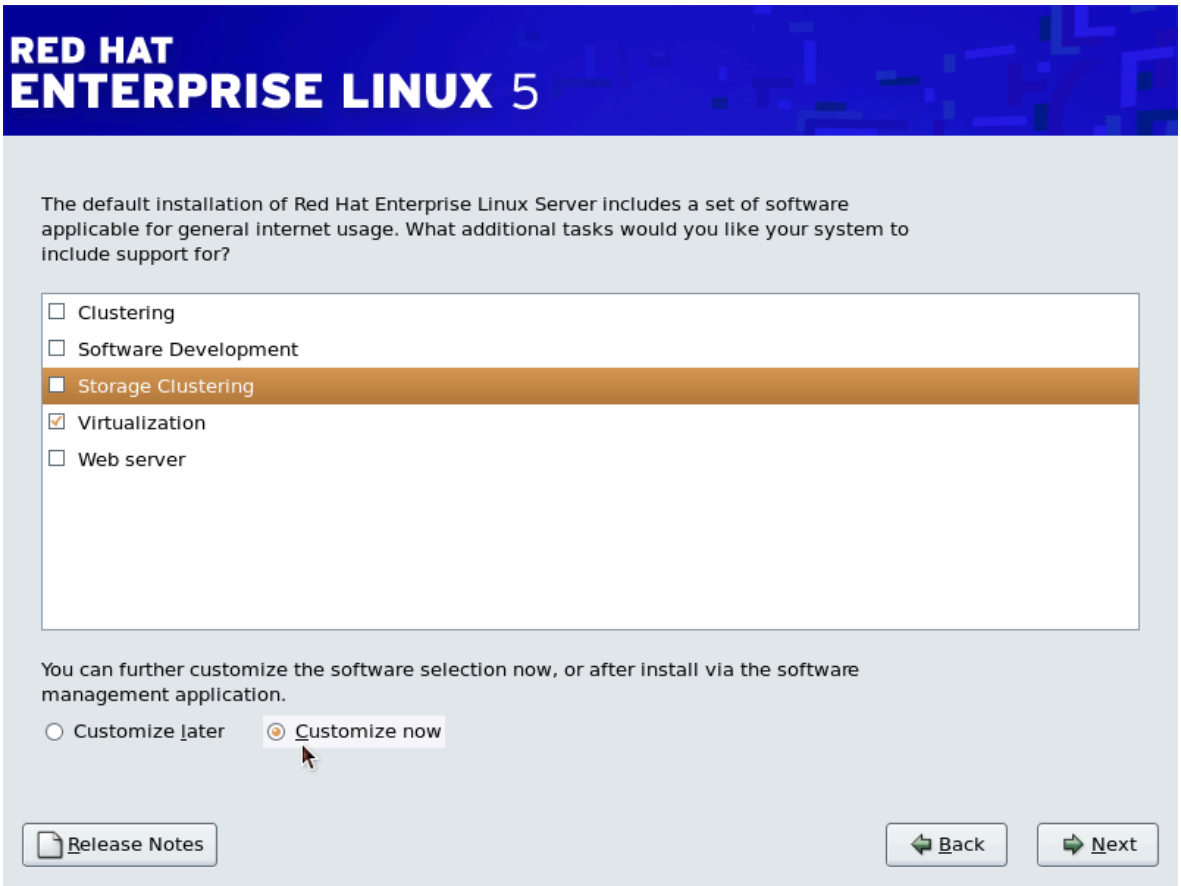
В этой секции будет рассказано о том, как установить утилиты виртуализации и пакеты Xen в процессе установки новой системы Fedora 12.

Нужна помощь в установке?

Руководство по установке на сайте *http://docs.fedoraproject.org* подробно рассматривает процесс установки Fedora 12.

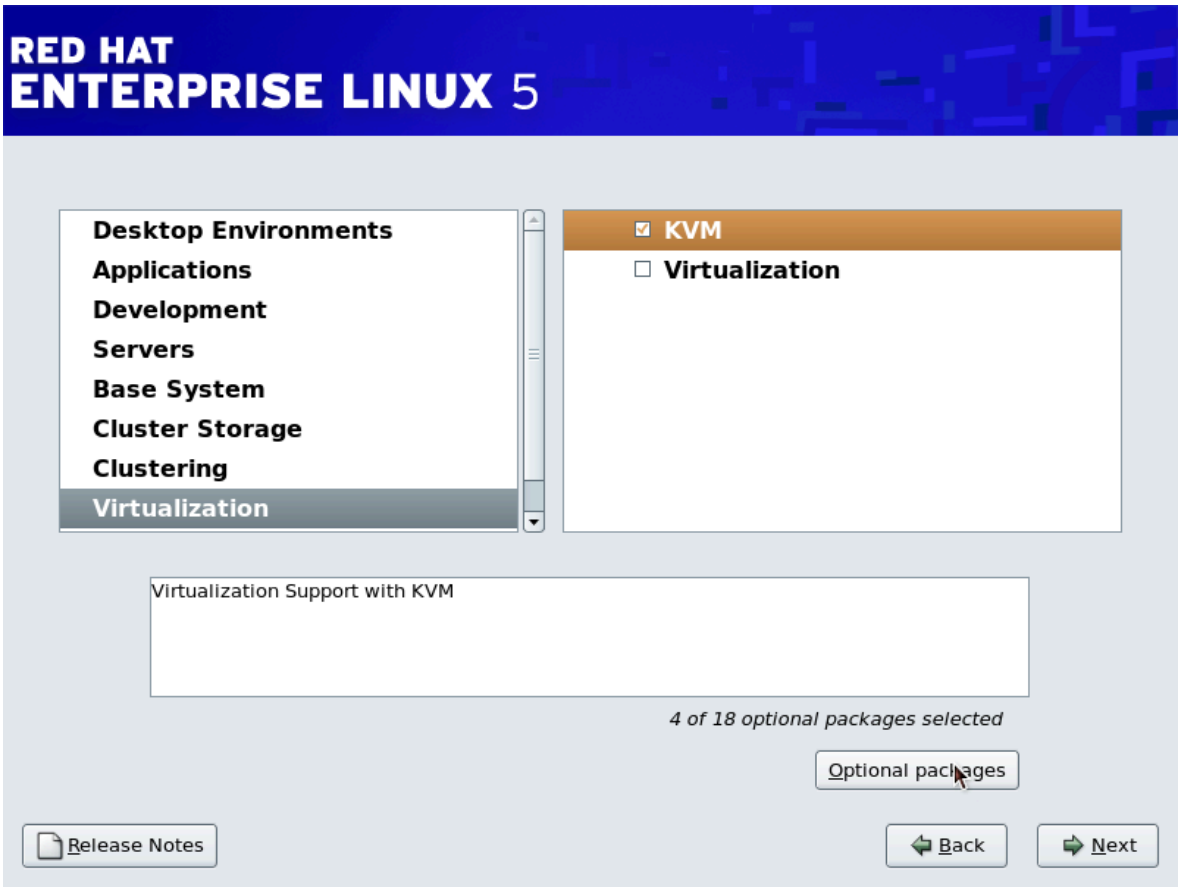
1. Запустите интерактивную установку Fedora 12 с установочного носителя (CD-ROM, DVD, PXE).

2. Продолжите установку как обычно до этапа выбора пакетов.



   Выберите группу **Виртуализация** и выберите **Настроить сейчас**.

3. Снимите флажок напротив группы **Виртуализация** и отметьте **KVM** для выбора гипервизора KVM, **`virt-manager`**, **`libvirt`** и **`virt-viewer`**.



4. **При необходимости измените список пакетов.**
   Можно изменить список пакетов в группе **Virtualization**.

Нажмите **Закрыть** и **Далее**.

Автоматизация установки пакетов KVM

Здесь рассмотрен процесс установки Fedora с пакетами KVM с помощью файла кикстарта, что позволяет выполнить одновременную установку для большого числа систем.

В секцию `%packages` добавьте следующее определение:

```
%packages
@kvm
```

Руководство по установке Fedora 12 на сайте *http://docs.fedoraproject.org* содержит подробную информацию об автоматизации процесса установки.

1.2. Установка пакетов KVM в существующей системе Fedora

Далее будет рассмотрен процесс установки гипервизора KVM в рабочей системе Fedora, начиная с версии 12.

Установка гипервизора KVM с помощью yum

Чтобы использовать возможности виртуализации в Fedora, необходимо установить пакет **`kvm`**, который содержит модуль ядра для гипервизора KVM.

Команда установки **kvm**:

```
# yum install kvm
```

Теперь установите дополнительные пакеты виртуализации.

Рекомендуемые пакеты виртуализации

python-virtinst
: Позволяет использовать **virt-install** для создания виртуальных машин.

libvirt
: Библиотека **libvirt** использует инфраструктуру виртуализации **xm** и текстовую утилиту **virsh** для управления виртуальными машинами.

libvirt-python
: Содержит модуль, который позволяет написанным на Python приложениям использовать предоставляемый библиотекой **libvirt** интерфейс.

virt-manager
: Менеджер виртуальных машин представляет собой графическую утилиту для управления виртуальными машинами и использует библиотеку **libvirt**.

Команда установки пакетов:

```
# yum install virt-manager libvirt libvirt-python python-virtinst
```

Обзор установки гостевых виртуальных машин

После завершения установки пакетов виртуализации в размещающей системе можно приступить к созданию виртуальных машин. В этой главе будут описаны процессы установки гостевых операционных систем на виртуальных машинах. Это можно сделать, нажав кнопку **Создать** (New) в окне менеджера виртуальных машин (**virt-manager**) или с помощью команды **`virt-install`**. Оба способа будут рассмотрены ниже.

Глава 3, Установка гостевой операционной системы содержит подробные инструкции по установке отдельных версий Fedora, а также других дистрибутивов Linux, Solaris и Windows.

2.1. Создание виртуальных машин с помощью virt-install

Виртуальную машину можно создать с помощью команды **`virt-install`**, которую можно либо запустить напрямую, либо включить в сценарий для автоматизации процесса создания виртуальных машин.

`virt-install` принимает параметры. Полный список можно просмотреть, выполнив команду

```
$ virt-install --help
```

На странице помощи **`virt-install`** также перечислены наиболее важные значения параметров.

Прежде чем вызвать команду **`virt-install`**, можно выполнить **`qemu-img`** для настройки хранилища.

Параметр `--vnc` позволяет открыть графическое окно установки виртуальной машины.

В этом примере гостевая система Red Hat Enterprise Linux 3 с именем *`rhel3support`* будет создана с CD-ROM, файловым образом блочного устройства размером 5 гигабайт и виртуальным сетевым окружением. Пример использует гипервизор KVM.

```
# virt-install --accelerate --hvm --connect qemu:///system \
        --network network:default \
        --name rhel3support --ram=756\
        --file=/var/lib/libvirt/images/rhel3support.img \
        --file-size=6 --vnc --cdrom=/dev/sr0
```

Пример 2.1. Создание гостя Red Hat Enterprise Linux 3 с помощью virt-install и KVM

```
# virt-install --name Fedora11 --ram 512 --file=/var/lib/libvirt/images/
Fedora11.img \
        --file-size=3 --vnc --cdrom=/var/lib/libvirt/images/Fedora11.iso
```

Пример 2.2. Создание гостя Fedora 11 с помощью virt-install

2.2. Создание виртуальных машин с помощью virt-manager

Менеджер виртуальных машин представляет собой приложение (virt-manager), с помощью которого можно ими управлять.

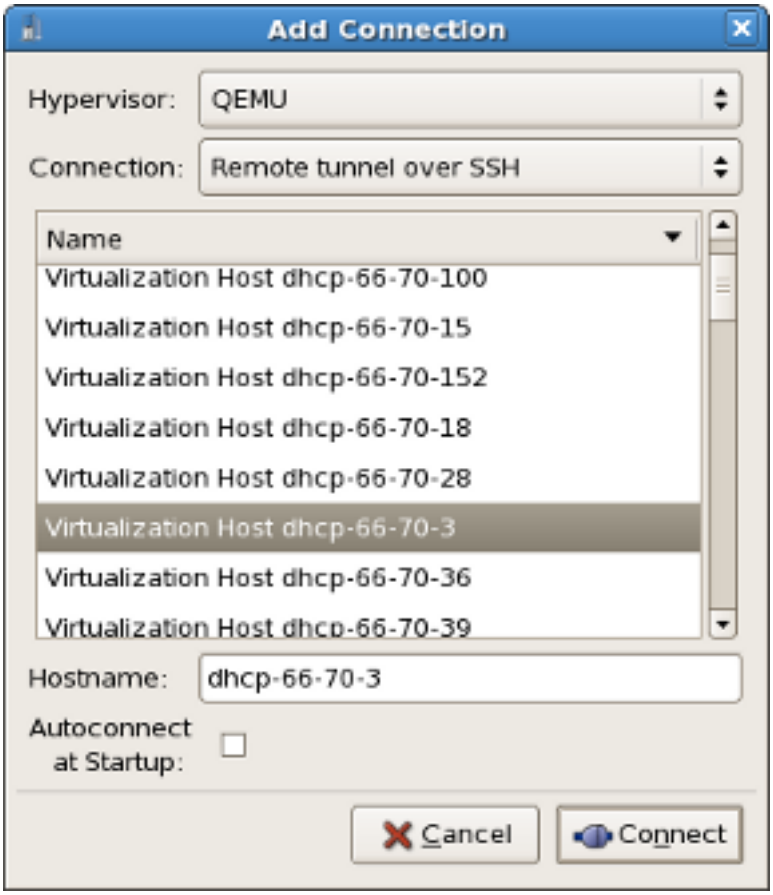
Процедура 2.1. Создание виртуальной машины с помощью **virt-manager**

1. Для запуска **virt-manager** в командной строке от лица root выполните

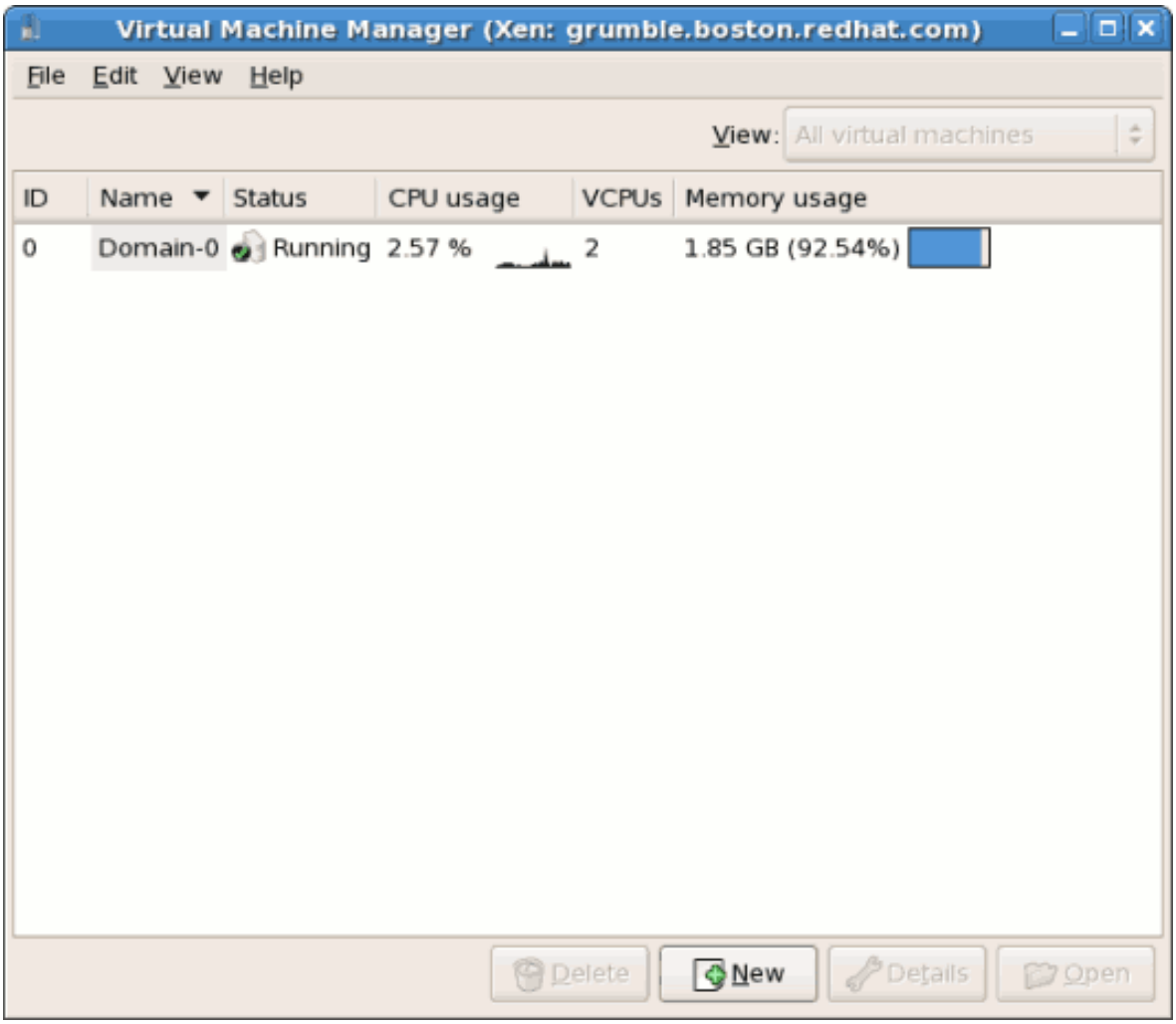
   ```
   # virt-manager &
   ```

   Откроется новое окно графического интерфейса **virt-manager**. Если у вас нет прав доступа root, кнопка создания новой виртуальной машины будет недоступна.

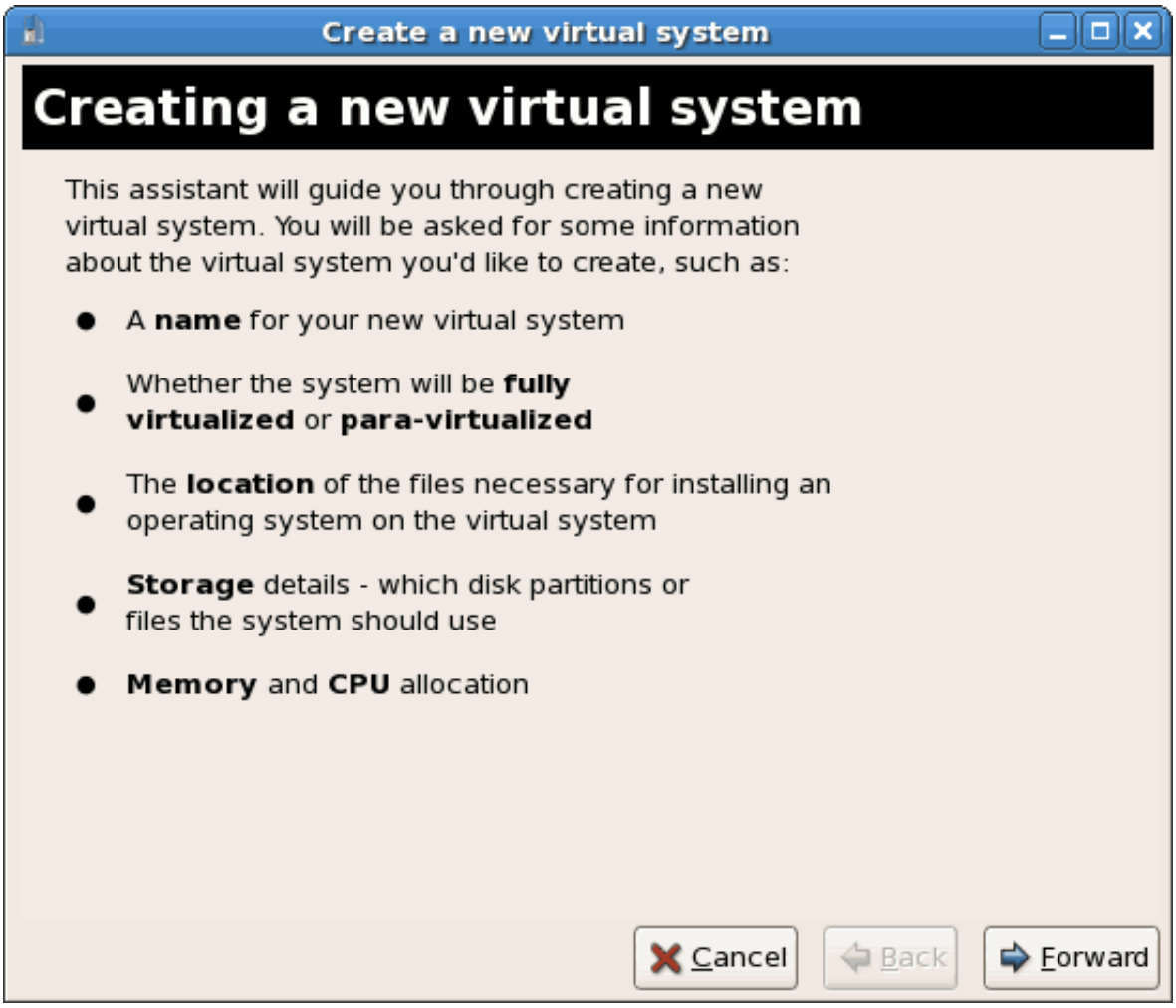
2. В меню **Файл** (File) -> выберите **Открыть соединение** (Open Connection). В появившемся окне выберите гипервизор и нажмите кнопку подключения.



3. Новую виртуальную машину можно создать, нажав кнопку **Создать** (New) в главном окне **virt-manager**.



4. В открывшемся окне будет показана сводка данных, которые необходимо предоставить для создания виртуальной машины.

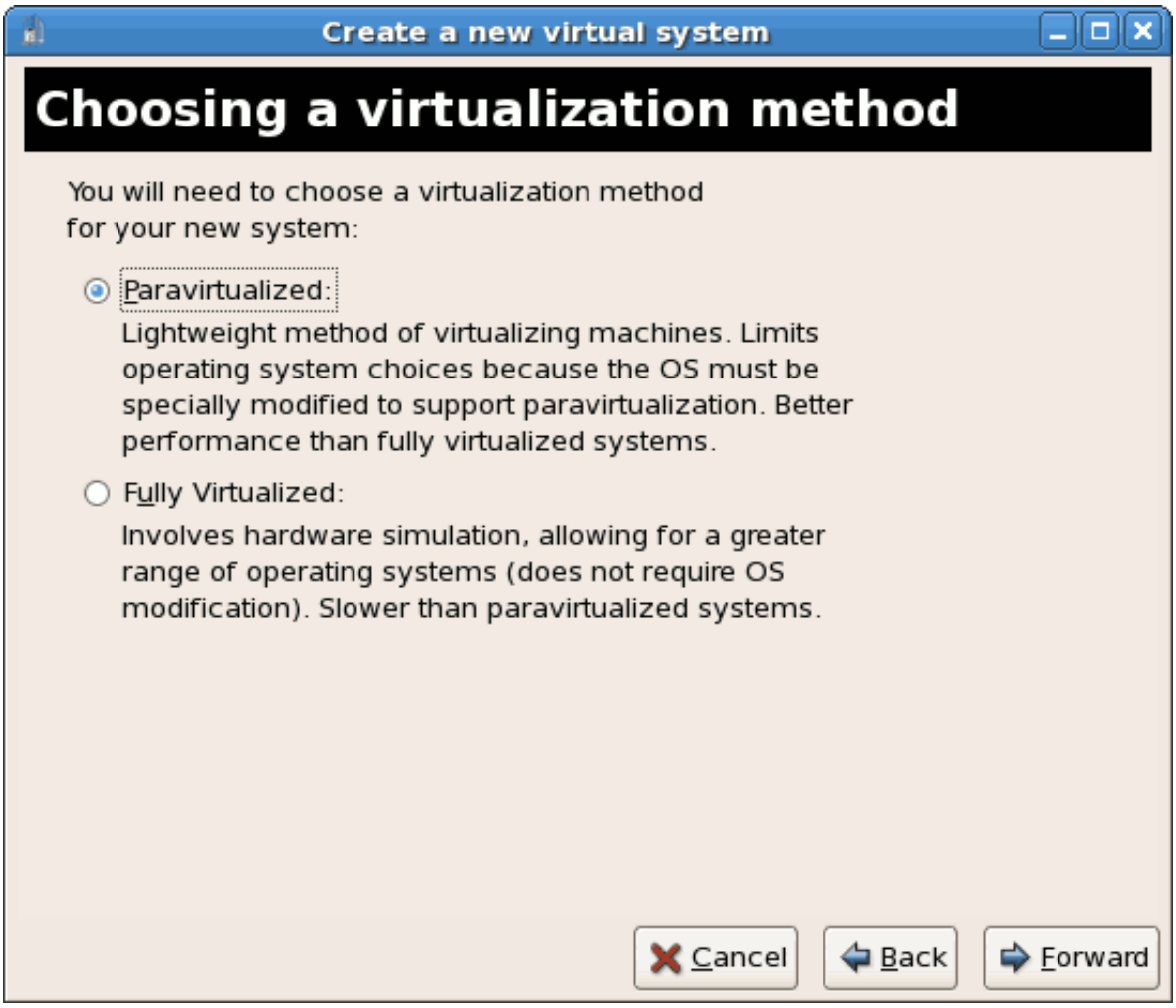


Просмотрите информацию и нажмите кнопку продолжения.

5. Выберите тип виртуализации (полная или паравиртуализация).

   Полная виртуализация требует наличия процессора Intel® VT или AMD-V. Если расширений виртуализации нет, то выбор полностью виртуализированной системы будет недоступен. Если же в этот момент не выполняется ядро **`kernel-xen`**, то будет недоступен выбор паравиртуализации.

   При подключении к гипервизору KVM будет доступна только полная виртуализация.



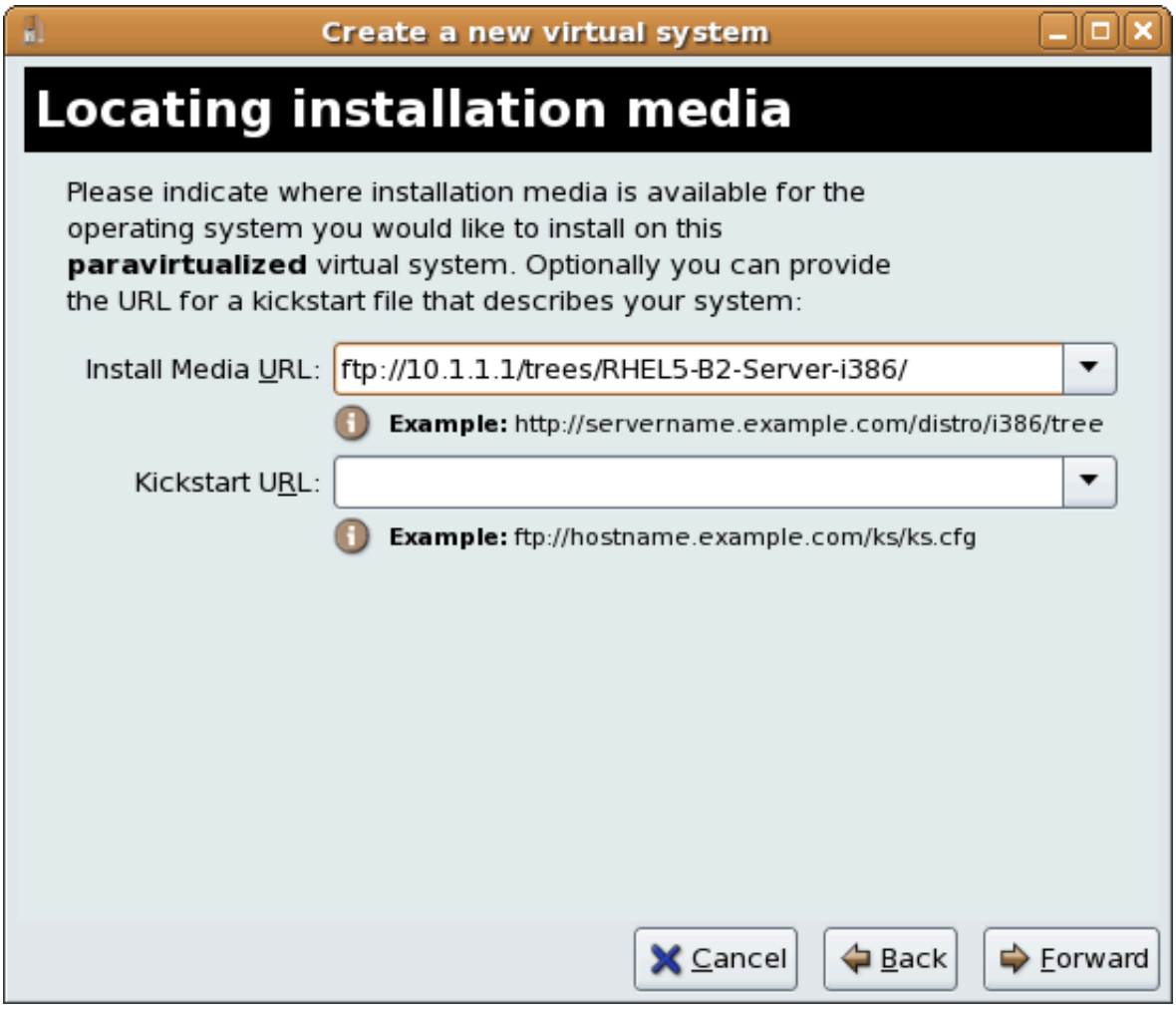
Нажмите кнопку продолжения.

6. В следующем окне можно выбрать установочный носитель. Окно будет выглядеть по разному в зависимости от сделанного на предыдущем этапе выбора.

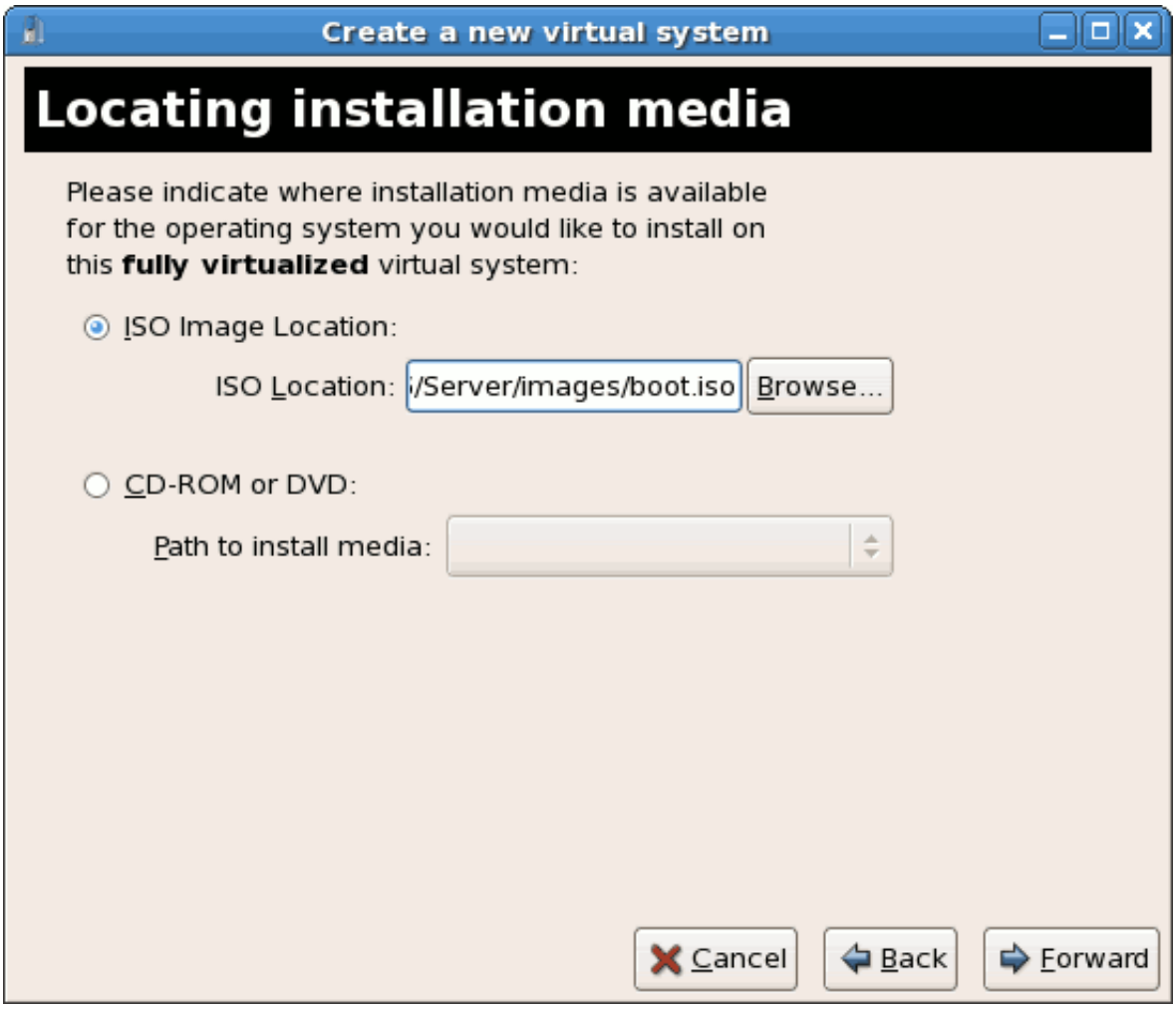
   a. Для выполнения паравиртуализированной установки понадобится установочное дерево, доступ к которому возможен по HTTP, FTP или NFS. То есть ссылка на установочный носитель должна содержать дерево установки Fedora, которое доступно по NFS, FTP, HTTP, а для размещения сетевых служб и файлов можно использовать другой узел или зеркало.

      При использовании ISO-образов DVD или CD-ROM необходимо, чтобы монтируемые файлы были доступны по одному из уже перечисленных протоколов.

      Или же можно просто скопировать дерево установки с зеркала Fedora.



b. При установке полностью виртуализированной системы надо будет указать путь к локальному загрузочному образу DVD или CD (с расширением *.iso или *.img). Обычно Windows-установка использует DVD, CD или ISO-файл, а установки UNIX и Linux в большинстве случаев устанавливают базовую систему с ISO, прежде чем перейти к завершающим этапам установки, где будет использоваться сетевое дерево установки.



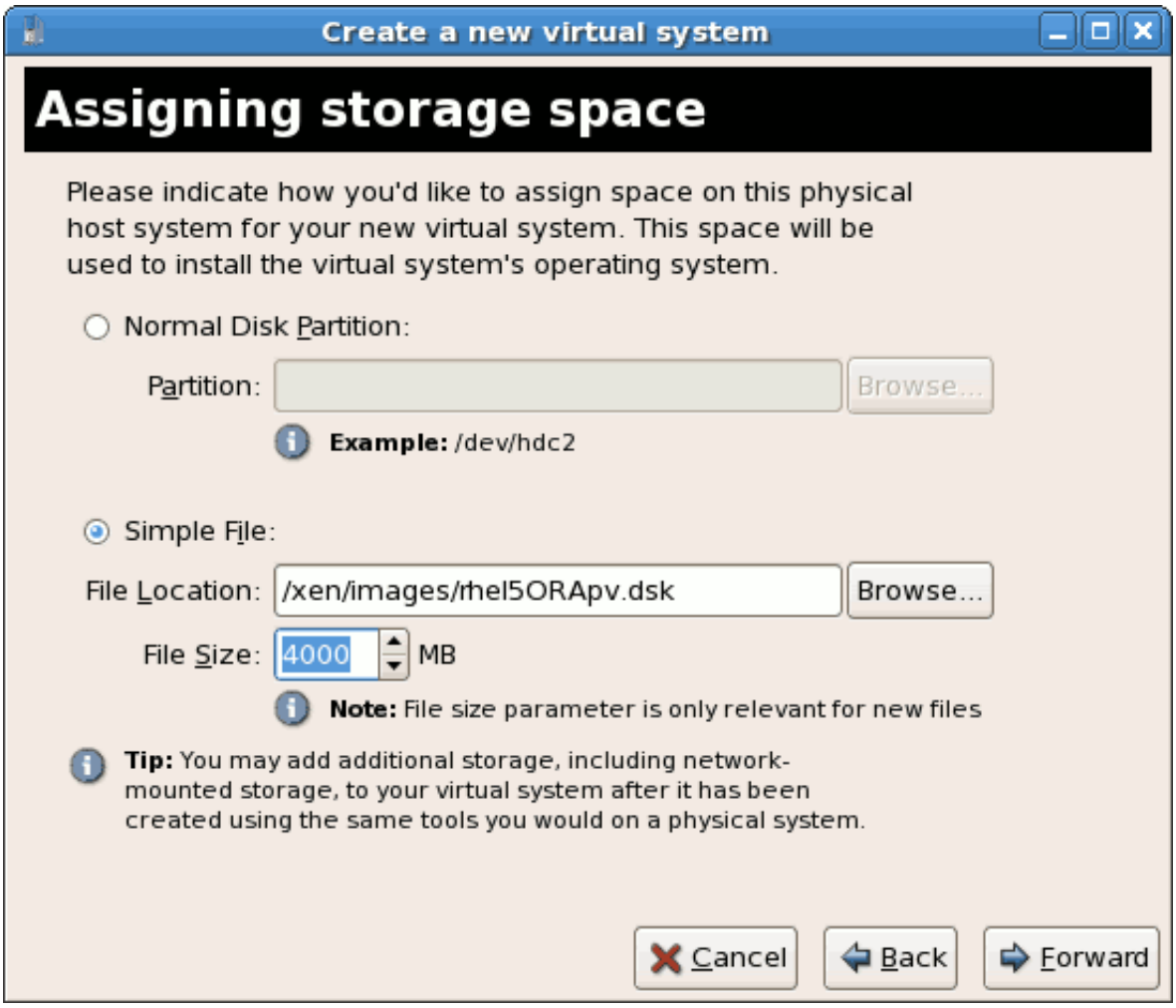
Нажмите кнопку продолжения.

7. The **Assigning storage space** window displays. Choose a disk partition, LUN or create a file based image for the guest storage.

   Файловые образы виртуальных машин традиционно хранятся в каталоге **`/var/lib/xen/images/`**. SELinux запрещает использование других каталогов. *Раздел 7.1, «Виртуализация и SELinux»* содержит подробную информацию об установке виртуальных машин.

   Your guest storage image should be larger than the size of the installation, any additional packages and applications, and the size of the guests swap file. The installation process will choose the size of the guest's swap file based on size of the RAM allocated to the guest.

   Allocate extra space if the guest needs additional space for applications or other data. For example, web servers require additional space for log files.



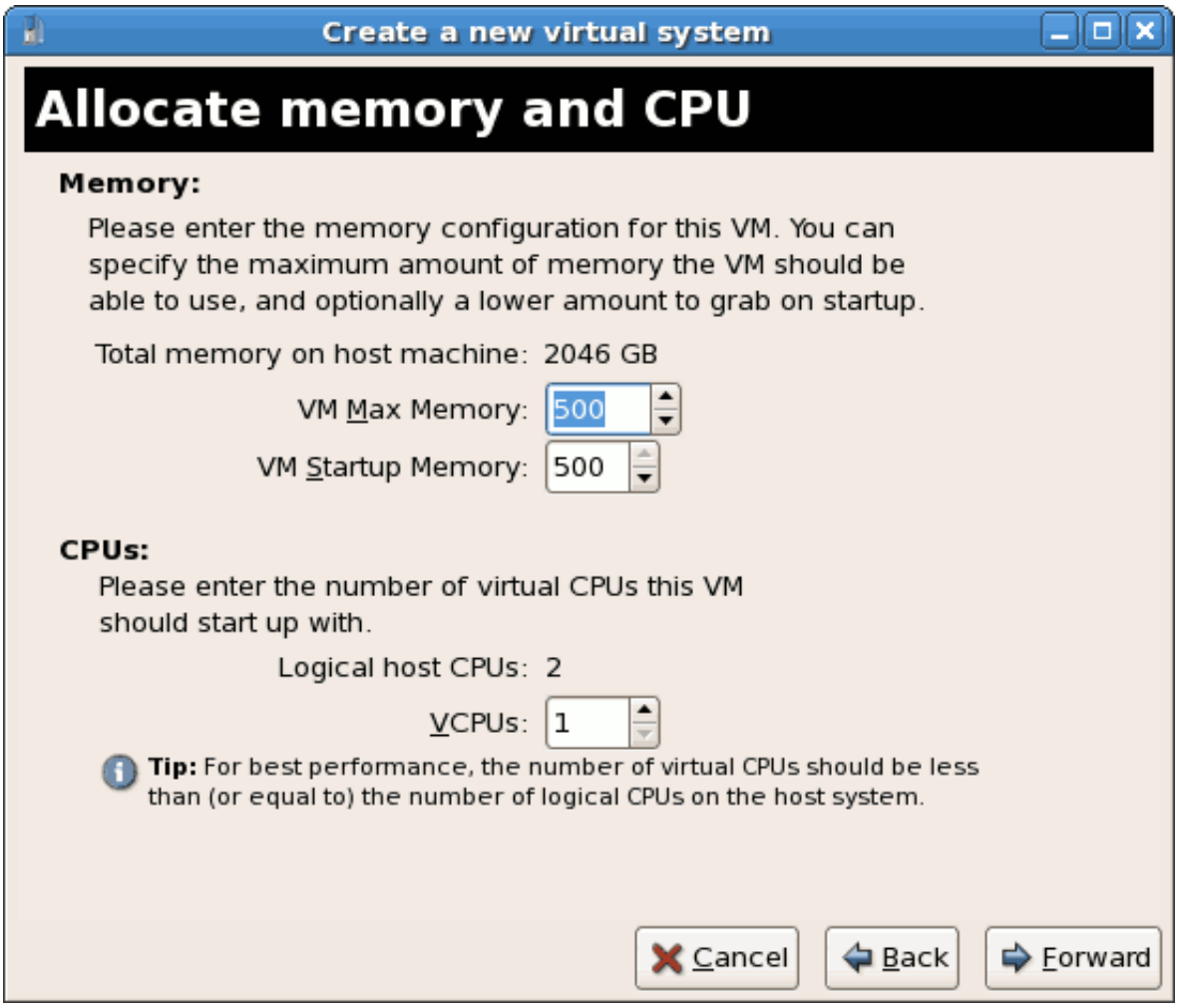
Choose the appropriate size for the guest on your selected storage type and click the **Forward** button.

> **Замечание**
>
> Для хранения образов виртуальных машин рекомендуется использовать стандартный каталог **`/var/lib/xen/images/`** . Если же вы хотите изменить каталог (например, на **`/xen/images/`**), прежде чем приступить к установке, потребуется его добавить в политику SELinux. Позднее будет описано, как изменить политику SELinux.

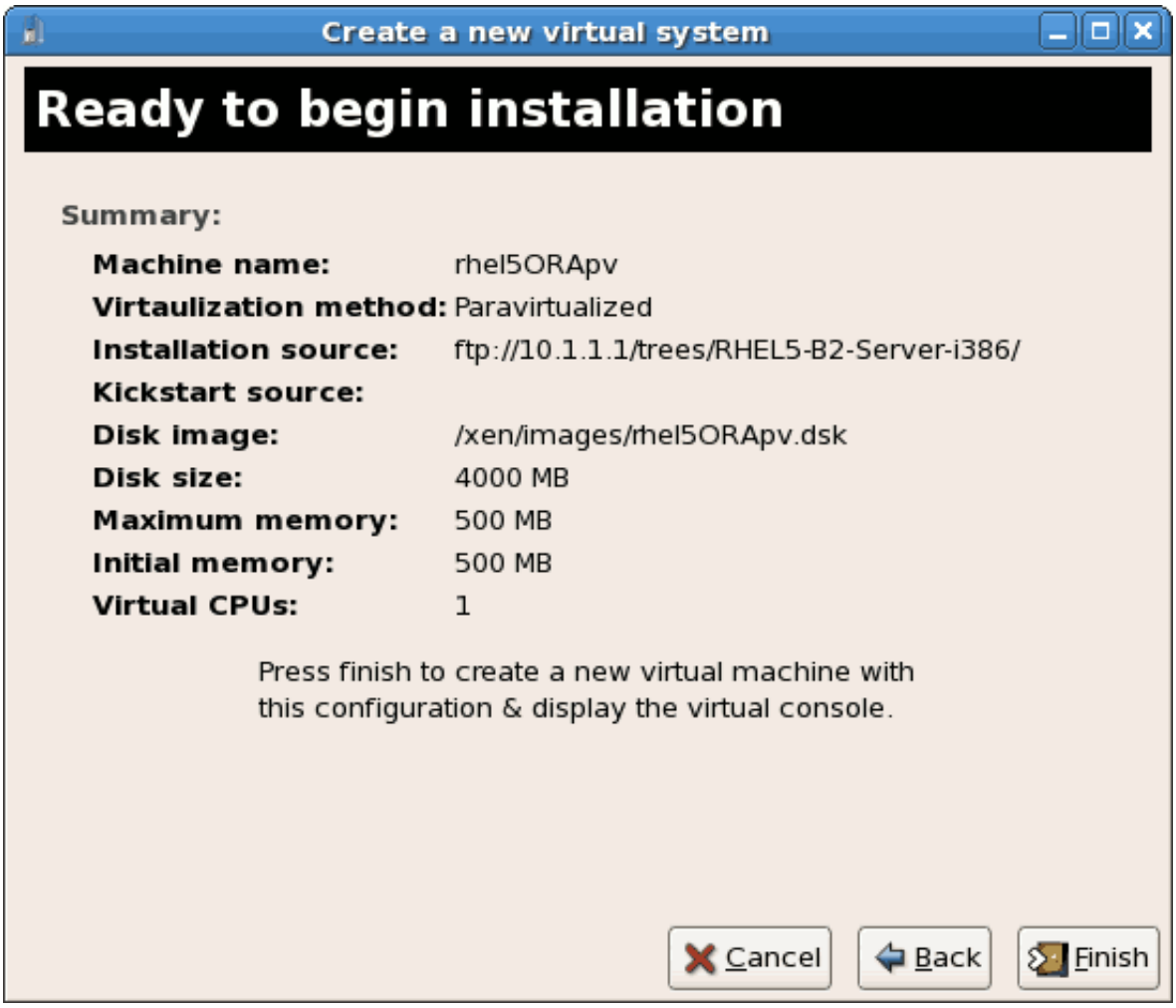
8. The Allocate memory and CPU window displays. Choose appropriate values for the virtualized CPUs and RAM allocation. These values affect the host's and guest's performance.

   Виртуальной машине понадобится достаточный для ее работы объем оперативной памяти (как минимум 512 Мбайт). Стоит помнить, что они используют физическую память. Выполнение слишком большого числа гостей или предоставление размещающей системе недостаточного объема памяти может привести к повышенному использованию виртуальной памяти и области подкачки. Как известно, виртуальная память значительно медленнее физической, как следствие, работа системы существенно замедлится. Этого можно избежать, выделив гостевым системам достаточный объем памяти.

Assign sufficient virtual CPUs for the virtualized guest. If the guest runs a multithreaded application assign the number of virtualized CPUs it requires to run most efficiently. Do not assign more virtual CPUs than there are physical processors (or hyper-threads) available on the host system. It is possible to over allocate virtual processors, however, over allocating has a significant, negative affect on guest and host performance due to processor context switching overheads.



9. На следующем экране будут показаны введенные вами данные. Чтобы внести изменения, нажмите кнопку возврата, а чтобы начать установку, нажмите кнопку завершения.



Появится окно VNC, в котором можно следить за установкой.

Так выглядит процесс создания виртуальных машин с помощью **`virt-manager`**. *Глава 3, Установка гостевой операционной системы* содержит пошаговые инструкции по установке большинства операционных систем.

2.3. Установка виртуальных машин с помощью PXE

Для установки виртуальной машины с помощью PXE необходимо наличие общего сетевого устройства — сетевого моста. Ниже будет рассмотрено, как создать такой мост и использовать его при PXE-установке.

1. **Создайте новый мост**
   a. Создайте файл сценария в каталоге **`/etc/sysconfig/network-scripts/`**. В приведенном примере файл с именем **`ifcfg-installation`** содержит определение моста с именем *`installation`*.

```
# cd /etc/sysconfig/network-scripts/
# vim ifcfg-installation
DEVICE=installation
TYPE=Bridge
```

```
BOOTPROTO=dhcp
ONBOOT=yes
```

Warning

The line, *`TYPE=Bridge`*, is case-sensitive. It must have uppercase 'B' and lower case 'ridge'.

b.
```
Запустите созданный мост.
# ifup installation
```

c. К нему еще не были добавлены интерфейсы. Выполните команду **`brctl show`** для получения информации о всех сетевых мостах в системе.

```
# brctl show
bridge name     bridge id               STP enabled     interfaces
installation    8000.000000000000       no
virbr0          8000.000000000000       yes
```

Мост **`virbr0`** используется по умолчанию утилитой **`libvirt`** для преобразования адресов NAT.

2. **Добавьте интерфейс**

Откройте файл конфигурации интерфейса и добавьте параметр **`BRIDGE`** и укажите имя созданного выше моста.

```
# Intel Corporation Gigabit Network Connection
DEVICE=eth1
BRIDGE=installation
BOOTPROTO=dhcp
HWADDR=00:13:20:F7:6E:8E
ONBOOT=yes
```

Перезапустите сетевое окружение или полностью перезагрузите систему.

```
# service network restart
```

Убедитесь, что интерфейс был подключен:

```
# brctl show
bridge name     bridge id               STP enabled     interfaces
installation    8000.001320f76e8e       no              eth1
virbr0          8000.000000000000       yes
```

3. **Обеспечение защиты**

Configure **`iptables`** to allow all traffic to be forwarded across the bridge.

```
# iptables -I FORWARD -m physdev --physdev-is-bridged -j ACCEPT
# service iptables save
# service iptables restart
```

Disable iptables on bridges

Alternatively, prevent bridged traffic from being processed by **iptables** rules. In **/etc/sysctl.conf** append the following lines:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

Reload the kernel parameters configured with **sysctl**

```
# sysctl -p /etc/sysctl.conf
```

4. **Перезапустите libvirt**
   Restart the **libvirt** daemon.

```
# service libvirtd reload
```

Мост успешно настроен, можно приступить к установке.

PXE-установка с помощью virt-install

В строке **virt-install** добавьте параметр **--network=bridge:МОСТ** (замените «МОСТ» именем моста, в данном случае — «installation»). Для PXE-установок используется параметр *--pxe*.

```
# virt-install --accelerate --hvm --connect qemu:///system \
    --network=bridge:installation --pxe\
    --name EL10 --ram=756 \
    --vcpus=4
    --os-type=linux --os-variant=rhel5
    --file=/var/lib/libvirt/images/EL10.img \
```

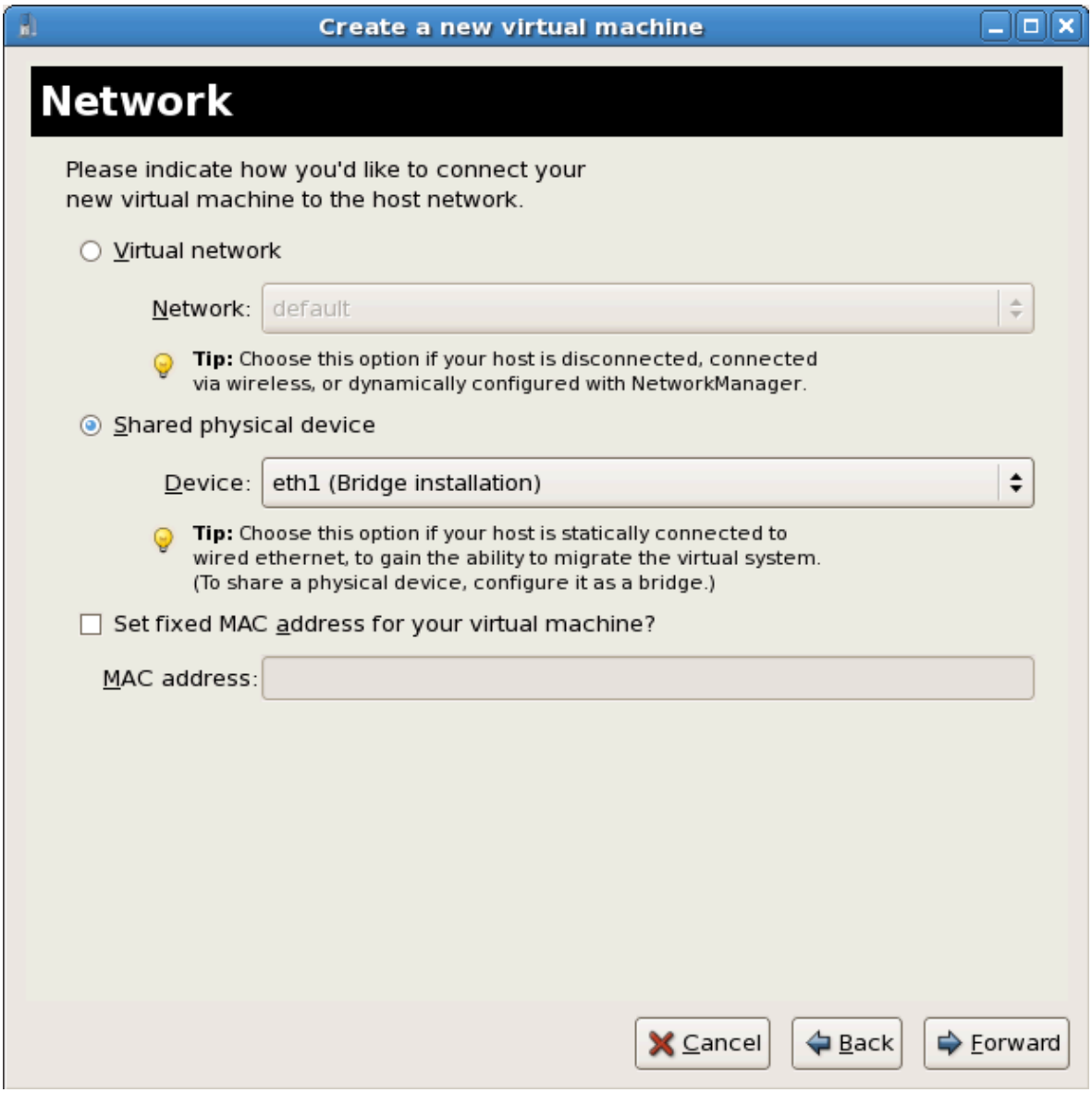
Пример 2.3. PXE-установка с помощью virt-install

PXE-установка с помощью virt-manager

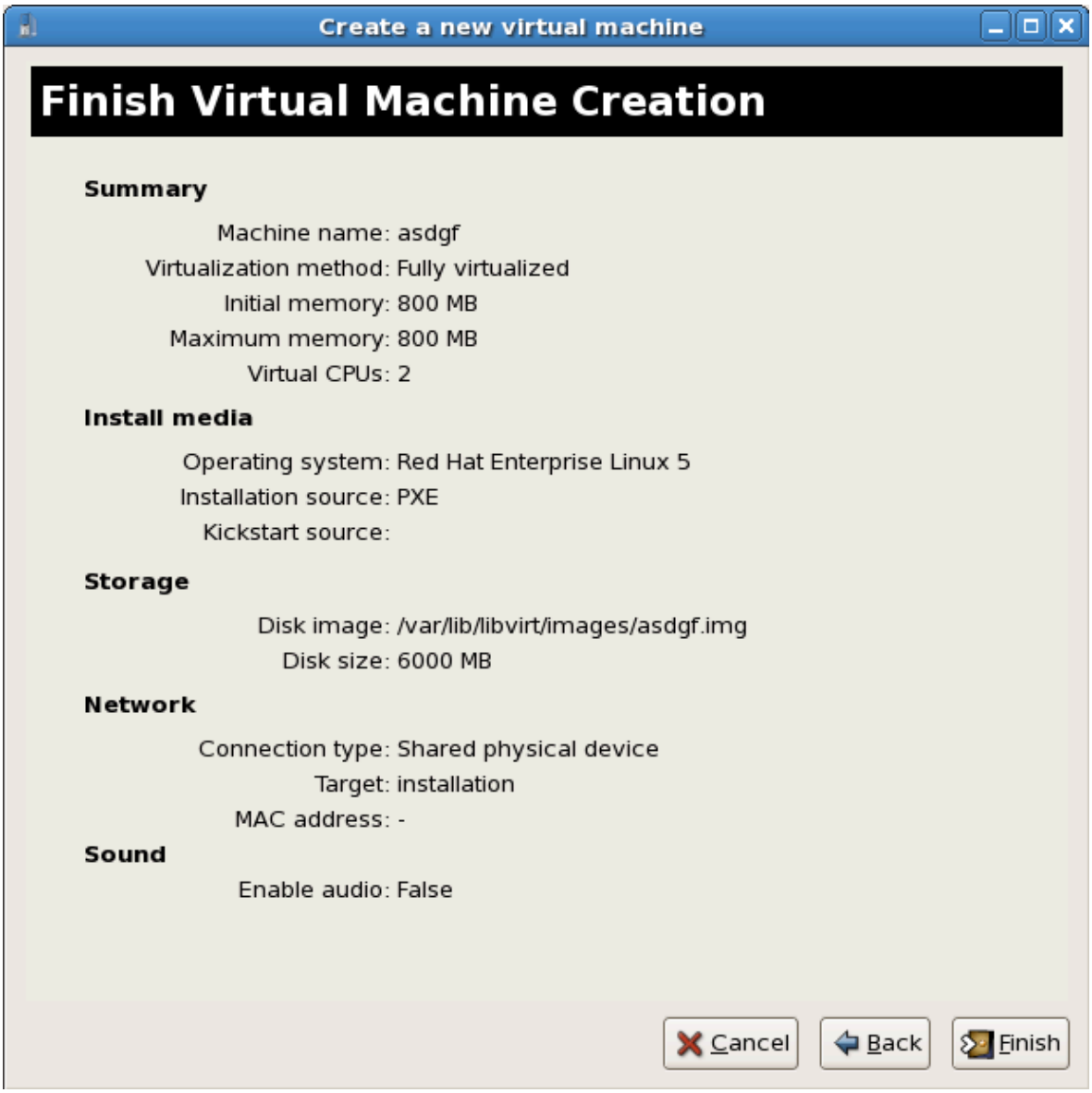
Приведенные здесь действия отличаются от стандартной последовательности шагов при установке с помощью virt-manager (см. *Глава 3, Установка гостевой операционной системы*).

1. **Выберите PXE**
   В качестве способа установки выберите PXE.

2. **Выберите мост**
   Отметьте **Общее физическое устройство** (Shared physical device) и выберите созданный ранее мост.



3. **Начните установку**
   Все готово к установке.



Будет отправлен DHCP-запрос и, если найден действующий PXE-сервер, начнется процесс установки виртуальной машины.

Установка гостевой операционной системы

В этой главе будут описаны процессы установки разных гостевых операционных систем в виртуализированном окружении Fedora. *Глава 2, Обзор установки гостевых виртуальных машин* содержит общую информацию об установке.

3.1. Установка Red Hat Enterprise Linux 5 в качестве паравиртуализированного гостя

Дальше будет описан процесс установки Red Hat Enterprise Linux 5 в качестве паравиртуализированной гостевой системы. Паравиртуализация характеризуется более высокой скоростью работы по сравнению с полной виртуализацией, в то же время обладая всеми ее достоинствами. Для организации паравиртуализации необходимо специальное ядро **`kernel-xen`**.

Важное замечание

Паравиртуализация возможна при наличии гипервизора Xen. С гипервизором KVM паравиртуализация невозможна.

Прежде чем приступить к установке, убедитесь, что у вас есть права доступа root.

Здесь рассматривается установка Red Hat Enterprise Linux с удаленного сервера. Приведенные инструкции по установке аналогичны инструкциям по выполнению минимальной установки с Live CD.

С помощью virt-manager или virt-install создайте паравиртуализированные гостевые системы Red Hat Enterprise Linux 5. *Раздел 2.2, «Создание виртуальных машин с помощью virt-manager»* содержит подробные инструкции для **`virt-manager`**.

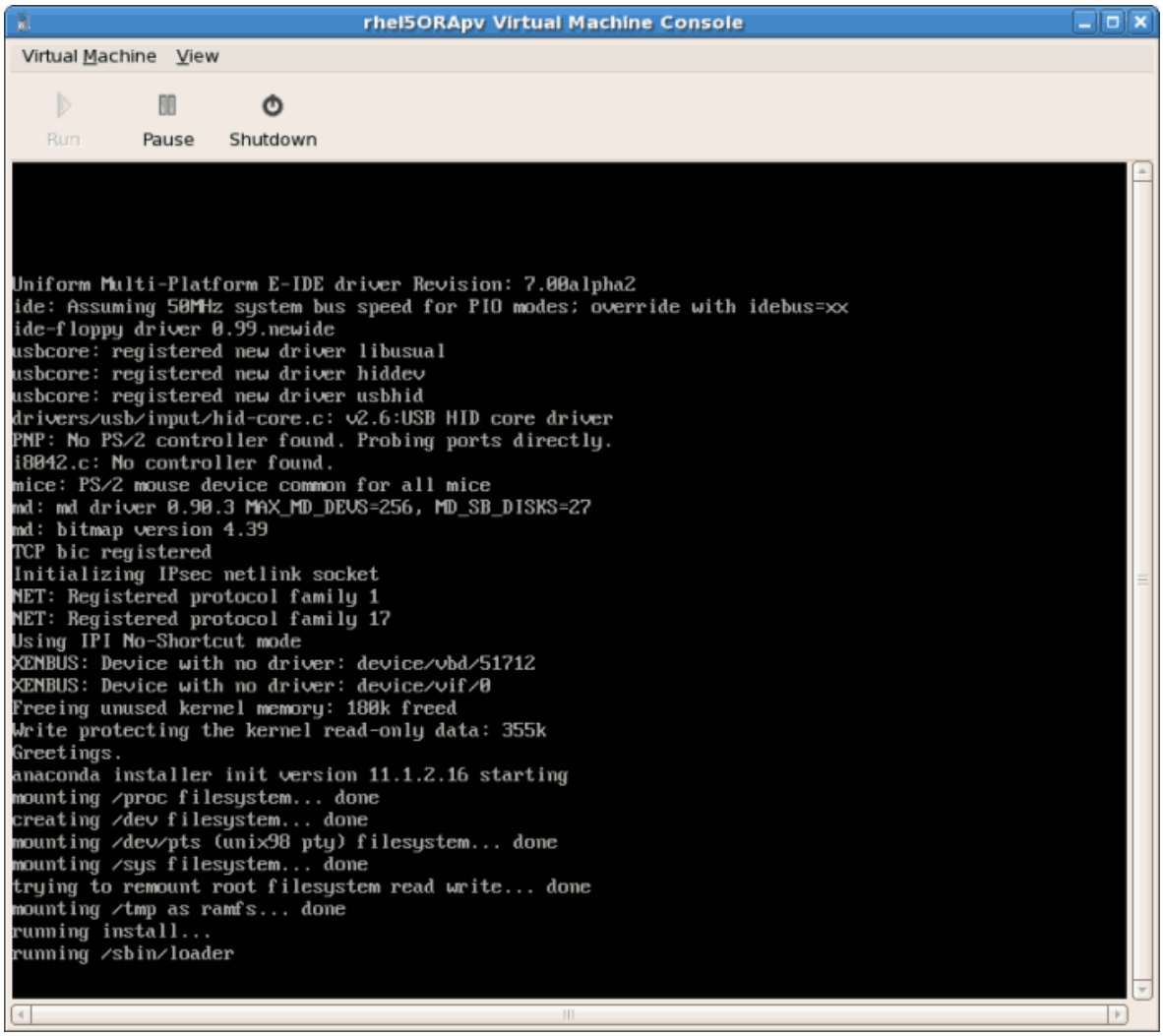
Создайте паравиртуализированную гостевую систему с помощью **`virt-install`** (для графической установки укажите **`--vnc`**). В приведенном ниже примере будет создана система с именем *`rhel5PV`*, ее образ расположен в *`rhel5PV.dsk`*, локальное зеркало дерева установки Red Hat Enterprise Linux 5 — в *`ftp://10.1.1.1/trees/CentOS5-B2-Server-i386/`*.

```
# virt-install -n rhel5PV -r 500 \
-f /var/lib/libvirt/images/rhel5PV.dsk -s 3 --vnc -p \
-l ftp://10.1.1.1/trees/CentOS5-B2-Server-i386/
```

Автоматизация установки

Кикстарт позволяет автоматизировать установку Red Hat Enterprise Linux.

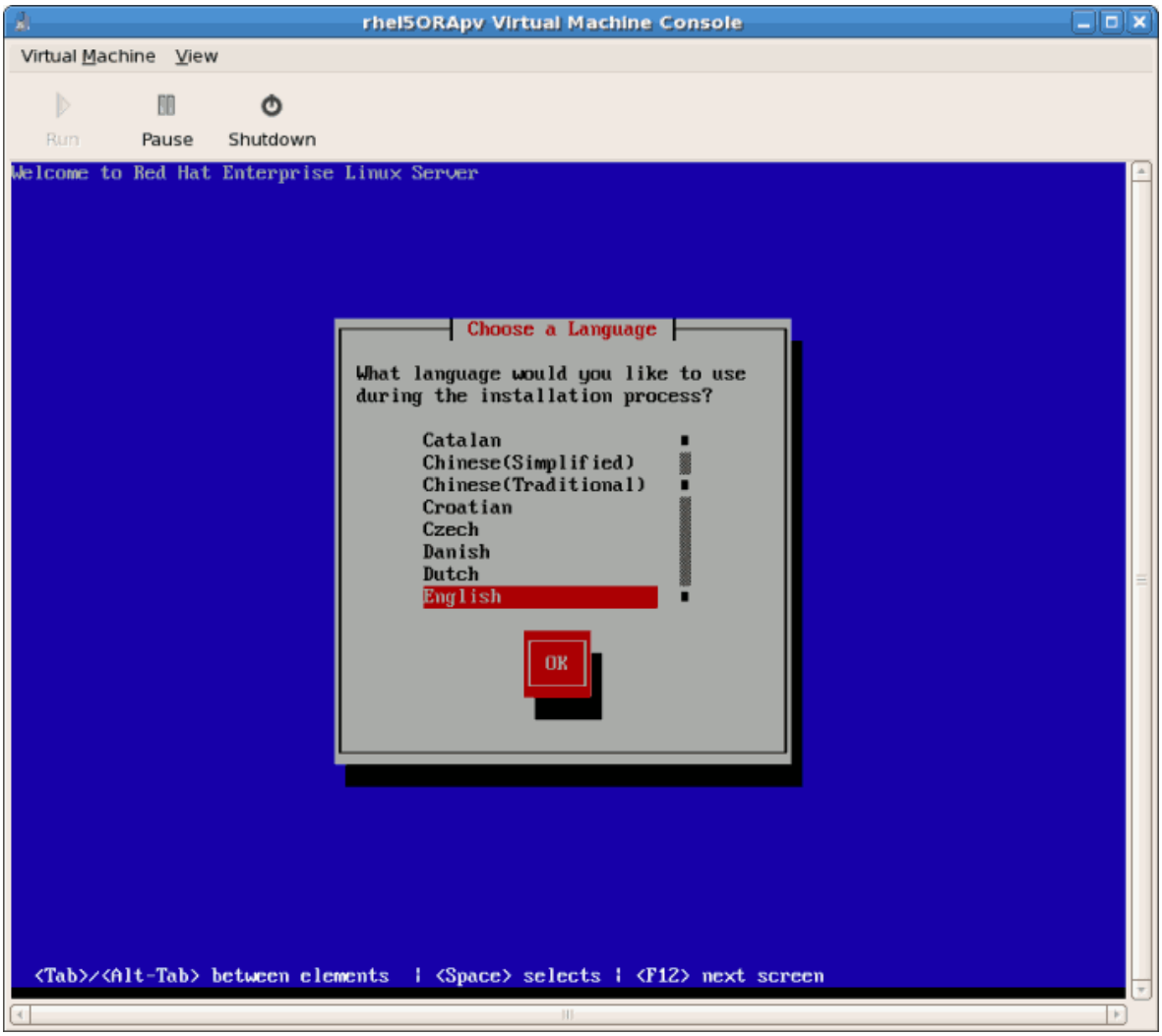
Независимо от того, выполняете ли вы обычную или кикстарт-установку, появится окно исходной загрузки гостевой системы.



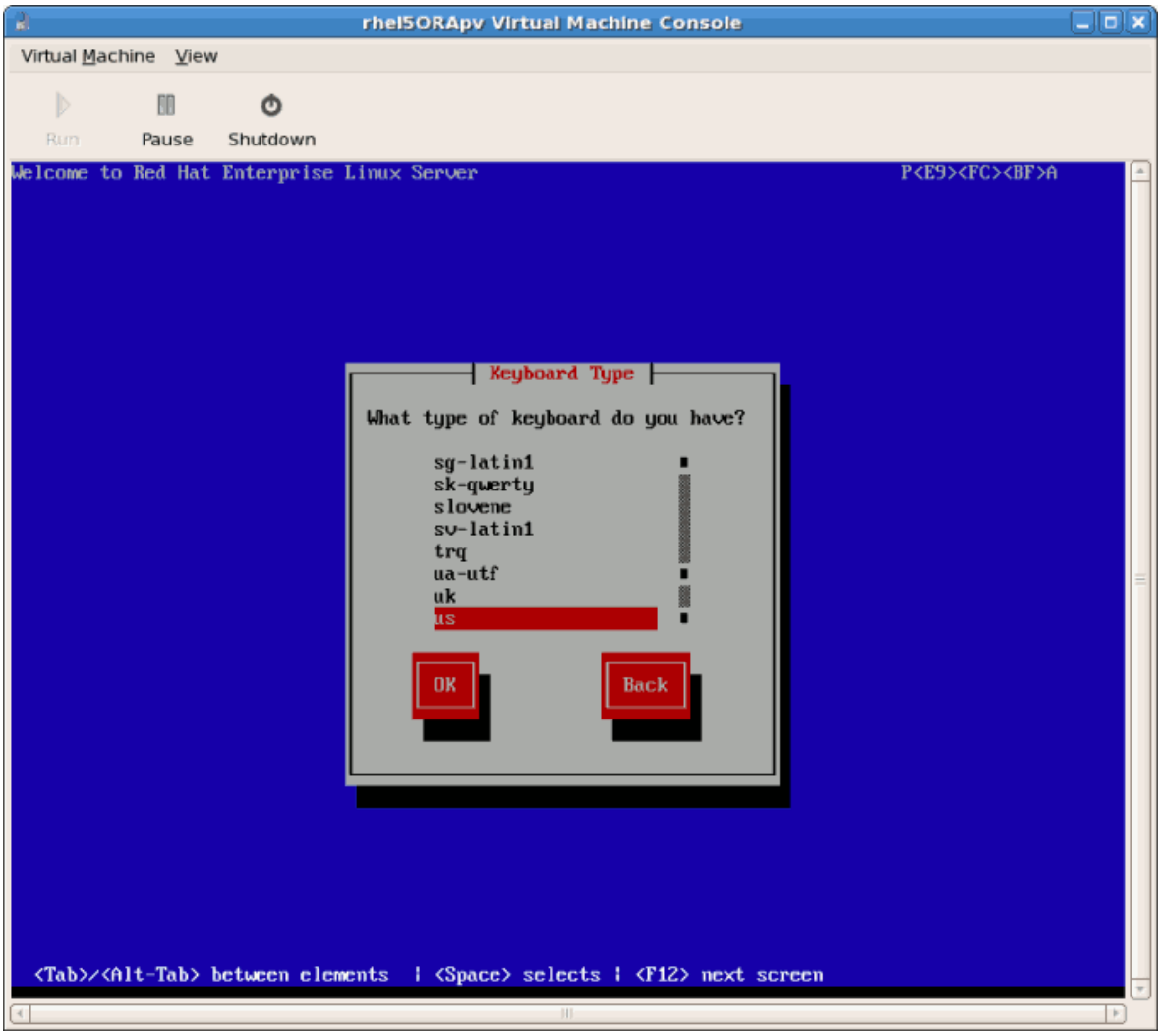
После завершения исходной загрузки начнется стандартный процесс установки Red Hat Enterprise Linux. В большинстве случаев можно принимать предложенные варианты ответов.

Процедура 3.1. Установка паравиртуализированной гостевой системы Red Hat Enterprise Linux

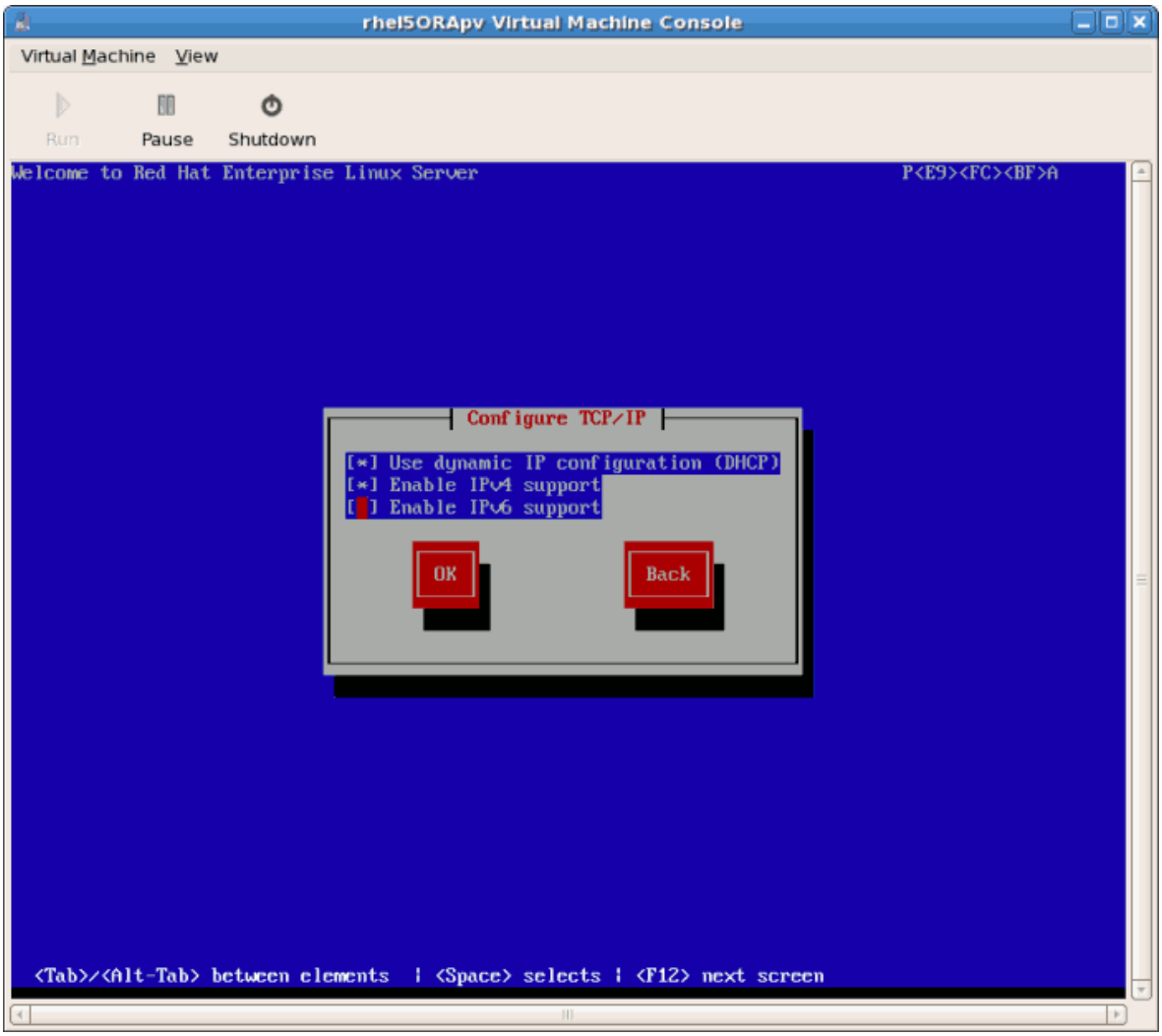
1. Выберите язык и нажмите **OK**.



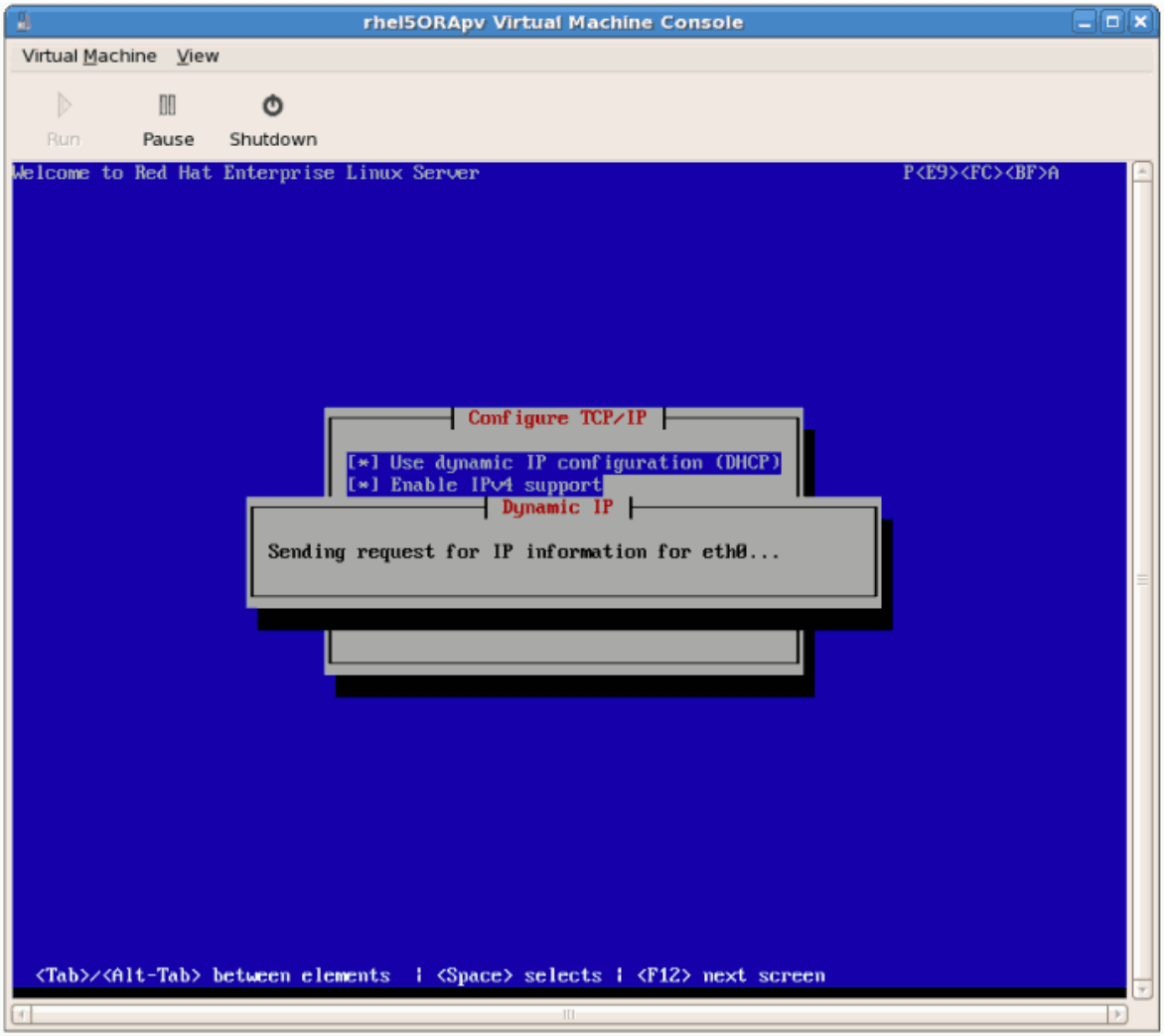
2. Выберите раскладку клавиатуры и нажмите **OK**.



3. Присвойте гостевой системе сетевой адрес. Можно выбрать DHCP (как показано ниже) или статический адрес.



4. Если вы выбрали DHCP, будет предпринята попытка получения IP-адреса.

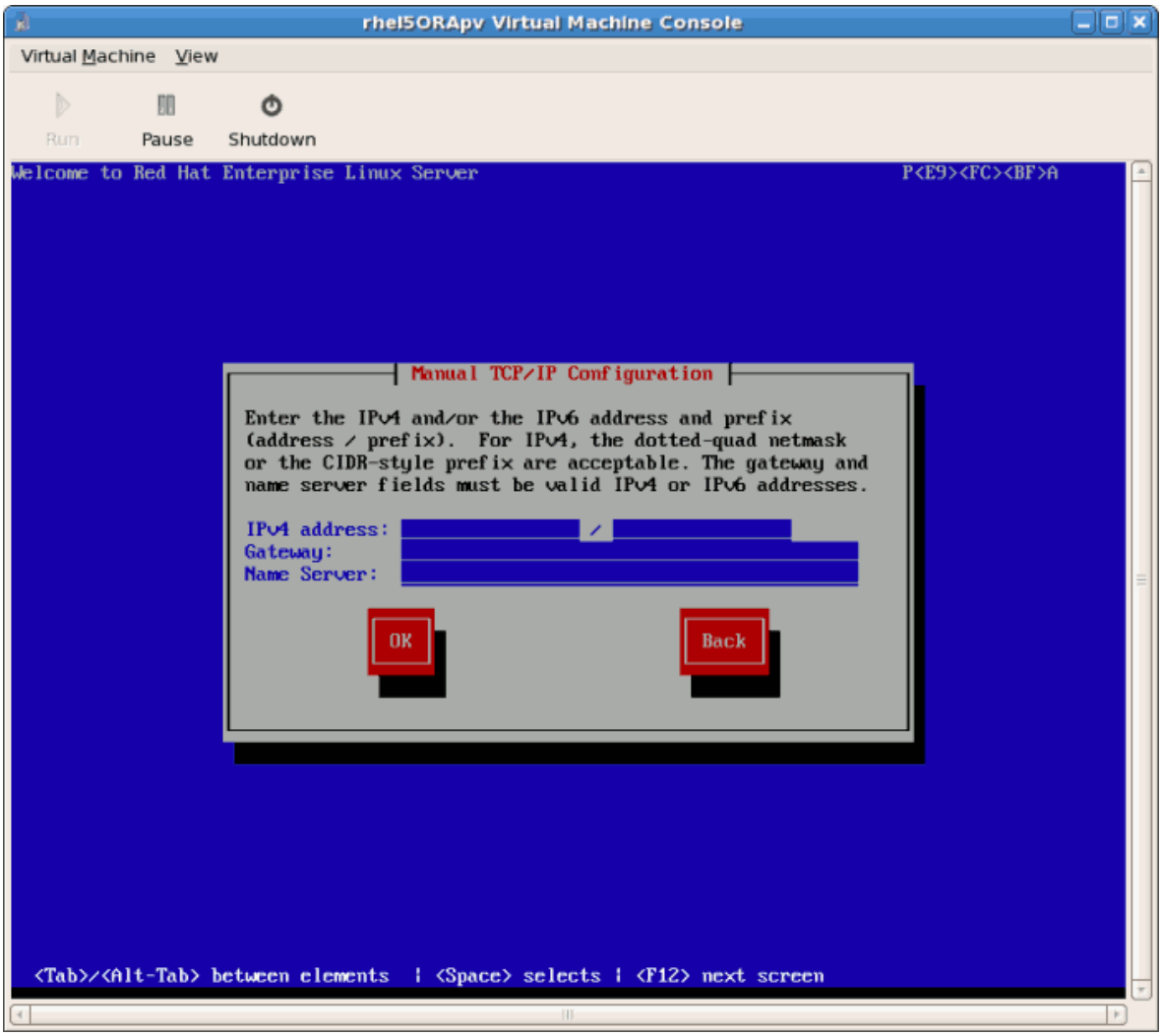


5. Если вы выбрали статический адрес, на следующем экране будет предложено ввести настройки сетевого окружения.

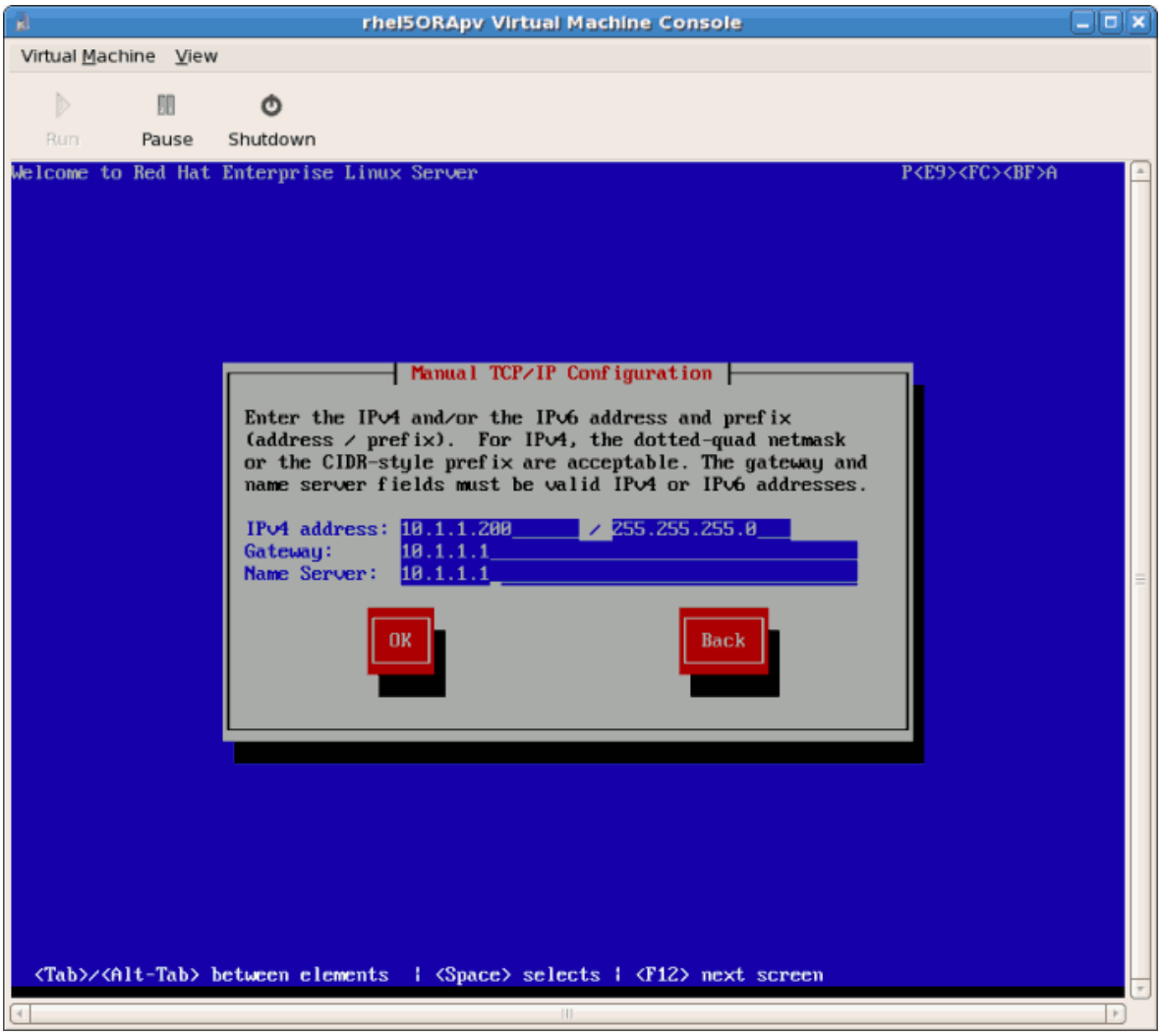
   a. Укажите IP-адрес и убедитесь, что с помощью этого адреса действительно можно достичь сервера с деревом установки.

   b. Укажите маску подсети, шлюз и адрес сервера имен.

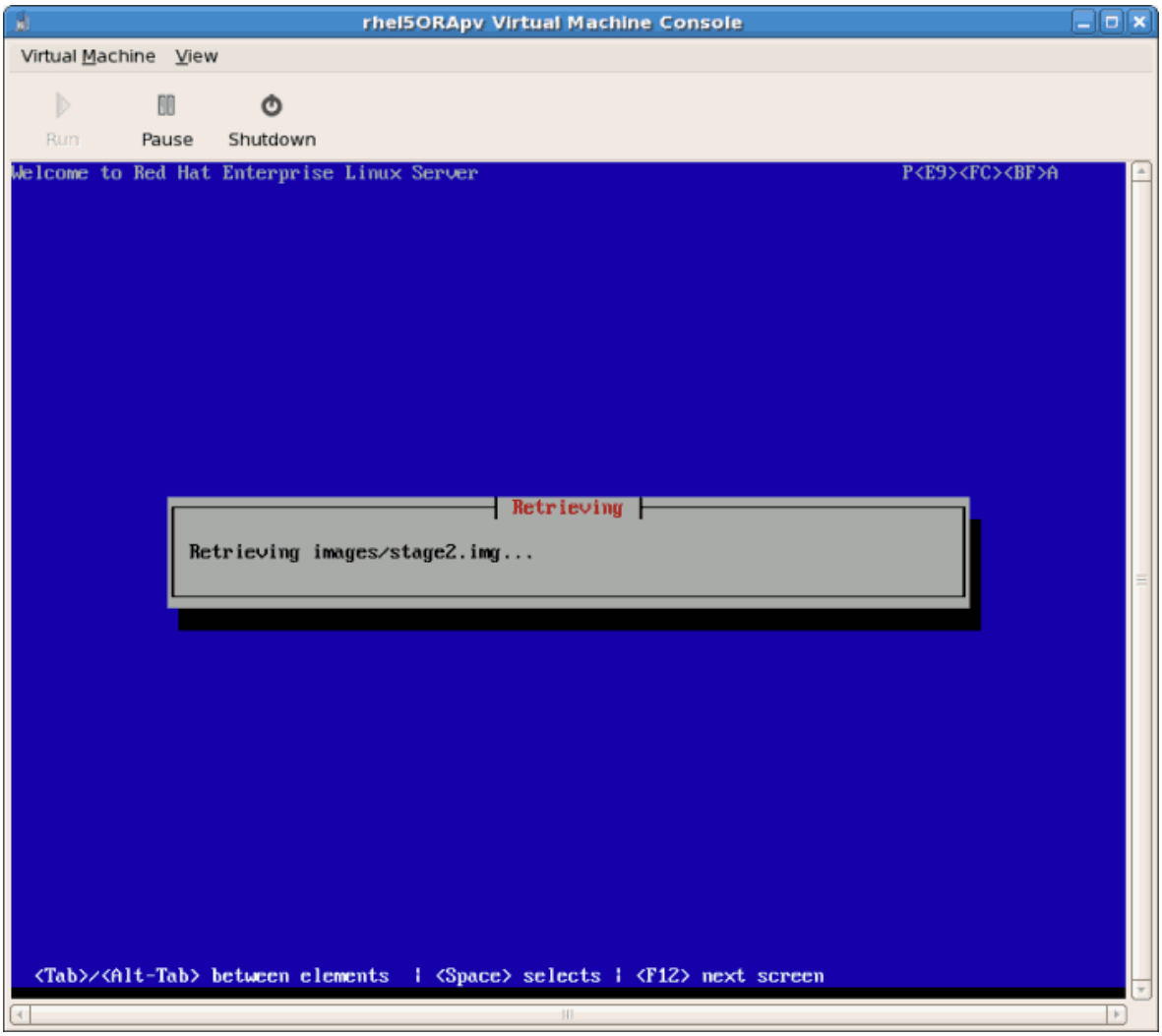
   Выберите язык и нажмите **OK**.



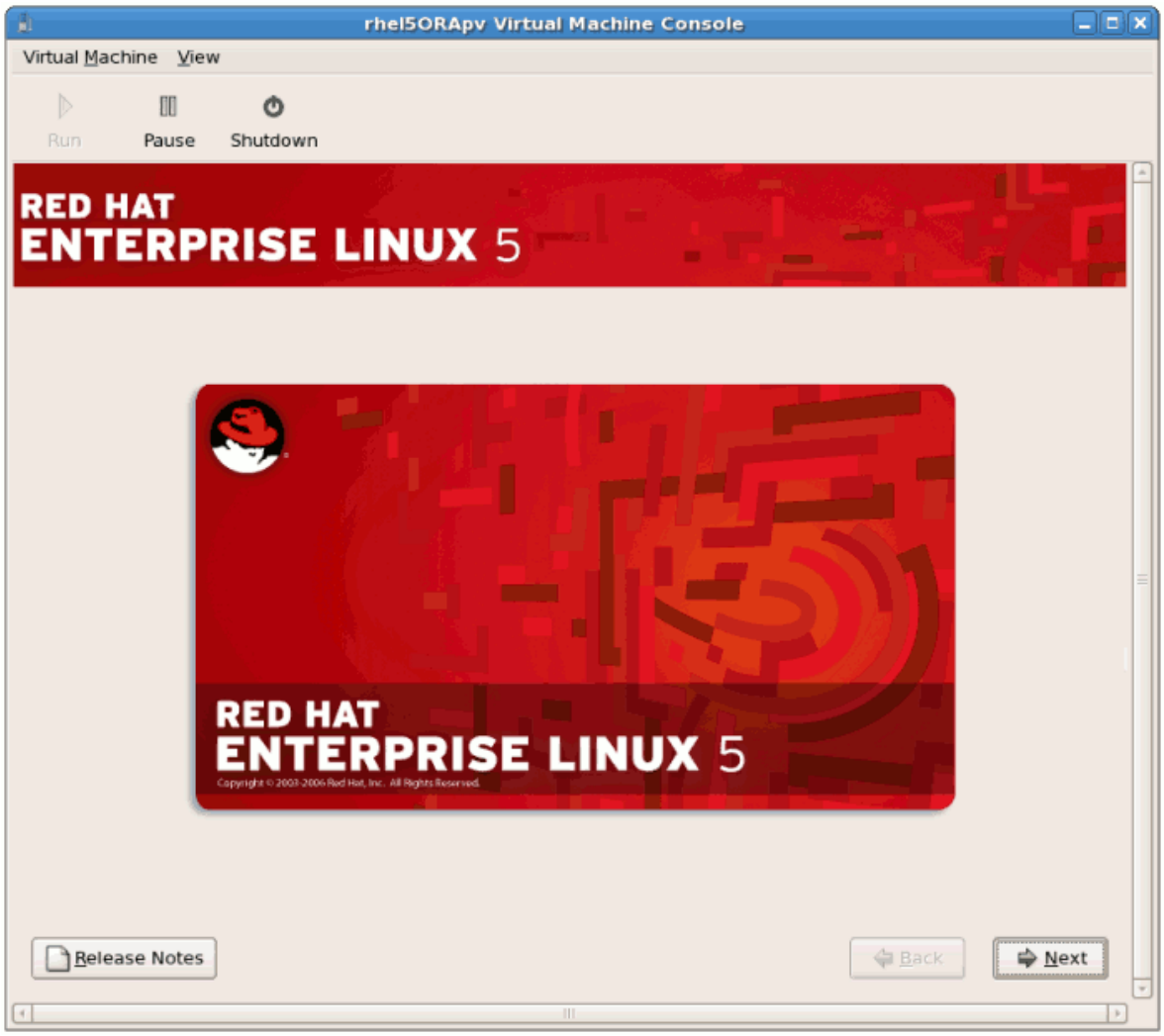
6. Ниже приведен пример настройки статического IP-адреса.



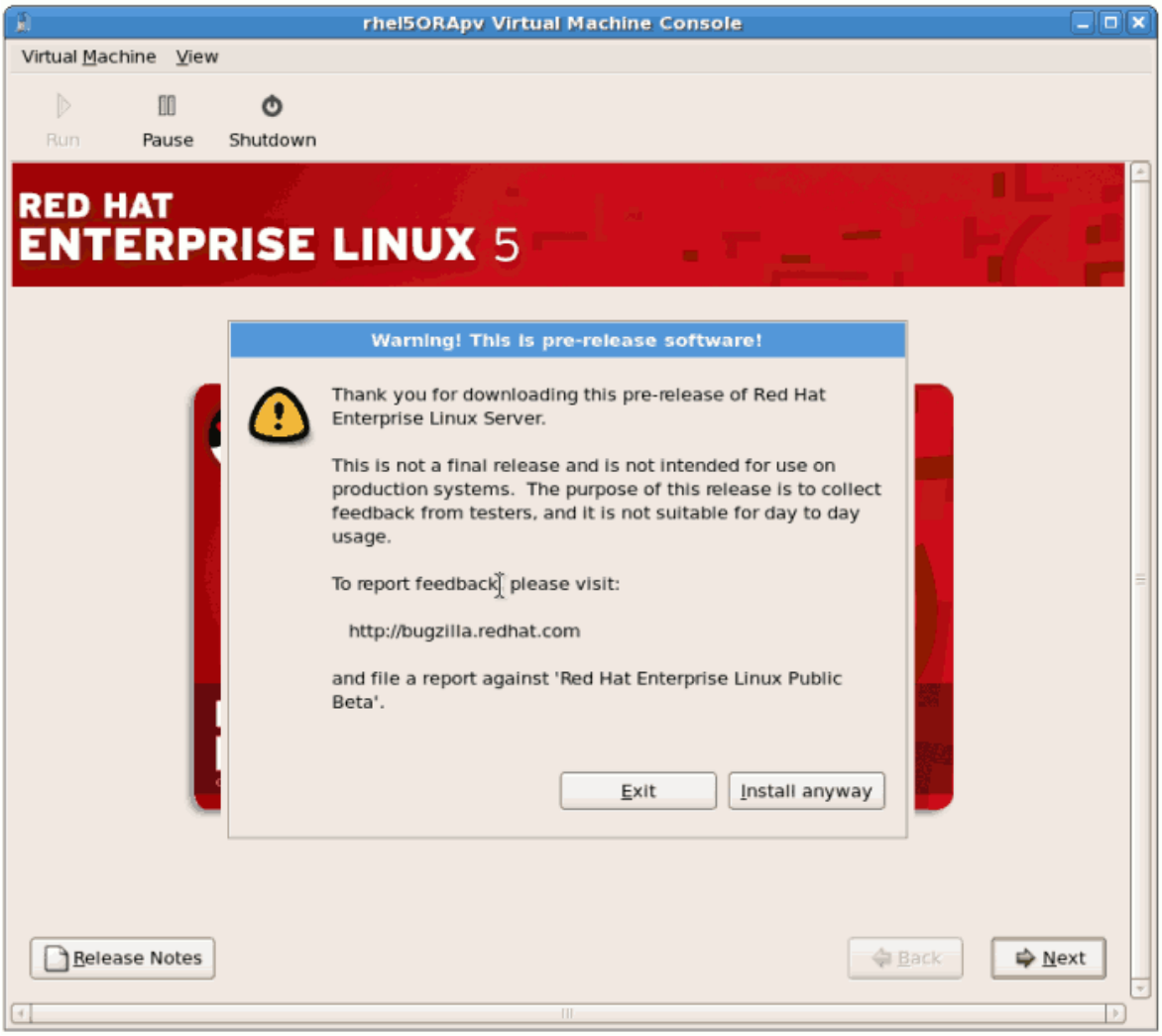
7. Процесс установки приступит к получению файлов с сервера.



Затем начнется графическая установка.

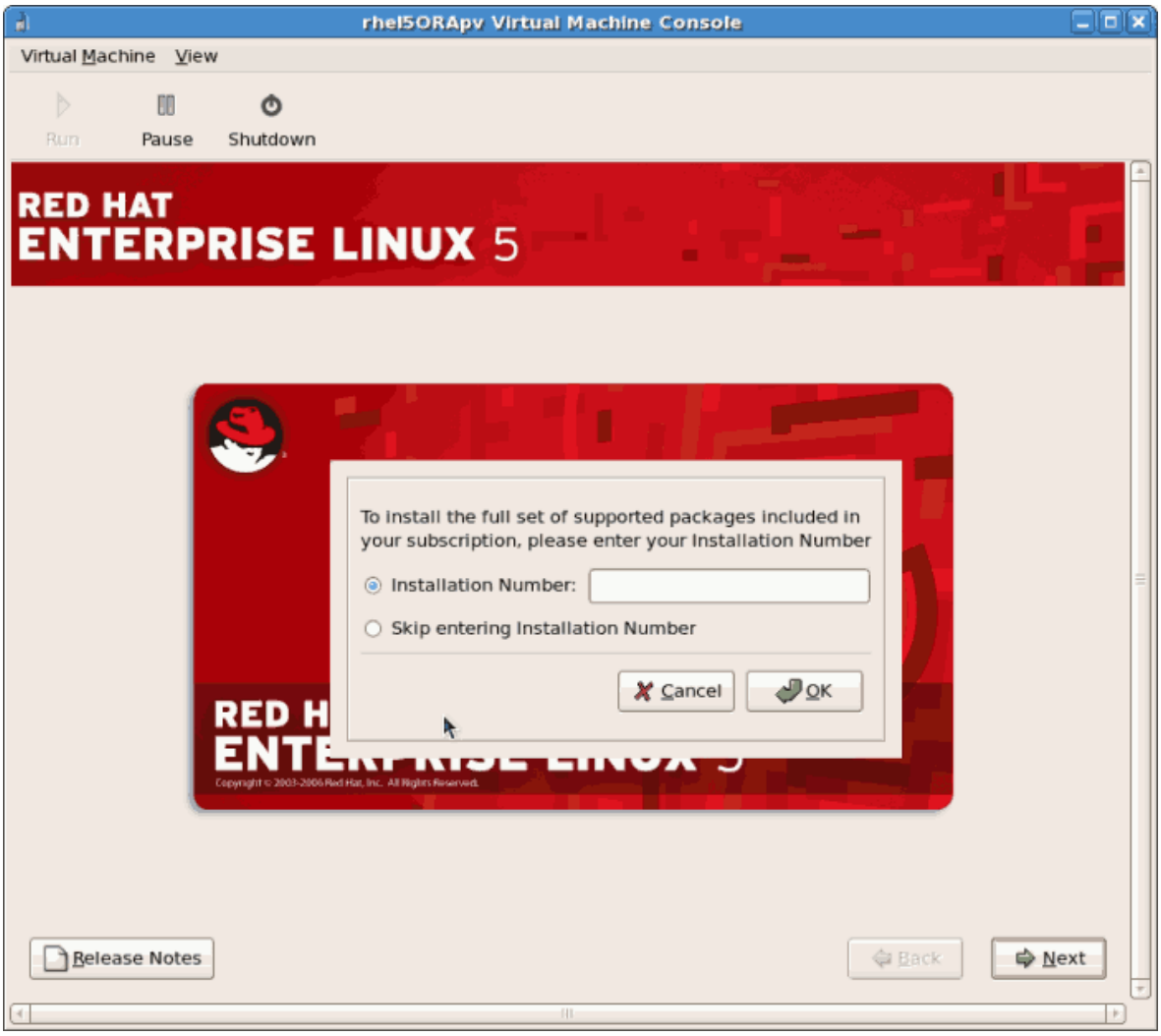


Если вы устанавливаете бета-версию, потребуется подтвердить, что вы действительно хотите установить эту операционную систему. Нажмите кнопку **Установить все равно**, затем **Далее**.



Процедура 3.2. Графическая установка

1. Введите код регистрации. Если у вас есть ключ подписки RHN, введите его в поле Код установки.

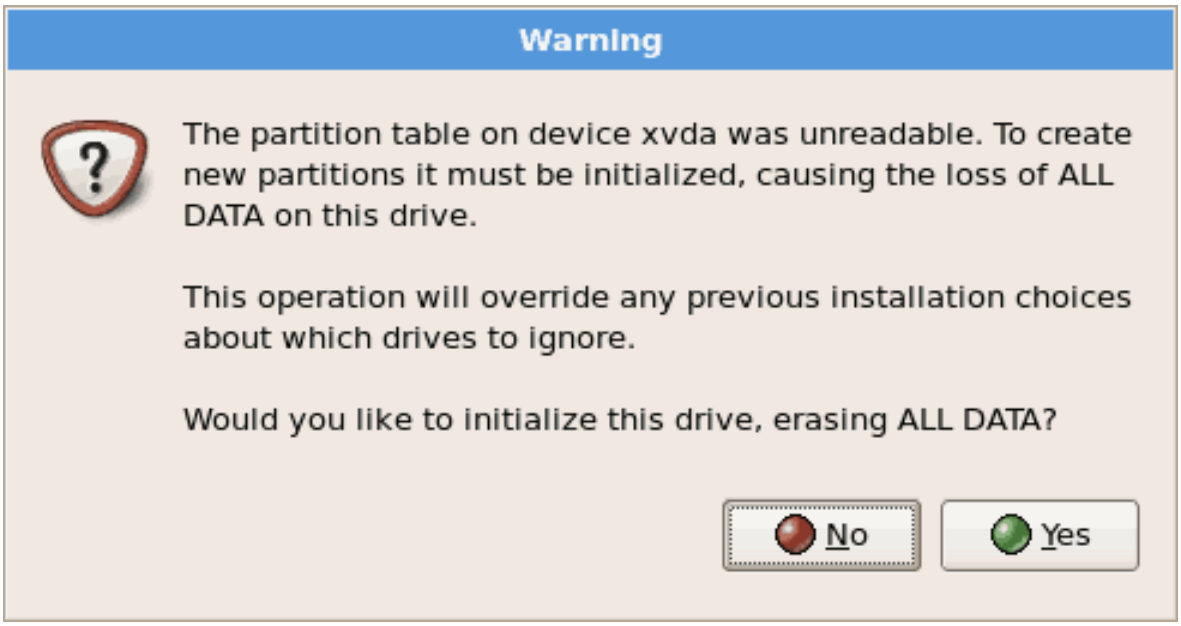


Note

Если вы пропустили регистрацию, данные учетной записи Red Hat Network можно подтвердить после установки с помощью команды **rhn_register** (потребуются права root).

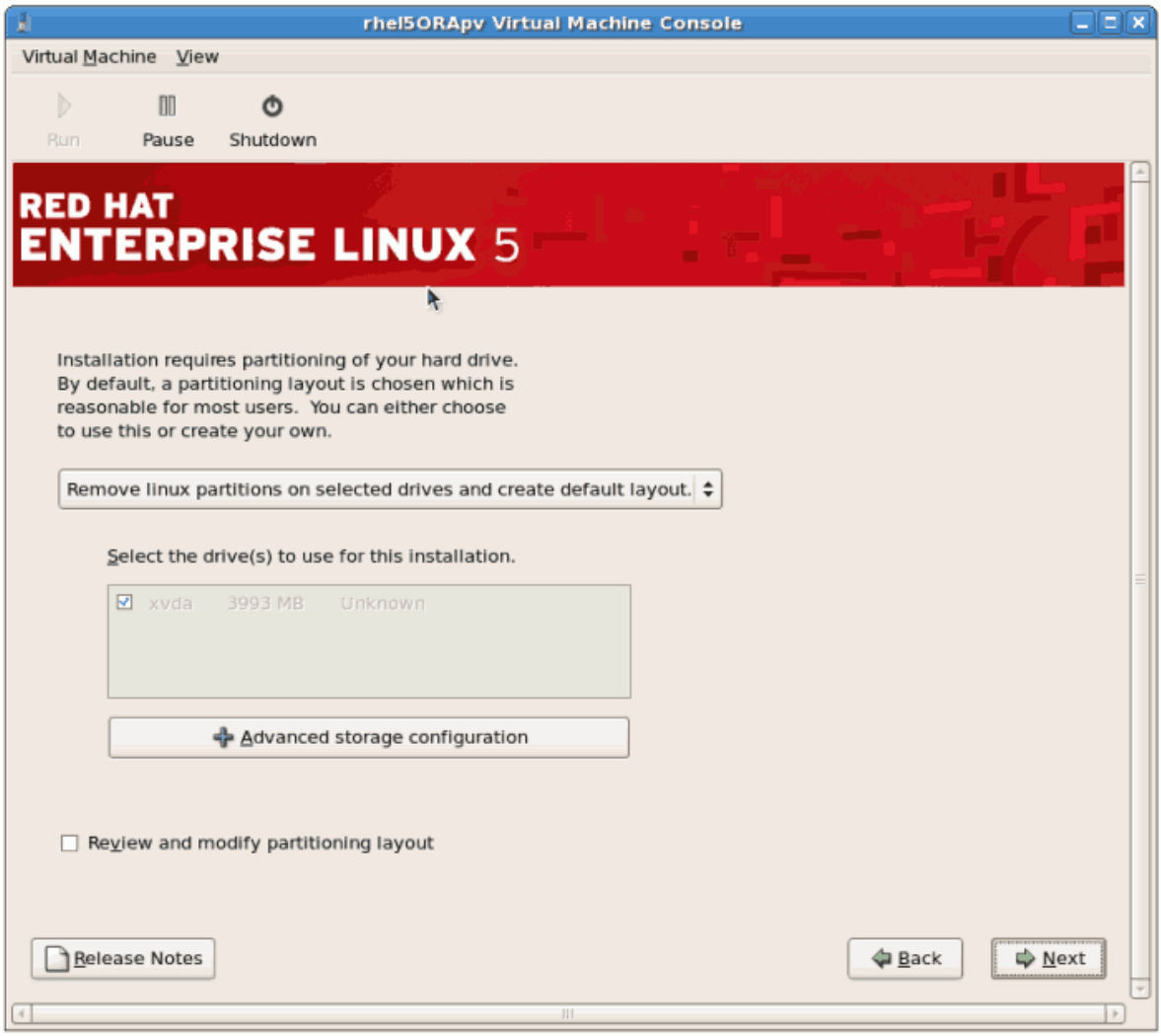
```
# rhn_register
```

2. Будет запрошено подтверждение удаления всех данных на выбранном носителе.



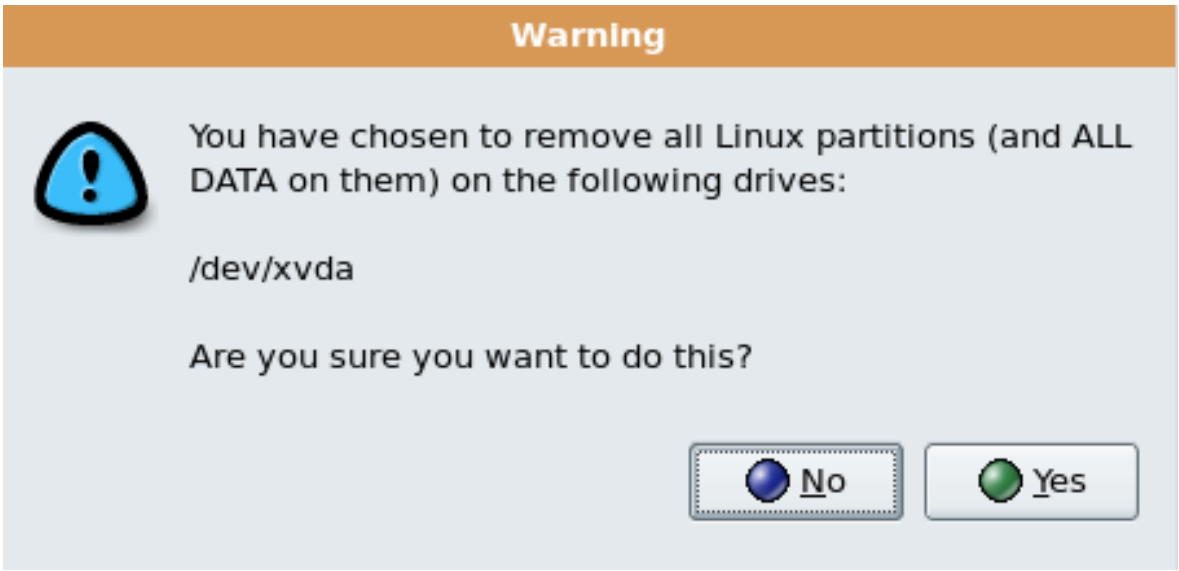
Нажмите **Да**.

3. На этом экране можно просмотреть настройки накопителя и схему разделов. Если вы планируете использовать iSCSI, можно изменить дополнительные настройки накопителя.



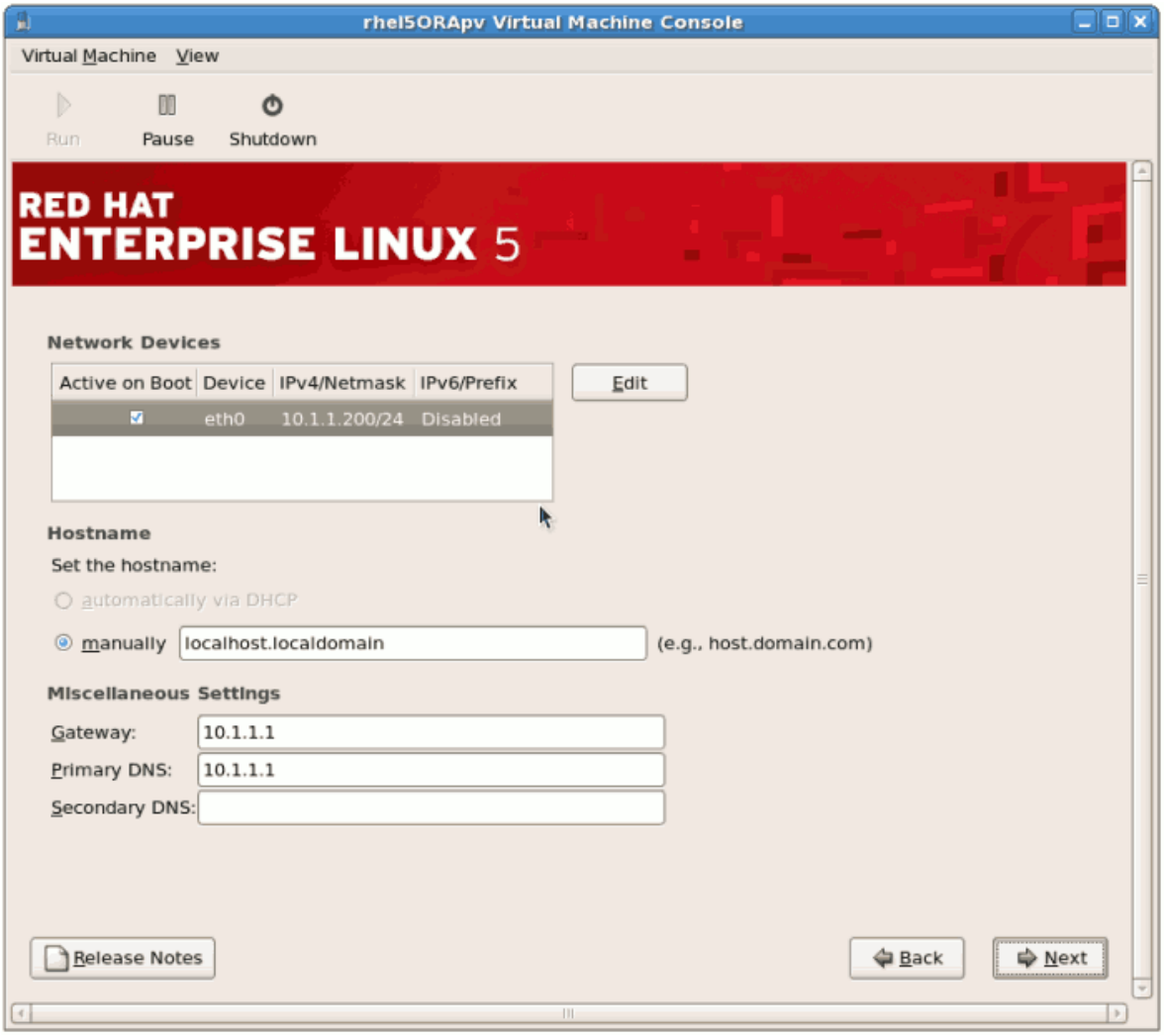
Завершив, нажмите **Далее**.

4. Подтвердите выбор устройства хранения.



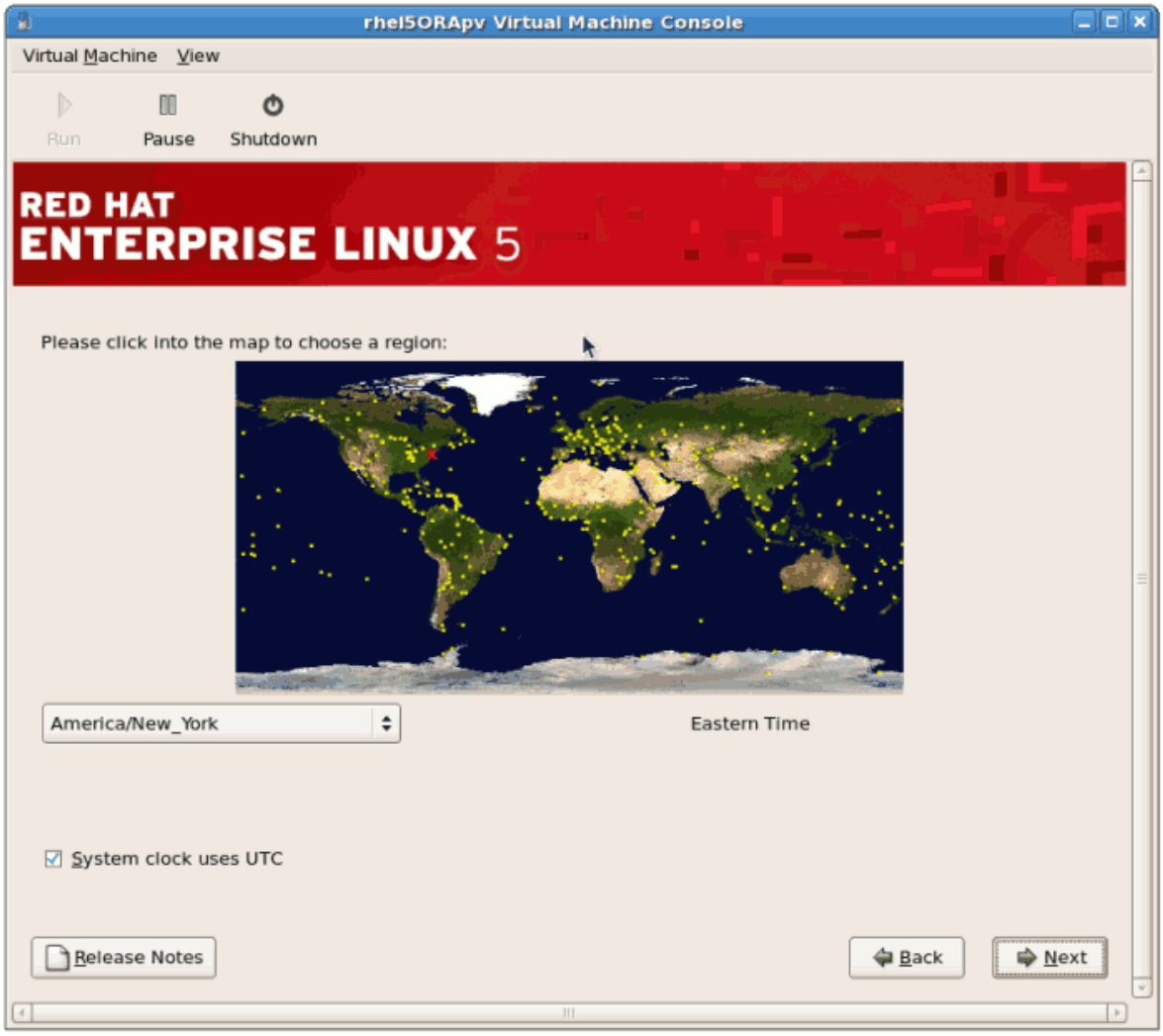
Нажмите **Да**.

5. Следующий экран содержит настройки сетевого окружения, которые были введены раньше в процессе установки. Измените их при необходимости.

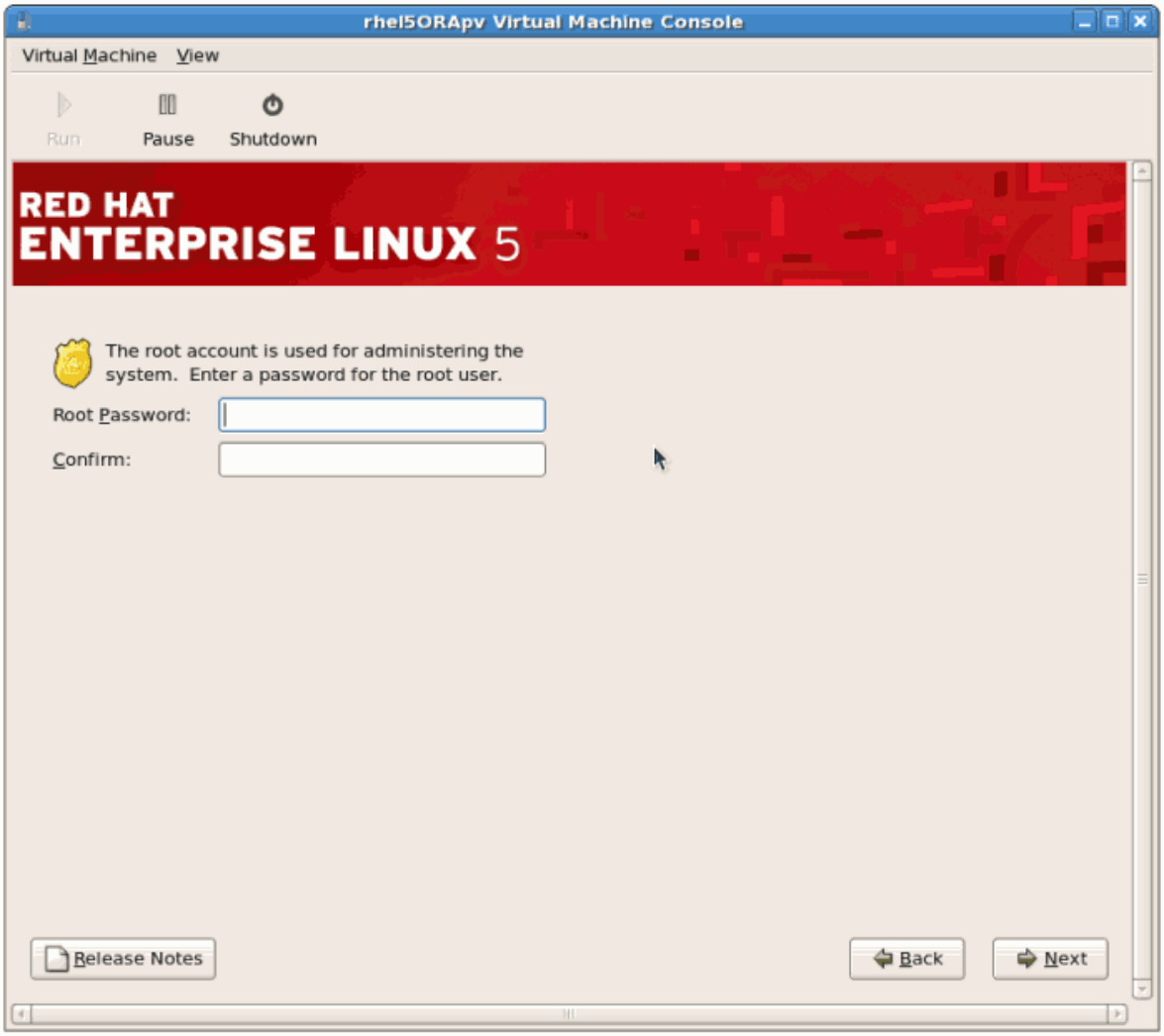


Нажмите **Далее**.

6. Выберите часовой пояс.

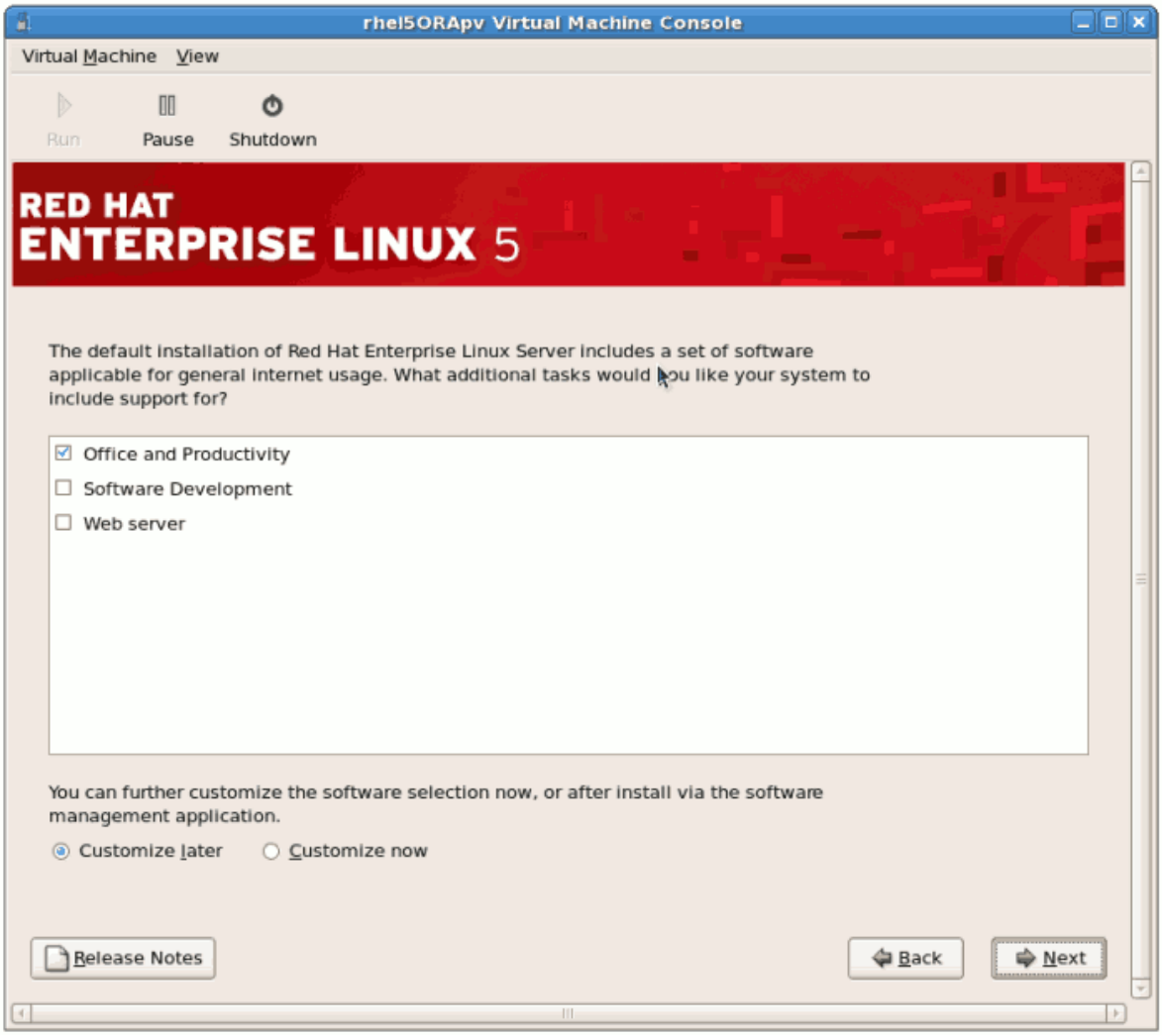


7. Укажите пароль root.



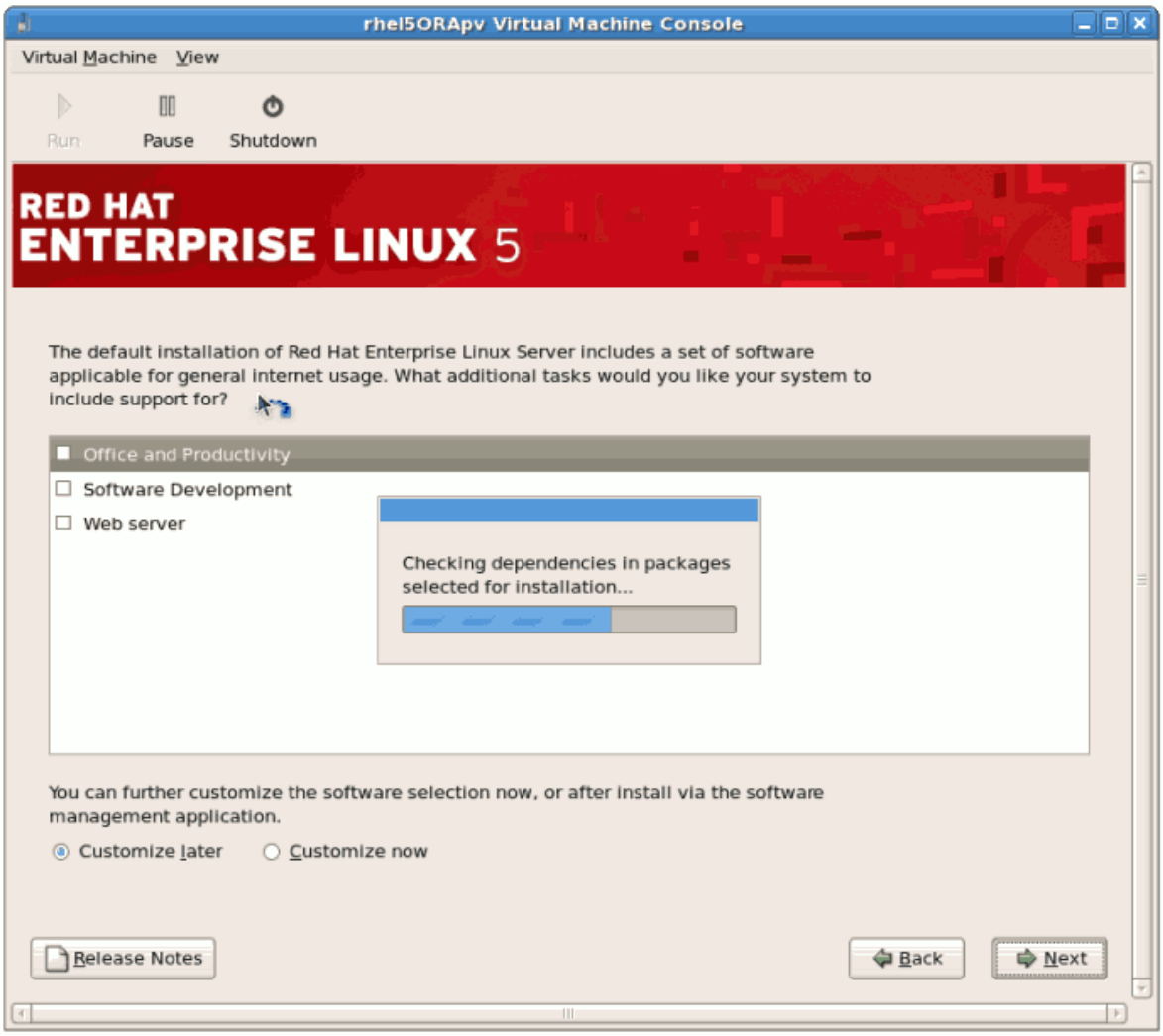
Нажмите **Далее**.

8. Выберите пакеты для установки, для этого нажмите **Настроить сейчас**. Не забудьте выбрать пакет **kernel-xen** из группы **Система** — он необходим для виртуализации.



Нажмите **Далее**.

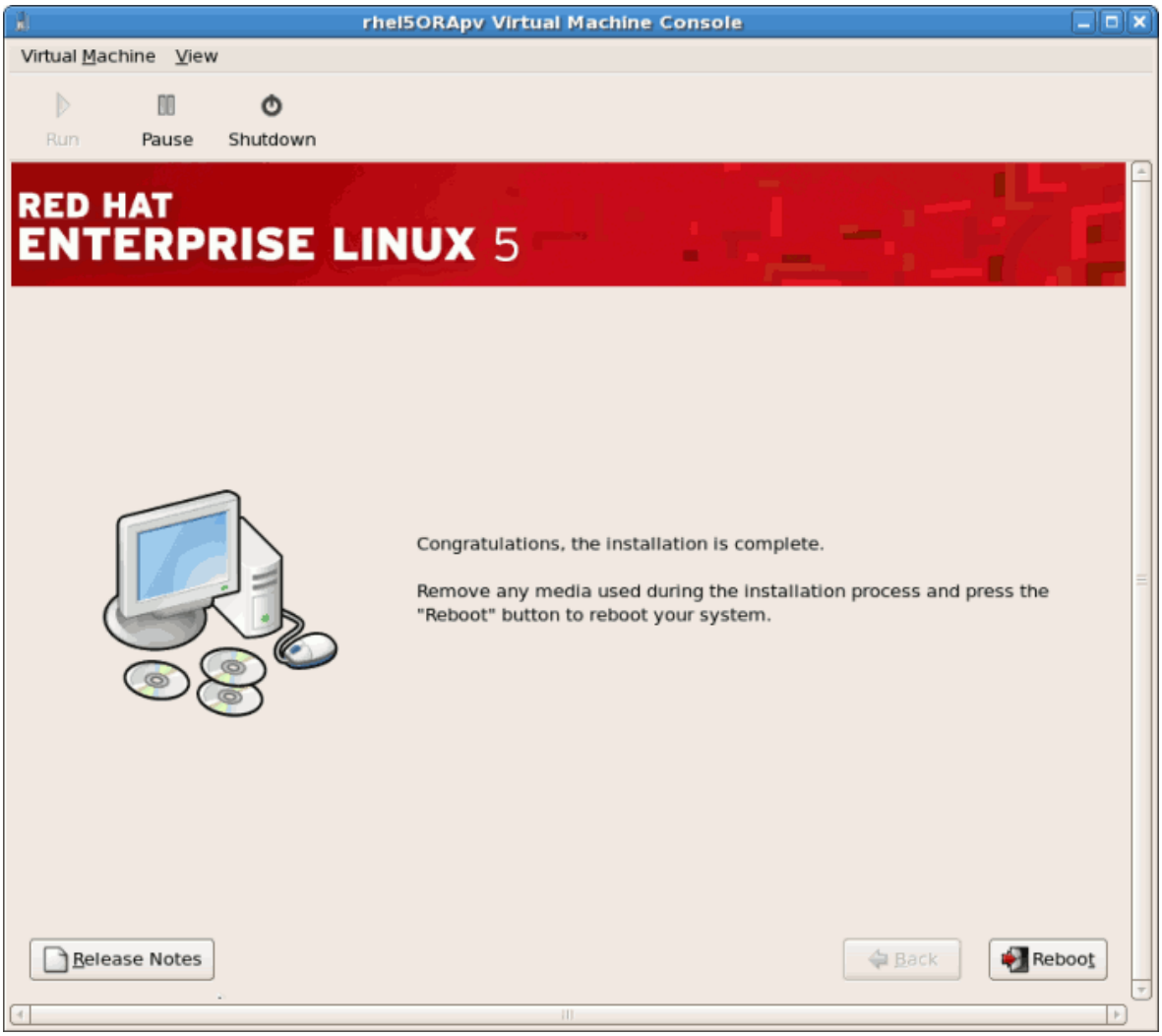
9. Будет выполнена проверка зависимостей.



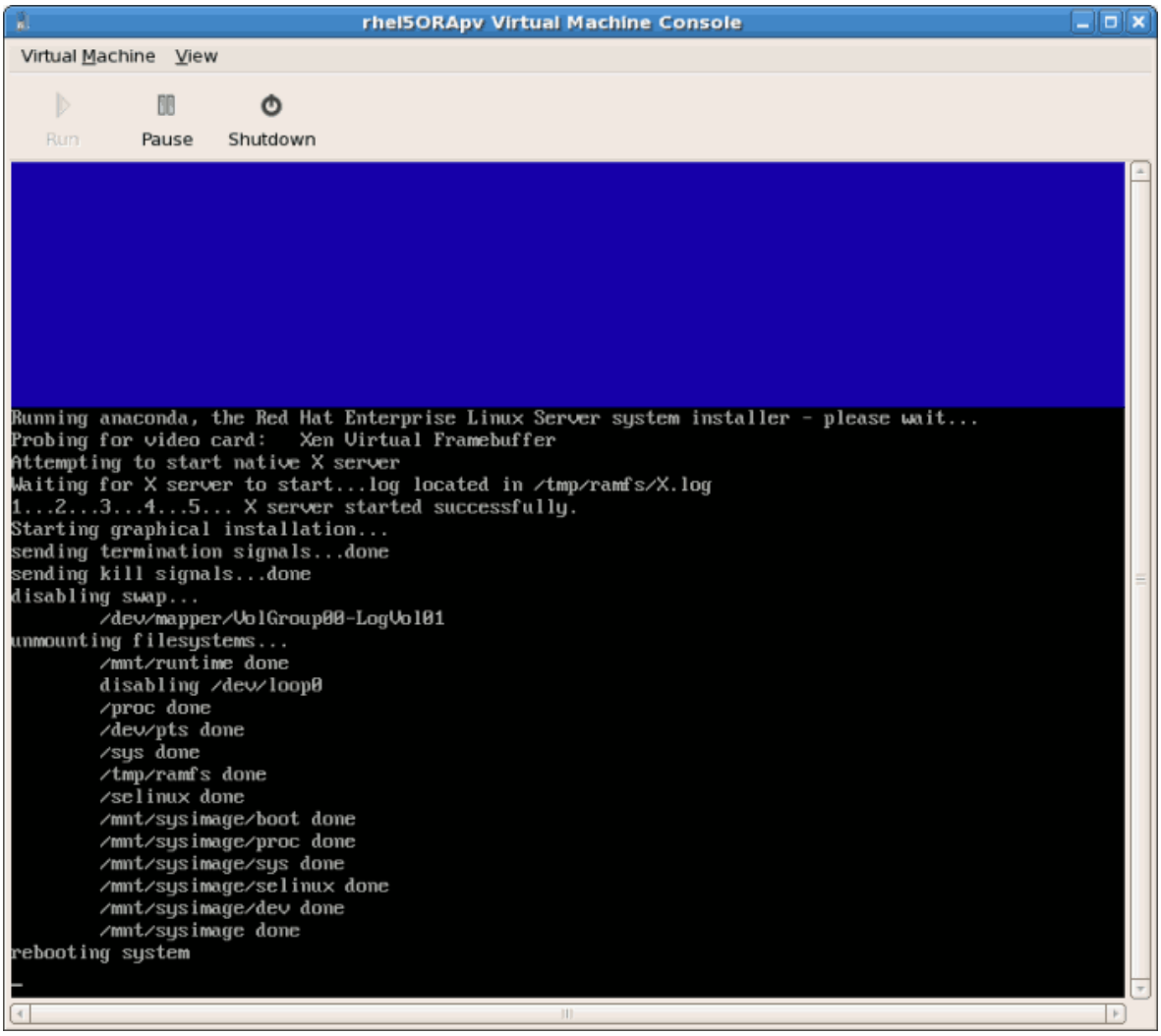
10. Нажмите **Далее**, чтобы приступить к установке.

11. Выбранные пакеты будут установлены автоматически.

12. После завершения установки потребуется перезагрузить гостевую систему.



13. Созданная виртуальная машина не будет перезагружена, а вместо этого завершит работу.



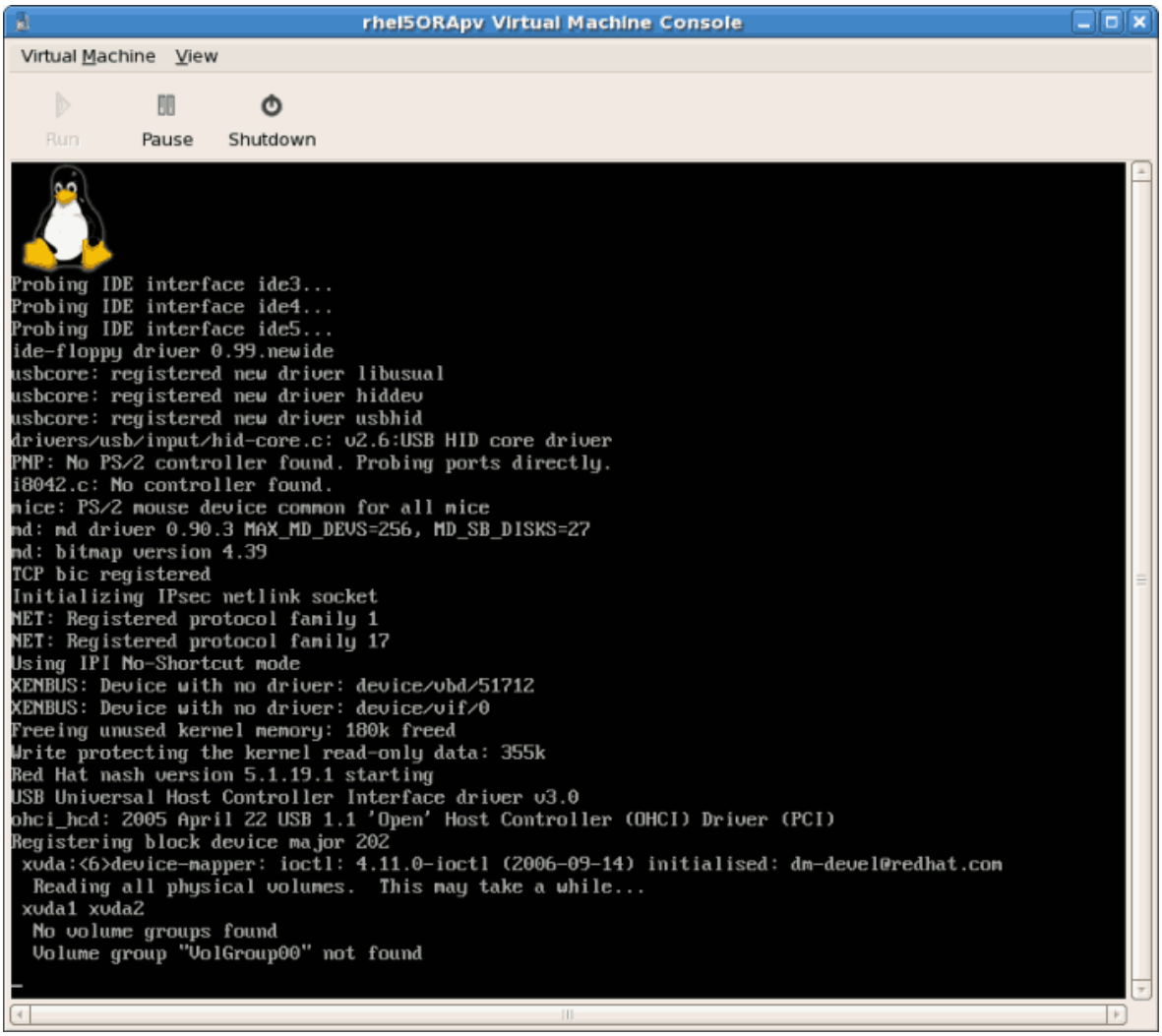
14. Загрузите виртуальную машину. Вы присвоили ей имя в ходе выполнения **`virt-install`** (см. *Раздел 3.1, «Установка Red Hat Enterprise Linux 5 в качестве паравиртуализированного гостя»*). В этом примере используется имя *`rhel5PV`*.

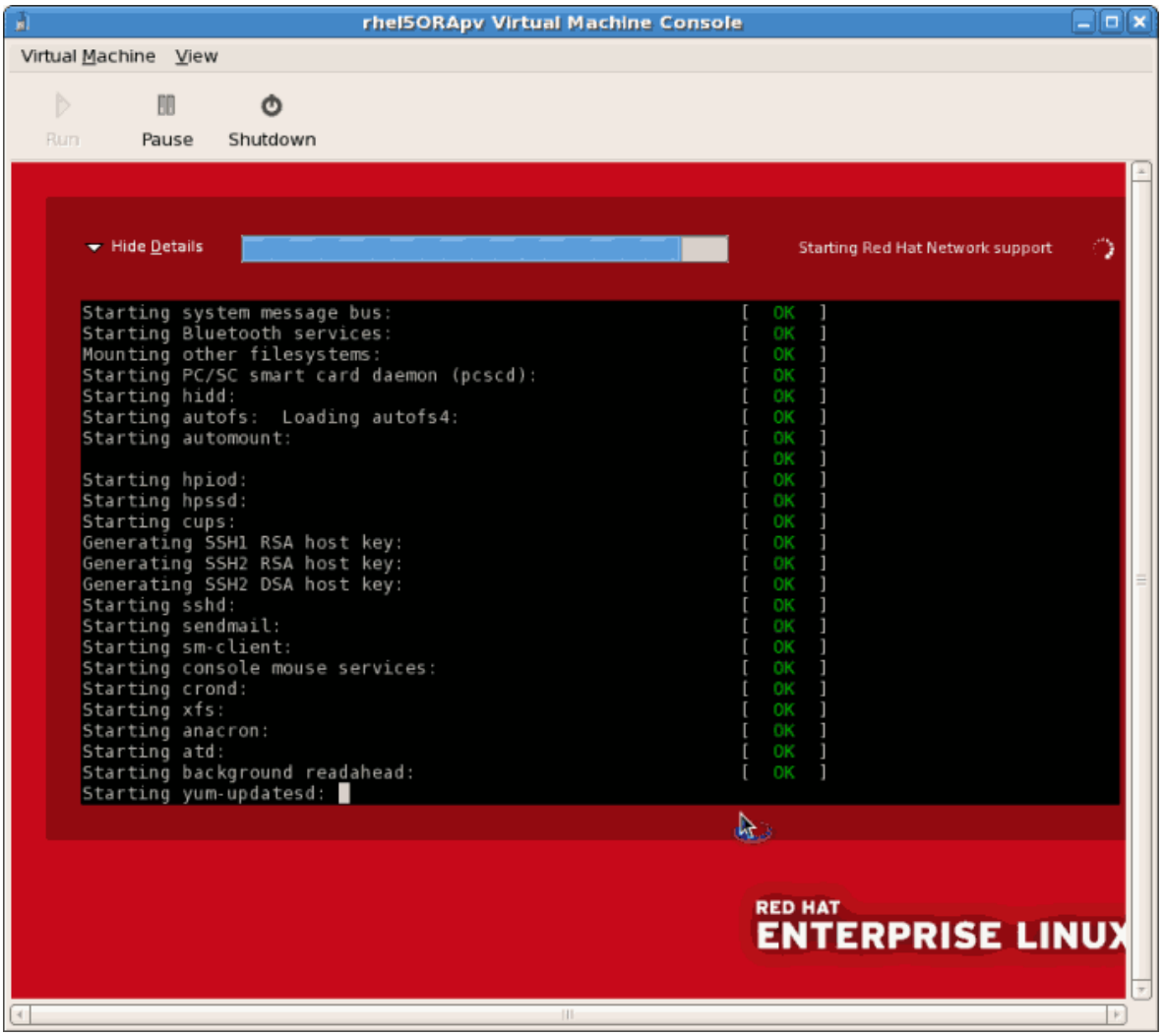
    Для этого выполните команду

    ```
    virsh reboot rhel5PV
    ```

    Или же откройте **`virt-manager`**, выберите имя гостевой системы и нажмите **Открыть**, затем **Запустить**.

    В открывшемся окне **VNC** вы увидите прогресс загрузки.


rhel5ORApv Virtual Machine Console
Virtual Machine View
Run
Pause
Shutdown
Probing IDE interface ide3...
Probing IDE interface ide4...
Probing IDE interface ide5...
ide-floppy driver 0.99.newide
usbcore: registered new driver libusual
usbcore: registered new driver hiddev
usbcore: registered new driver usbhid
drivers/usb/input/hid-core.c: v2.6:USB HID core driver
PNP: No PS/2 controller found. Probing ports directly.
i8042.c: No controller found.
mice: PS/2 mouse device common for all mice
md: md driver 0.90.3 MAX_MD_DEVS=256, MD_SB_DISKS=27
md: bitmap version 4.39
TCP bic registered
Initializing IPsec netlink socket
NET: Registered protocol family 1
NET: Registered protocol family 17
Using IPI No-Shortcut mode
XENBUS: Device with no driver: device/vbd/51712
XENBUS: Device with no driver: device/vif/0
Freeing unused kernel memory: 180k freed
Write protecting the kernel read-only data: 355k
Red Hat nash version 5.1.19.1 starting
USB Universal Host Controller Interface driver v3.0
ohci_hcd: 2005 April 22 USB 1.1 'Open' Host Controller (OHCI) Driver (PCI)
Registering block device major 202
xvda:<6>device-mapper: ioctl: 4.11.0-ioctl (2006-09-14) initialised: dm-devel@redhat.com
Reading all physical volumes. This may take a while...
xvda1 xvda2
No volume groups found
Volume group "VolGroup00" not found



15. При первой загрузке вы увидите экран настройки, который позволит выбрать параметры конфигурации для вашей гостевой системы.

16. Ознакомьтесь с текстом лицензионного соглашения и примите условия.

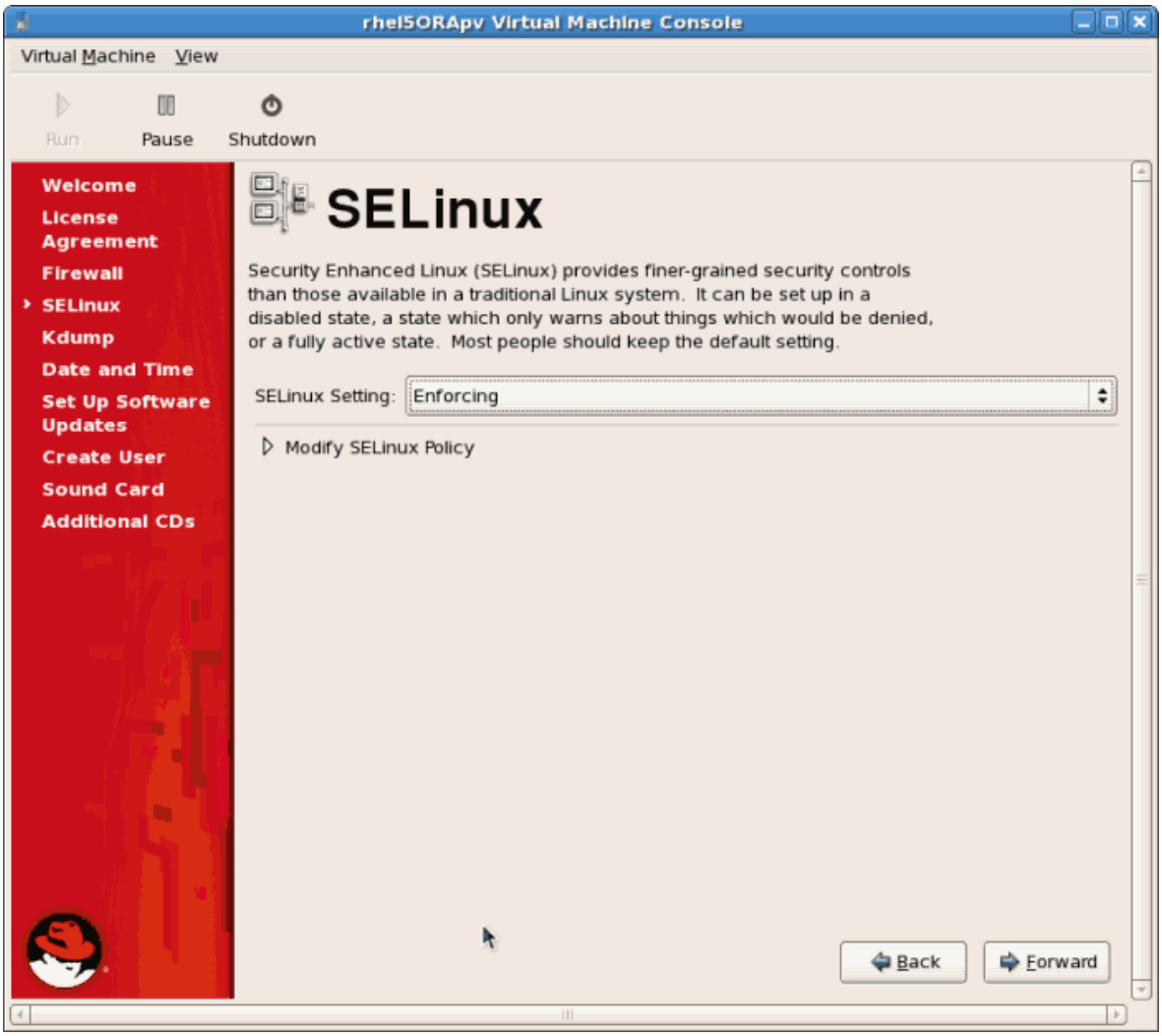
Нажмите **Далее**.

17. Настройте межсетевой экран.

Click **Forward** to continue.

- Если вы решили отключить межсетевой экран, потребуется повторно подтвердить выбор. Нажмите **Да**.

18. Затем настройте SELinux. Настоятельно рекомендуется установить политику SELinux в принудительном режиме.



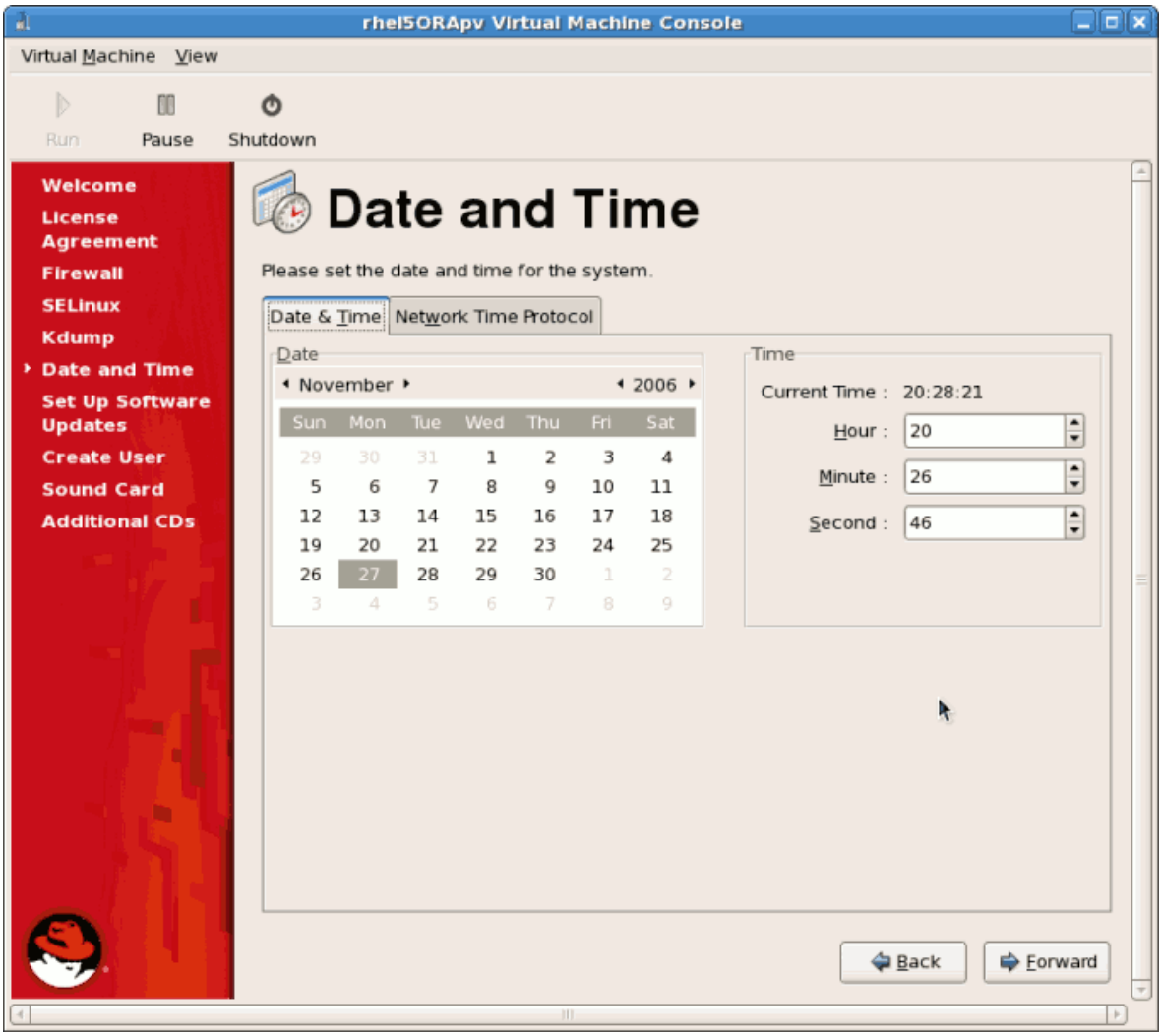
Click **Forward** to continue.

- Если вы отключите SELinux, появится предупреждающее сообщение. Нажмите **Да**.

19. Если требуется, настройте **kdump**.

Click **Forward** to continue.

20. Проверьте дату и время. Если вы установили паравиртуализированную систему, то дата и время должны синхронизироваться с гипервизором.

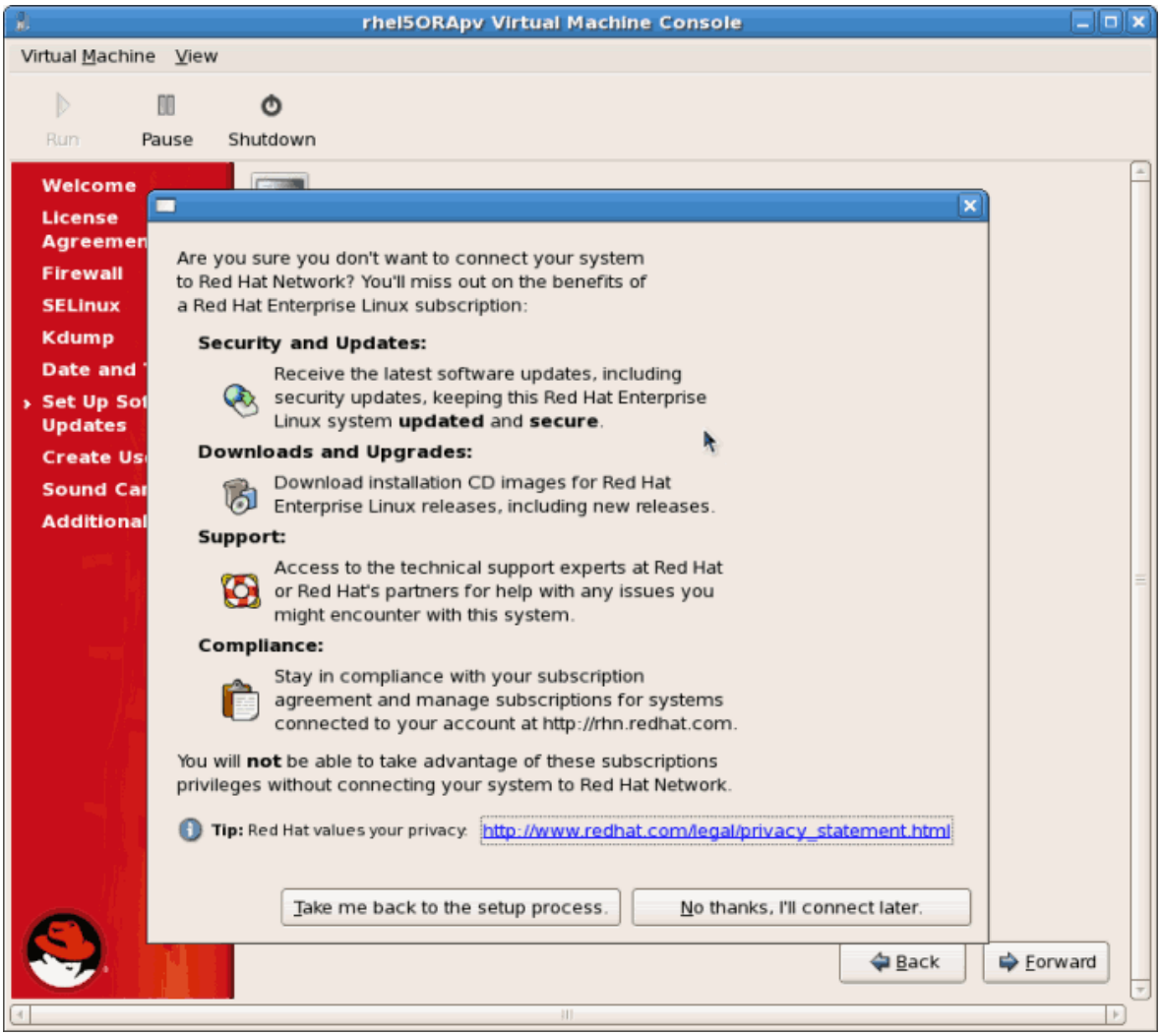


Click **Forward** to continue.

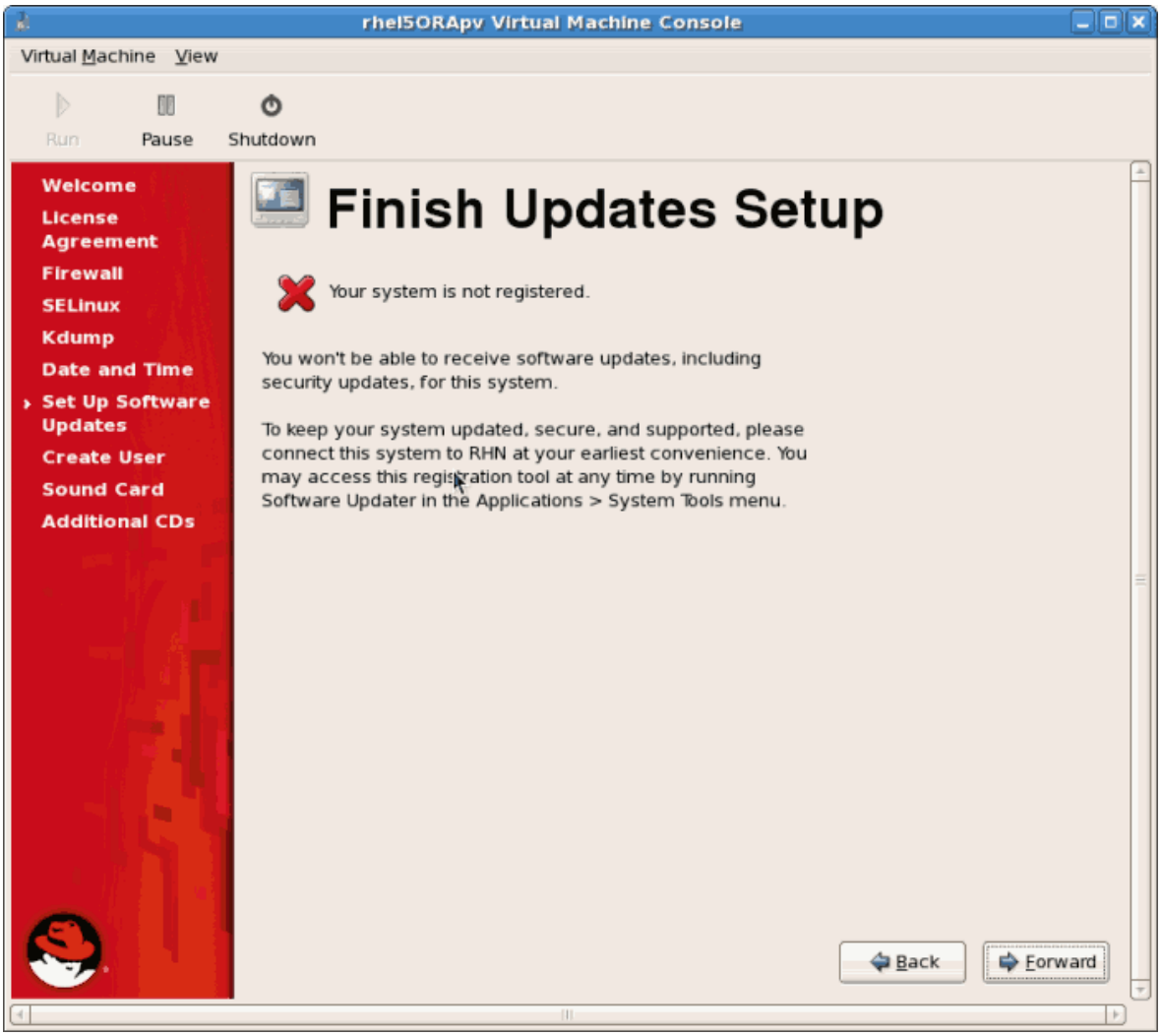
21. Настройте обновления программ. Если у вас есть подписка Fedora Network, систему можно зарегистрировать в RHN.

Click **Forward** to continue.

a. Подтвердите выбор.

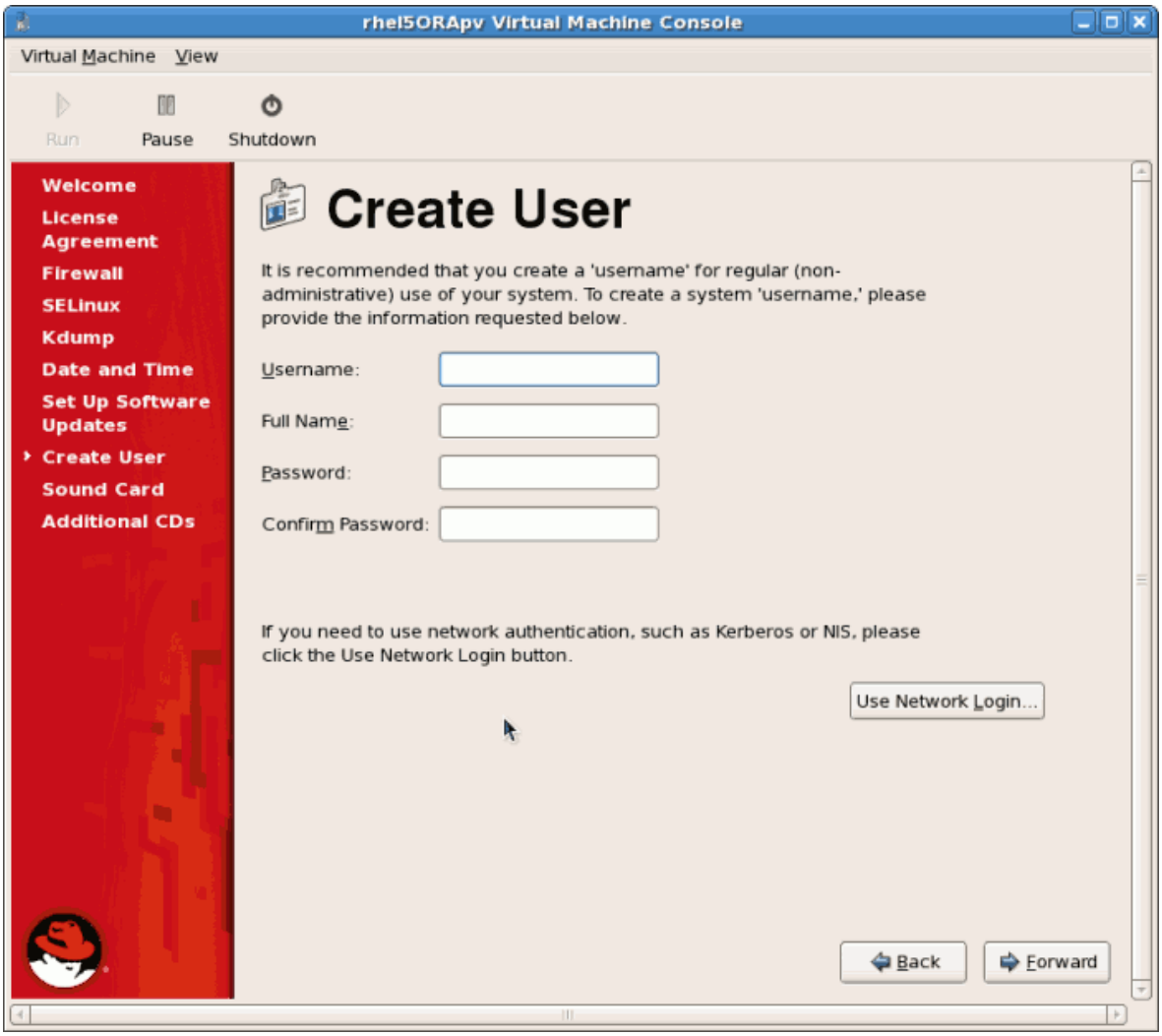


b. Если вы отказались от регистрации в RHN, после завершения настройки появится дополнительный экран.



Нажмите **Далее**.

22. Создайте учетную запись пользователя root. Рекомендуется также создать непривилегированного пользователя.

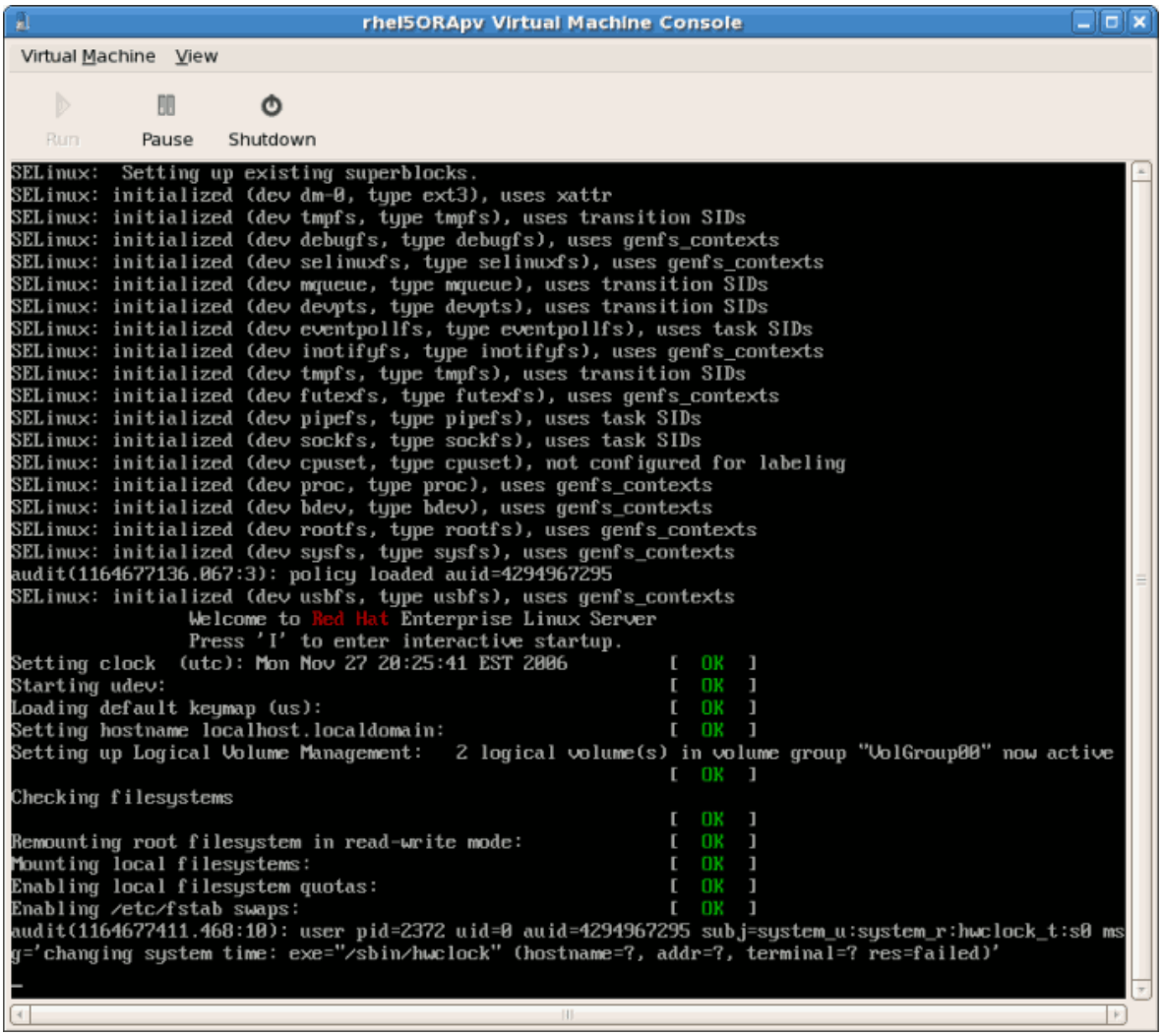


Нажмите **Далее**.

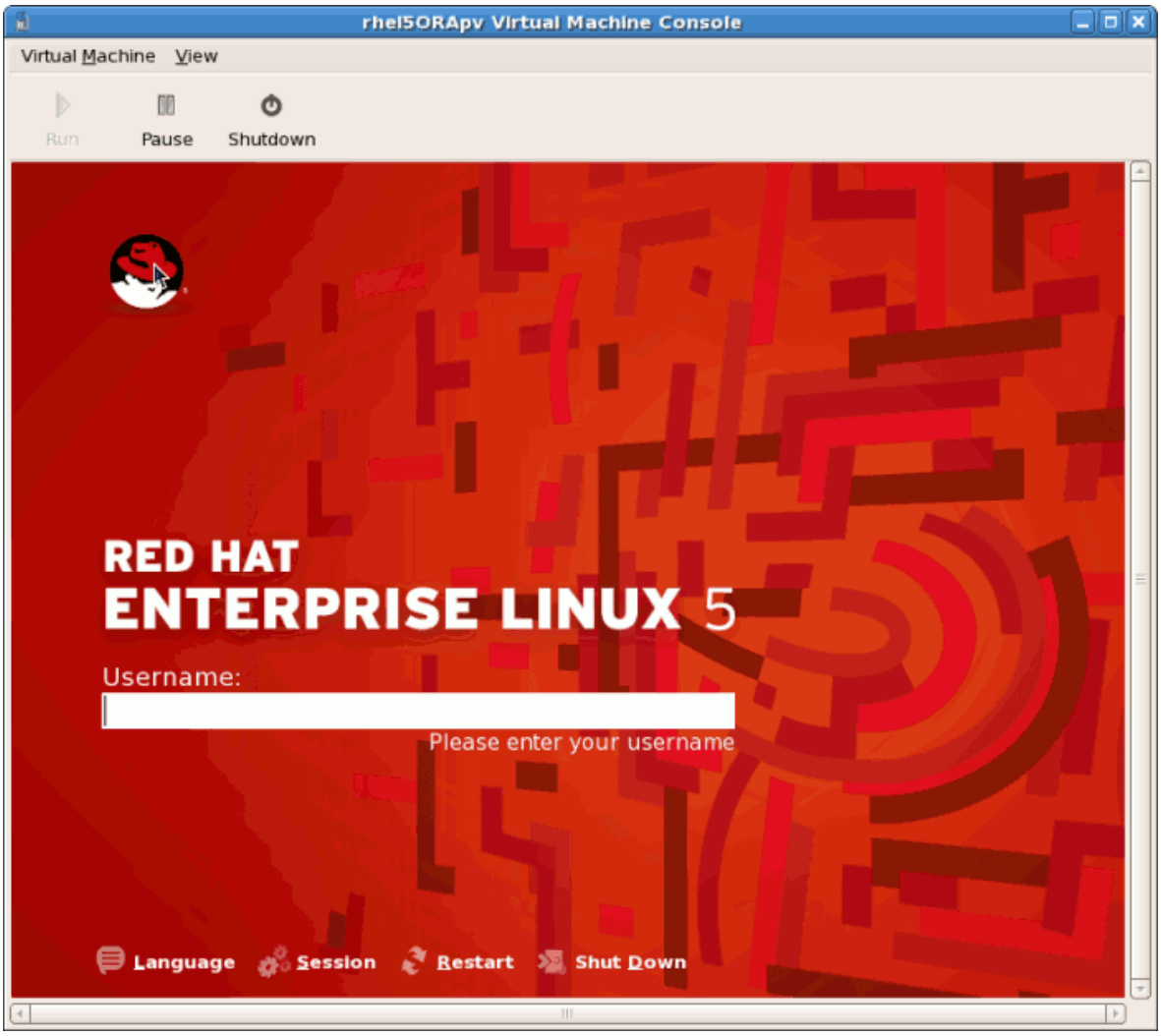
23. Если было обнаружено звуковая карта, настройте ее в следующем окне. Завершив, нажмите **Далее**.

24. Дальше можно выбрать установку дополнительных пакетов с компакт-диска. Обычно не рекомендуется устанавливать дополнительные пакеты на данном этапе, а сделать это позднее с помощью yum. Нажмите кнопку **Готово**.

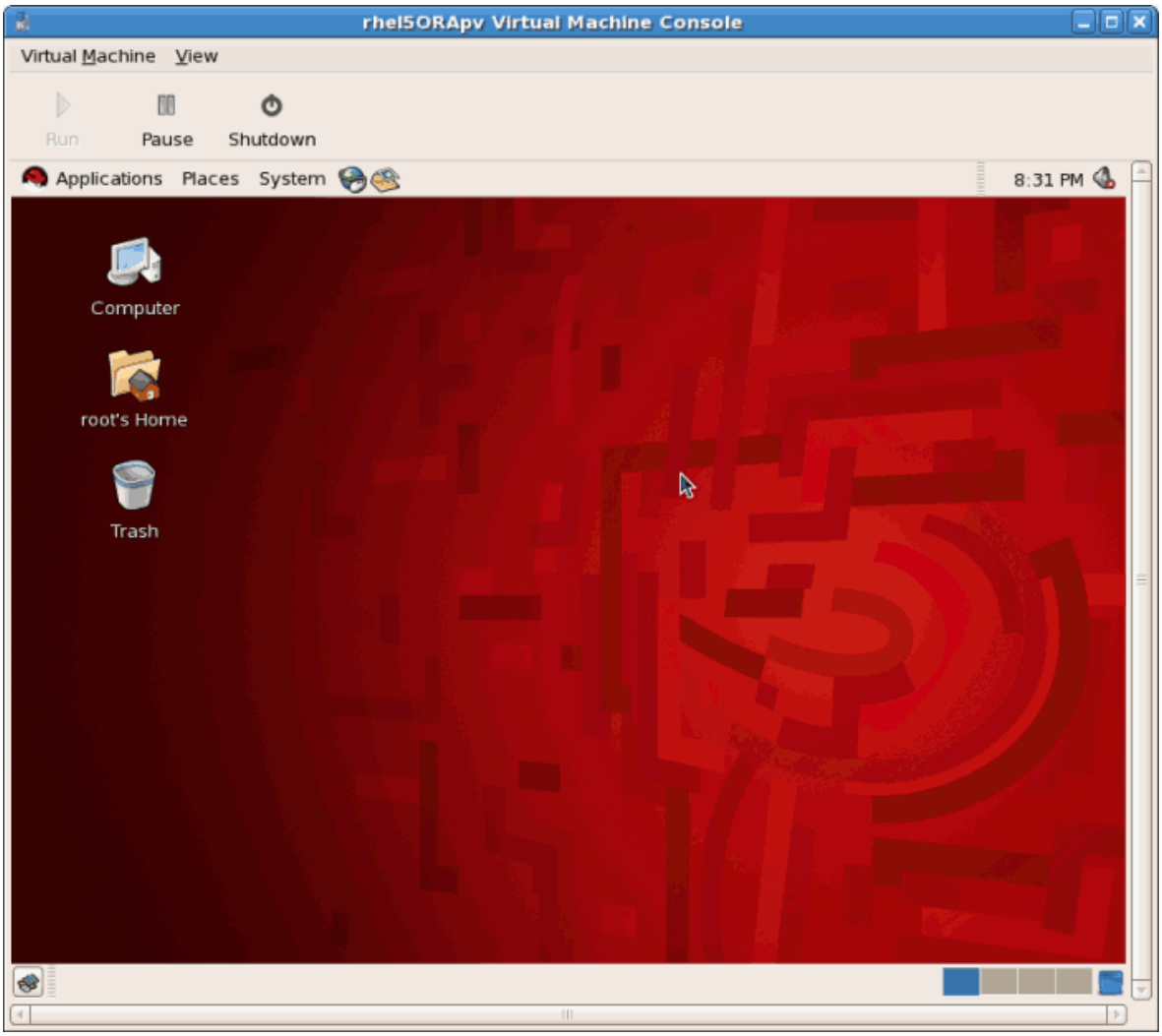
25. Процесс загрузки будет продолжен.



26. Откроется окно приветствия Red Hat Enterprise Linux 5. Войдите в систему, указав имя и пароль только что созданного пользователя.



27. Вы успешно установили паравиртуализированную гостевую систему Red Hat Enterprise Linux 5.



3.2. Установка Red Hat Enterprise Linux в качестве полностью виртуализированного гостя

Дальше будет описан процесс установки полностью виртуализированной гостевой системы Red Hat Enterprise Linux 5.

Процедура 3.3. Создание полностью виртуализированной гостевой системы Red Hat Enterprise Linux 5 с помощью virt-manager

1. **Open virt-manager**

   Start **`virt-manager`**. Launch the **Virtual Machine Manager** application from the **Applications** menu and **System Tools** submenu. Alternatively, run the **`virt-manager`** command as root.

2. **Select the hypervisor**

   Select the hypervisor. If installed, select Xen or KVM. For this example, select KVM. Note that presently KVM is named `qemu`.

   Подключитесь к гипервизору. Откройте меню **Файл** (File) и выберите **Открыть соединение...** (Add Connection..) (см. *Раздел 16.1, «Окно соединений»*).

Как только будет определено соединение для гипервизора, кнопка создания новой виртуальной машины станет доступна.

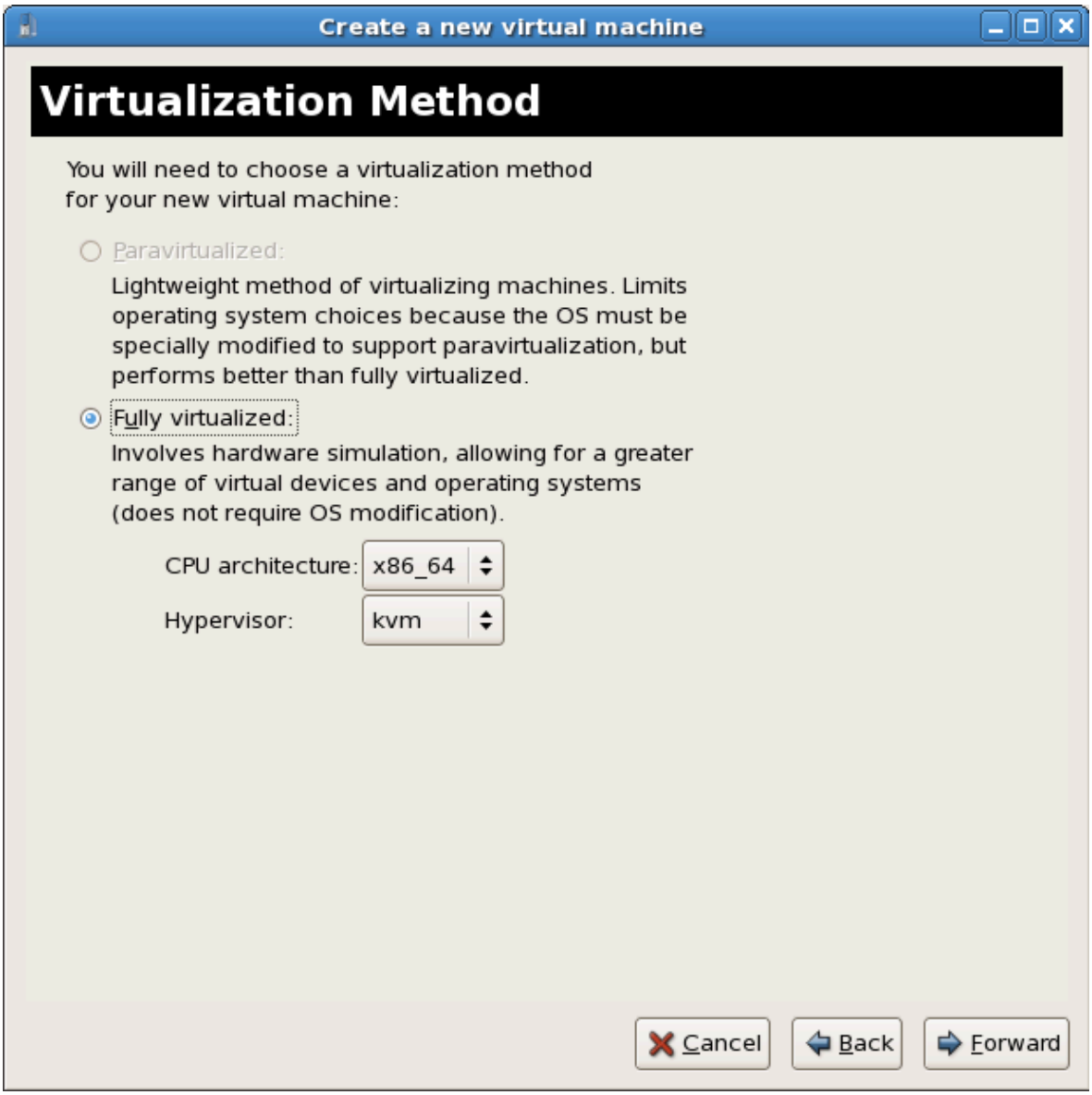
3. **Start the new virtual machine wizard**
   Pressing the **New** button starts the virtual machine creation wizard.

   Press **Forward** to continue.

4. **Name the virtual machine**
   Введите имя для виртуализированного гостя. Обратите внимание, что имя не может содержать пробелы и знаки пунктуации.

Нажмите **Далее**.

5. **Choose a virtualization method**
   На этом этапе можно выбрать тип виртуализации. При этом в качестве гипервизора необходимо указать тот гипервизор, который был выбран раньше (*Шаг 4*). В этом примере будет снова выбран KVM.



Нажмите **Далее** (Forward).

6. **Select the installation method**
   Выберите **Локальный установочный носитель** (Local install media), чтобы указать путь к ISO-образу или устройству чтения дисков, **Сетевое дерево установки** (Network install tree), чтобы указать ссылку на сервер HTTP, FTP, NFS, или же выберите **Сетевая загрузка** (Network boot) для выполнения установки с сервера PXE.

   В выпадающем списке **Тип ОС** (OS Type) выберите **Linux**, а в качестве самой системы — **Red Hat Enterprise Linux 5**.

Нажмите **Далее** (Forward).

7. **Locate installation media**
   Вы можете указать путь к ISO-образу или устройству чтения дисков. В приведенном примере будет указан путь к образу установочного DVD-диска Red Hat Enterprise Linux 5.

   a. Press the **Browse** button.

   b. Выберите файл ISO. Нажмите **Открыть** для подтверждения выбора.

   c. Выбранный файл будет служить источником установки.

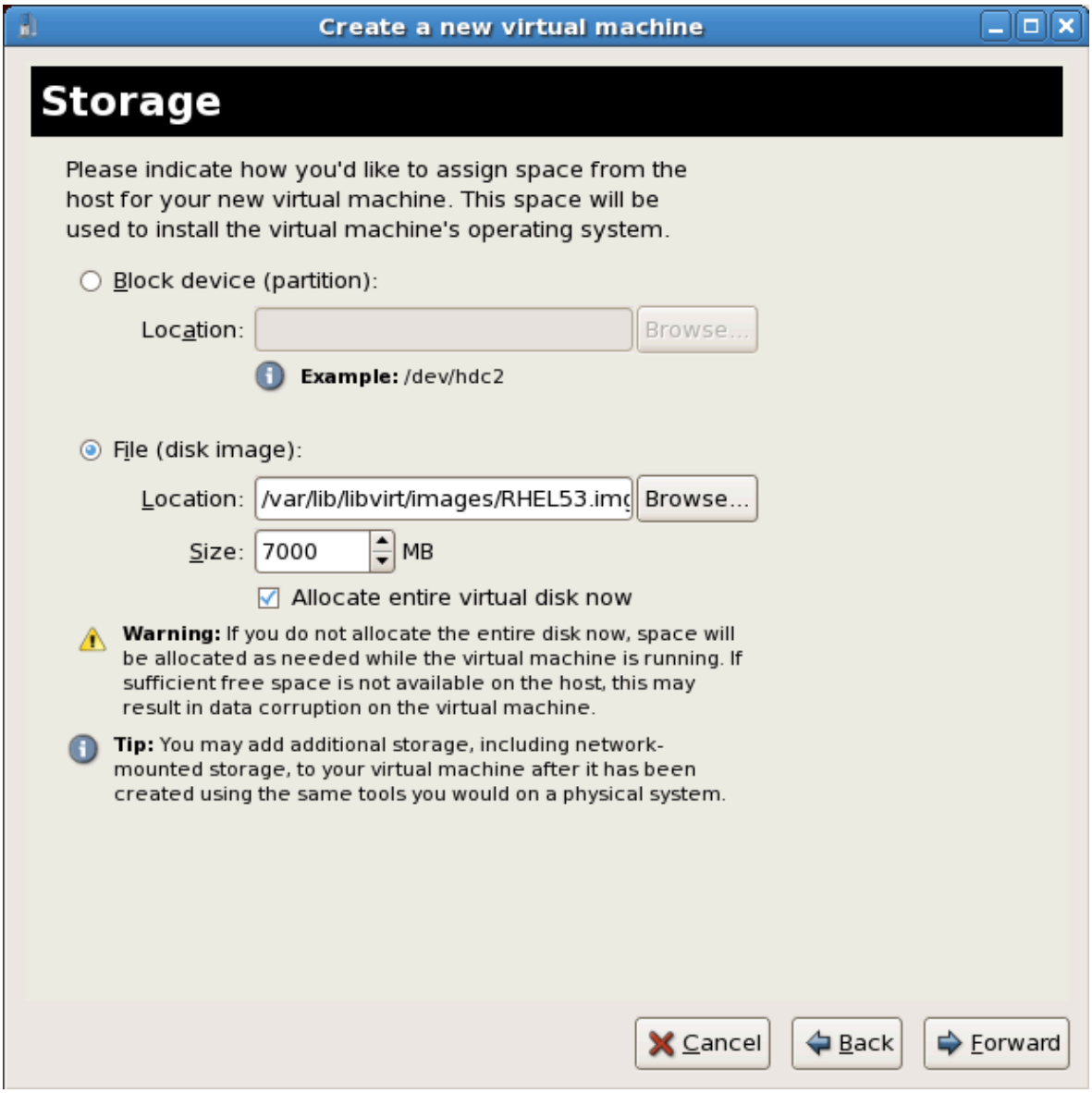
Нажмите **Далее**.

Image files and SELinux

Для хранения ISO-файлов и образов хранилищ рекомендуется использовать каталог **`/var/lib/libvirt/images/`**, так как другие каталоги могут требовать дополнительной настройки SELinux (см. *Раздел 7.1, «Виртуализация и SELinux»*).

8. **Storage setup**

   Можно выбрать физическое блочное устройство или файловый образ. Образы должны храниться в каталоге **`/var/lib/libvirt/images/`**. Убедитесь, что виртуализированному гостю предоставлено достаточно пространства.



Нажмите **Далее**.

Замечание о миграции

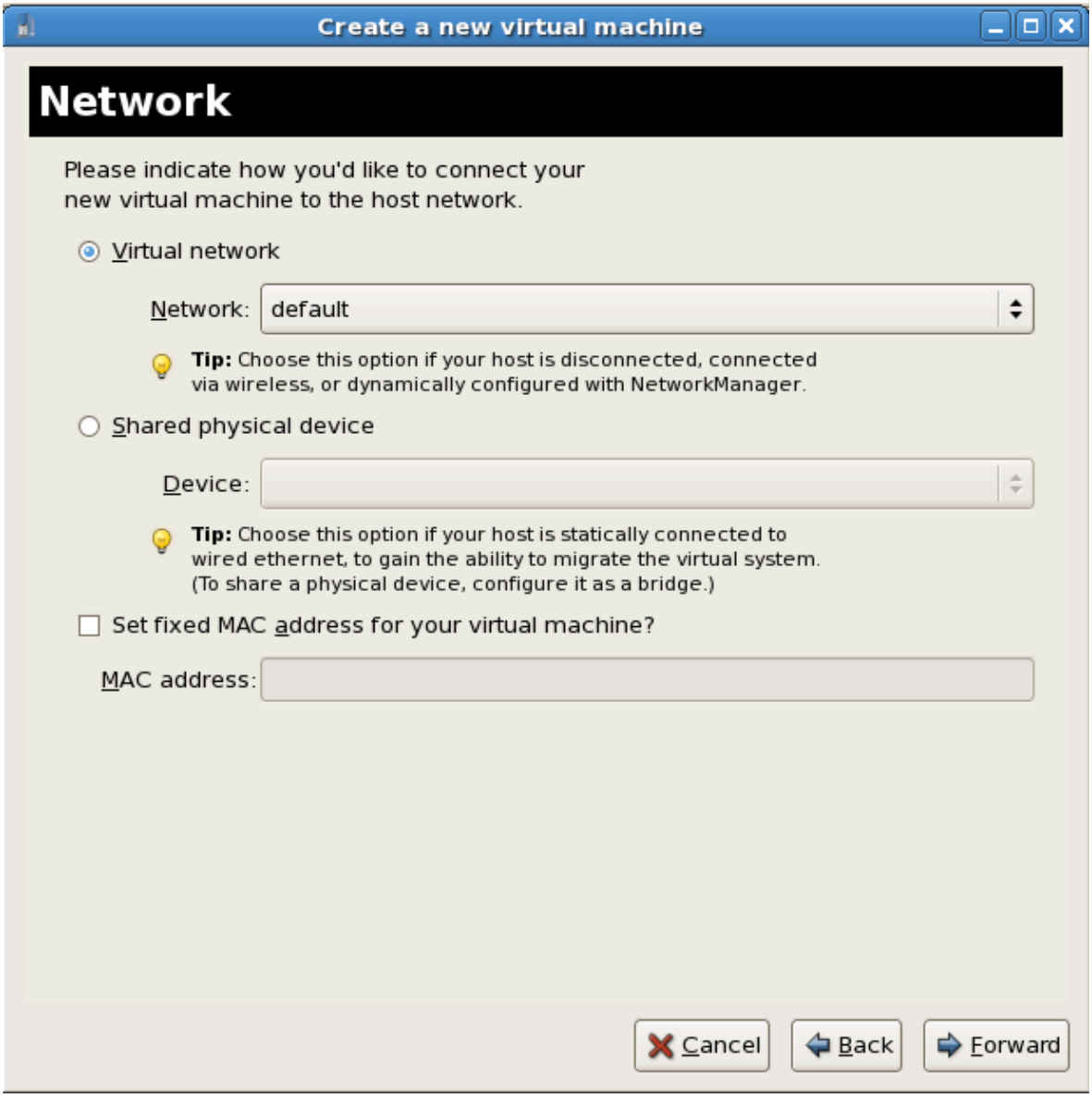
Для выполнения автономной и живой миграции необходимо, чтобы гостевые системы были установлены в общем сетевом хранилище. *Глава 5, Виртуализация и общие хранилища данных* содержит информацию о его настройке.

9. **Network setup**

   Select either **Virtual network** or **Shared physical device**.

   The virtual network option uses Network Address Translation (NAT) to share the default network device with the virtualized guest. Use the virtual network option for wireless networks.

   The shared physical device option uses a network bond to give the virtualized guest full access to a network device.



Press **Forward** to continue.

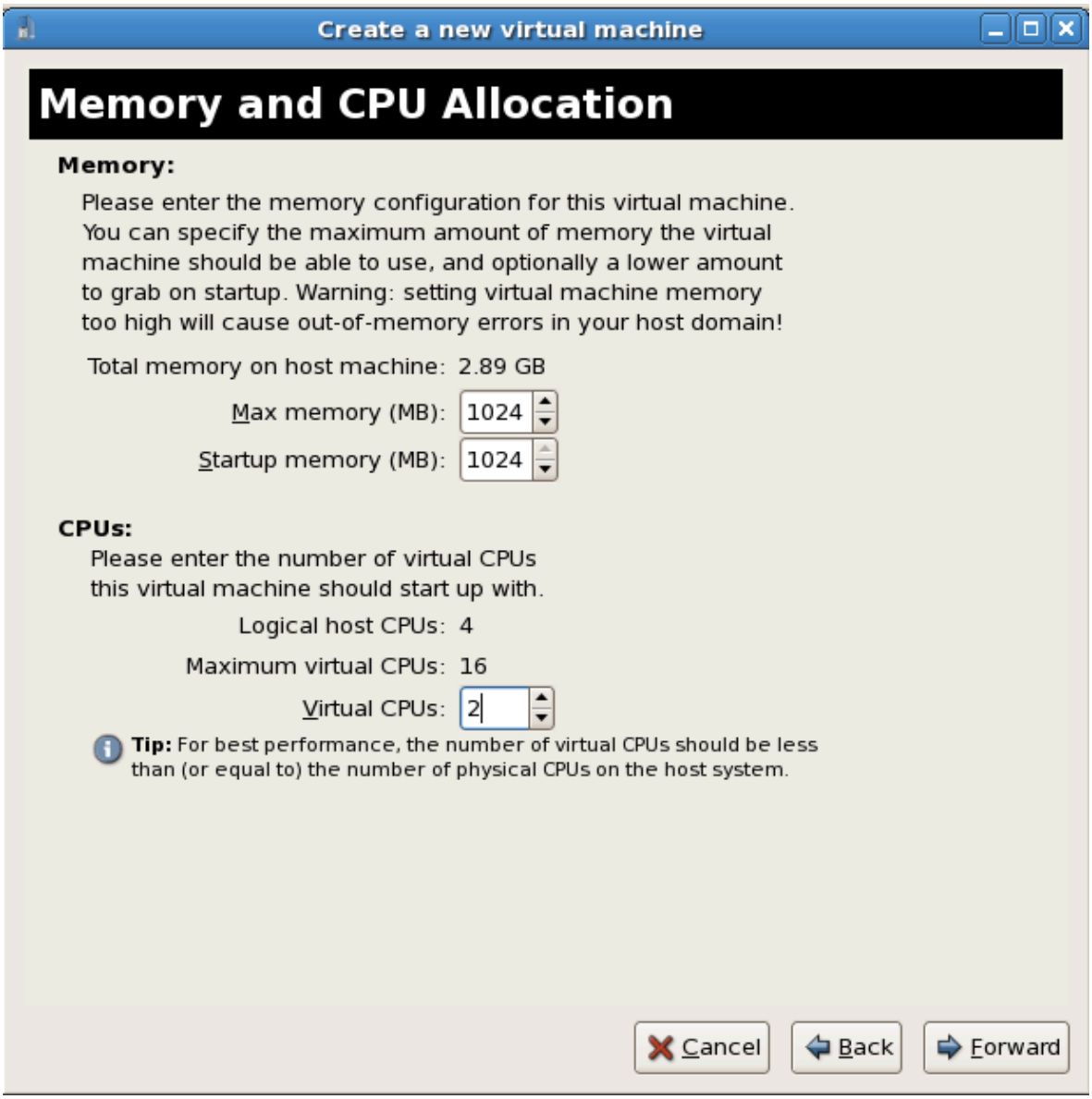
10. **Memory and CPU allocation**

    The Allocate memory and CPU window displays. Choose appropriate values for the virtualized CPUs and RAM allocation. These values affect the host's and guest's performance.

    Virtualized guests require sufficient physical memory (RAM) to run efficiently and effectively. Choose a memory value which suits your guest operating system and application requirements. Windows Server 2008. Remember, guests use physical RAM. Running too many guests or leaving insufficient memory for the host system results in significant usage of virtual memory and swapping. Virtual memory is significantly slower causing degraded system performance and responsiveness. Ensure to allocate sufficient memory for all guests and the host to operate effectively.

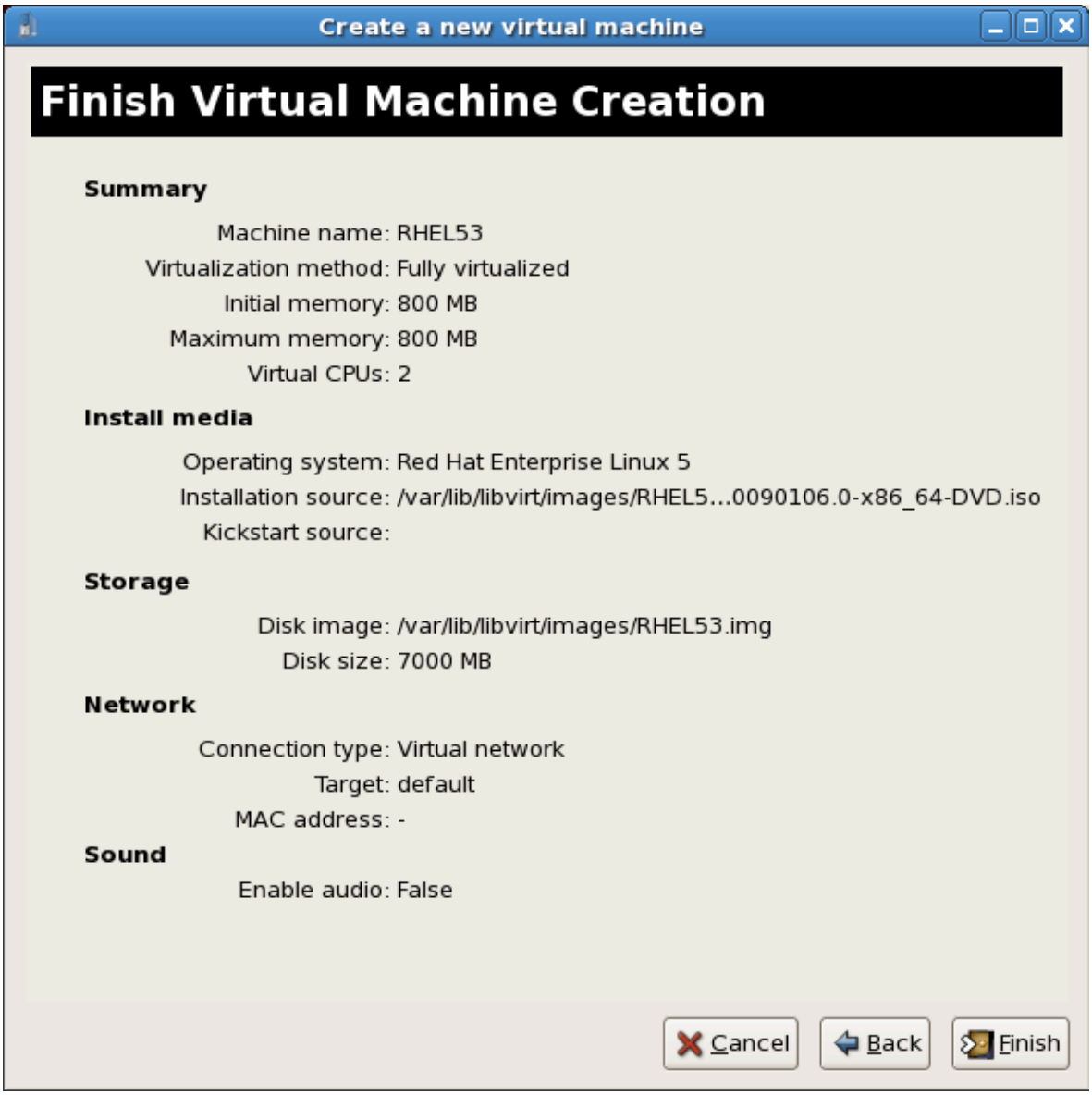
    Assign sufficient virtual CPUs for the virtualized guest. If the guest runs a multithreaded application assign the number of virtualized CPUs it requires to run most efficiently. Do not assign more virtual CPUs than there are physical processors (or hyper-threads) available on

the host system. It is possible to over allocate virtual processors, however, over allocating has a significant, negative affect on guest and host performance due to processor context switching overheads.



Press **Forward** to continue.

11. **Verify and start guest installation**
    В следующем окне проверьте настройки.



Нажмите **Завершить**, чтобы начать установку гостевой системы.

12. **Установка Linux**
    Дождитесь завершения процесса установки Red Hat Enterprise Linux 5 (сам процесс подробно рассмотрен в *руководстве по установке* на *странице документации Red Hat*[1]).

Вы успешно установили полностью виртуализированную гостевую систему Red Hat Enterprise Linux 5.

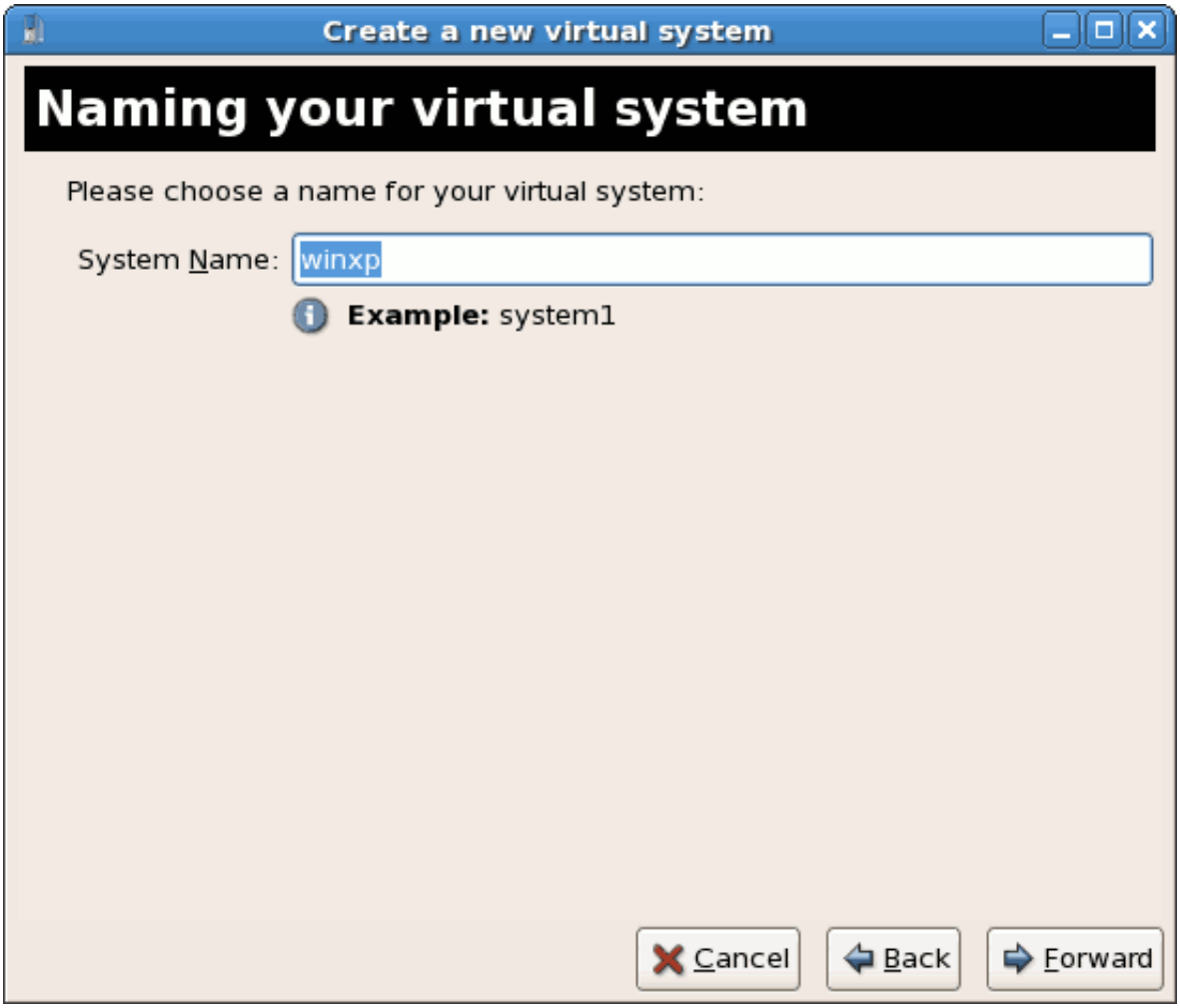
3.3. Установка Windows XP в качестве полностью виртуализированного гостя

Дальше будет описан процесс установки Windows XP в качестве полностью виртуализированного гостя на узле Linux.

Прежде чем приступить к установке, убедитесь, что у вас есть права доступа root.

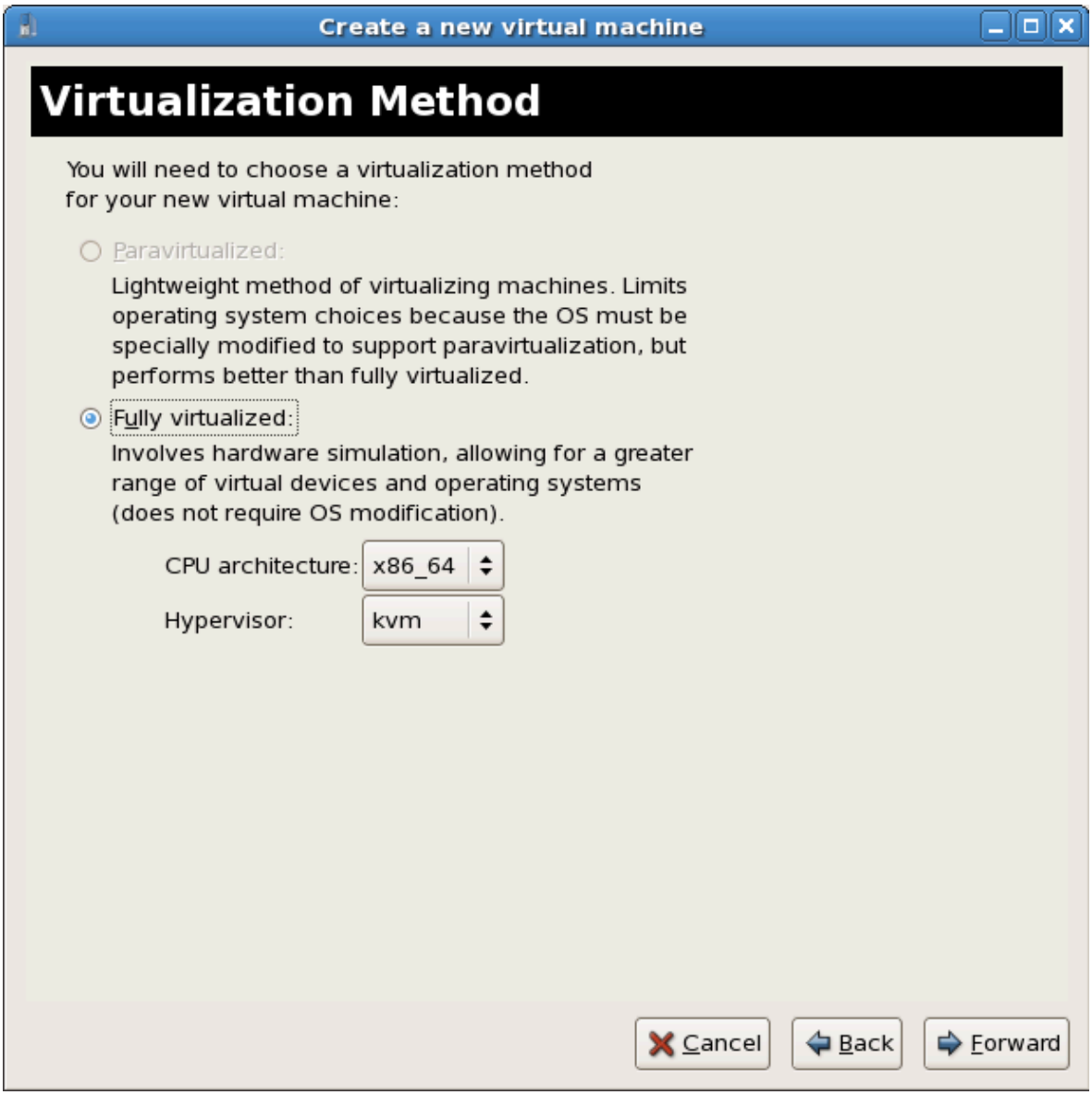
1. **Starting virt-manager**
   Чтобы начать сессию менеджера виртуальных машин, выберите соответствующий ему пункт в меню **Приложения > Система**. В открывшемся окне выберите **Файл > Открыть соединение** (File -> Open Connection) и нажмите кнопку **Создать** (New).

2. **Выберите имя для виртуальной системы**
   Введите имя новой системы и нажмите кнопку продолжения.



3. **Выберите тип виртуализации**
   В качестве гипервизора необходимо указать тот гипервизор, который был выбран раньше (*Шаг 1*). В этом примере будет выбран KVM.

   Напомним, Windows можно установить только в качестве полностью виртуализированного гостя.



4. **Выберите способ установки**
   В этом окне можно выбрать способ установки и тип операционной системы.

   Если вы планируете установить систему с CD-ROM или DVD, выберите устройство с установочным диском Windows. Если же вы используете ISO-файл, укажите путь к установочному образу Windows.

   В выпадающем списке **Тип ОС** выберите **Windows**, а в качестве самой системы укажите **Microsoft Windows XP**.

   В этой главе PXE-установка не рассматривается.

Press **Forward** to continue.

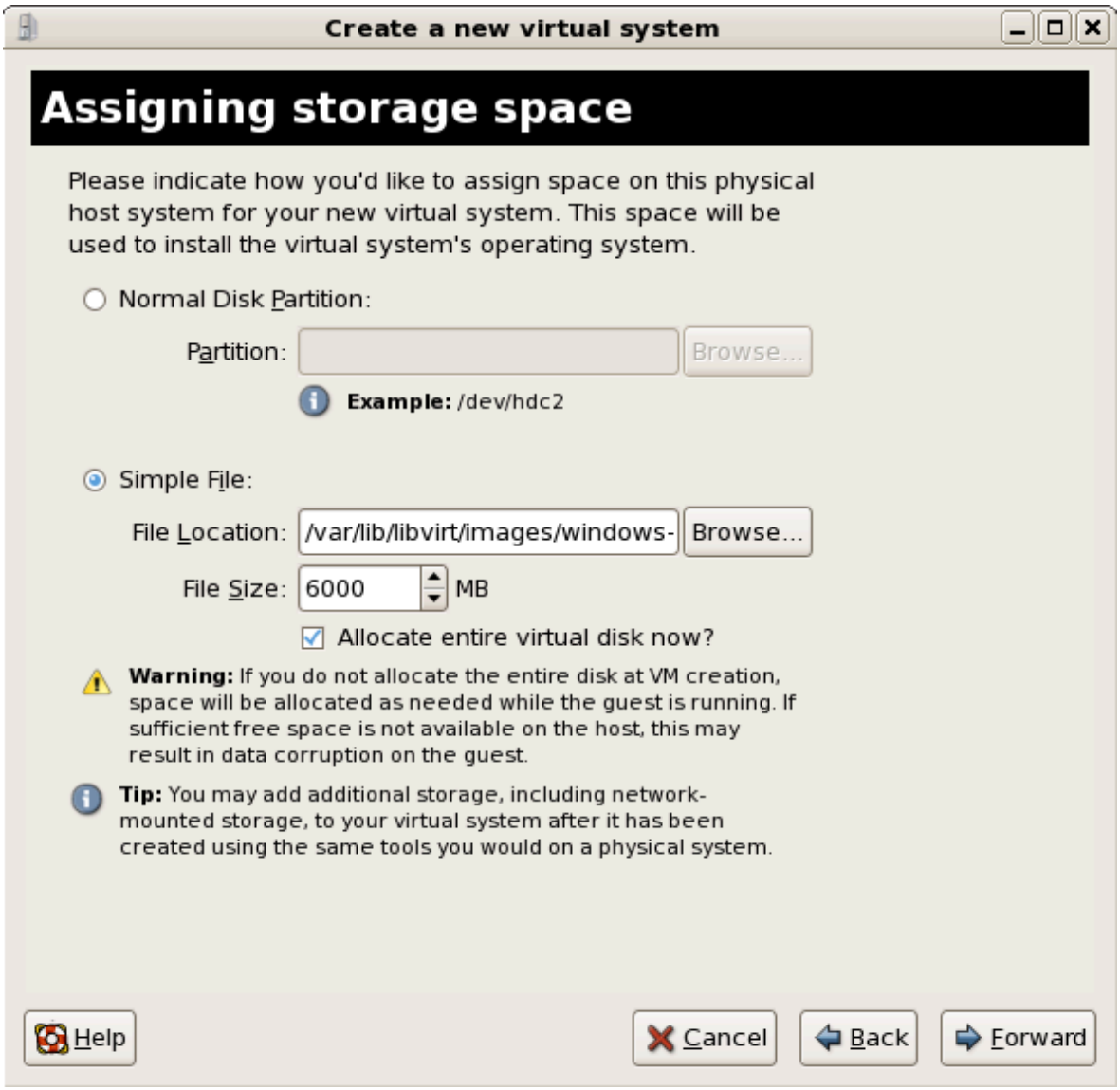
> **Image files and SELinux**
> Для хранения ISO-файлов и образов хранилищ рекомендуется использовать каталог **`/var/lib/libvirt/images/`**, так как другие каталоги могут требовать дополнительной настройки SELinux (см. *Раздел 7.1, «Виртуализация и SELinux»*).

5. The **Assigning storage space** window displays. Choose a disk partition, LUN or create a file based image for the guest storage.

   Файловые образы гостевых систем традиционно хранятся в каталоге **`/var/lib/libvirt/images/`**. SELinux запрещает использование других каталогов. *Раздел 7.1, «Виртуализация и SELinux»* содержит информацию, которая может помочь, если политика SELinux работает в строгом режиме.

Your guest storage image should be larger than the size of the installation, any additional packages and applications, and the size of the guests swap file. The installation process will choose the size of the guest's swap file based on size of the RAM allocated to the guest.

Allocate extra space if the guest needs additional space for applications or other data. For example, web servers require additional space for log files.



Choose the appropriate size for the guest on your selected storage type and click the **Forward** button.

Замечание

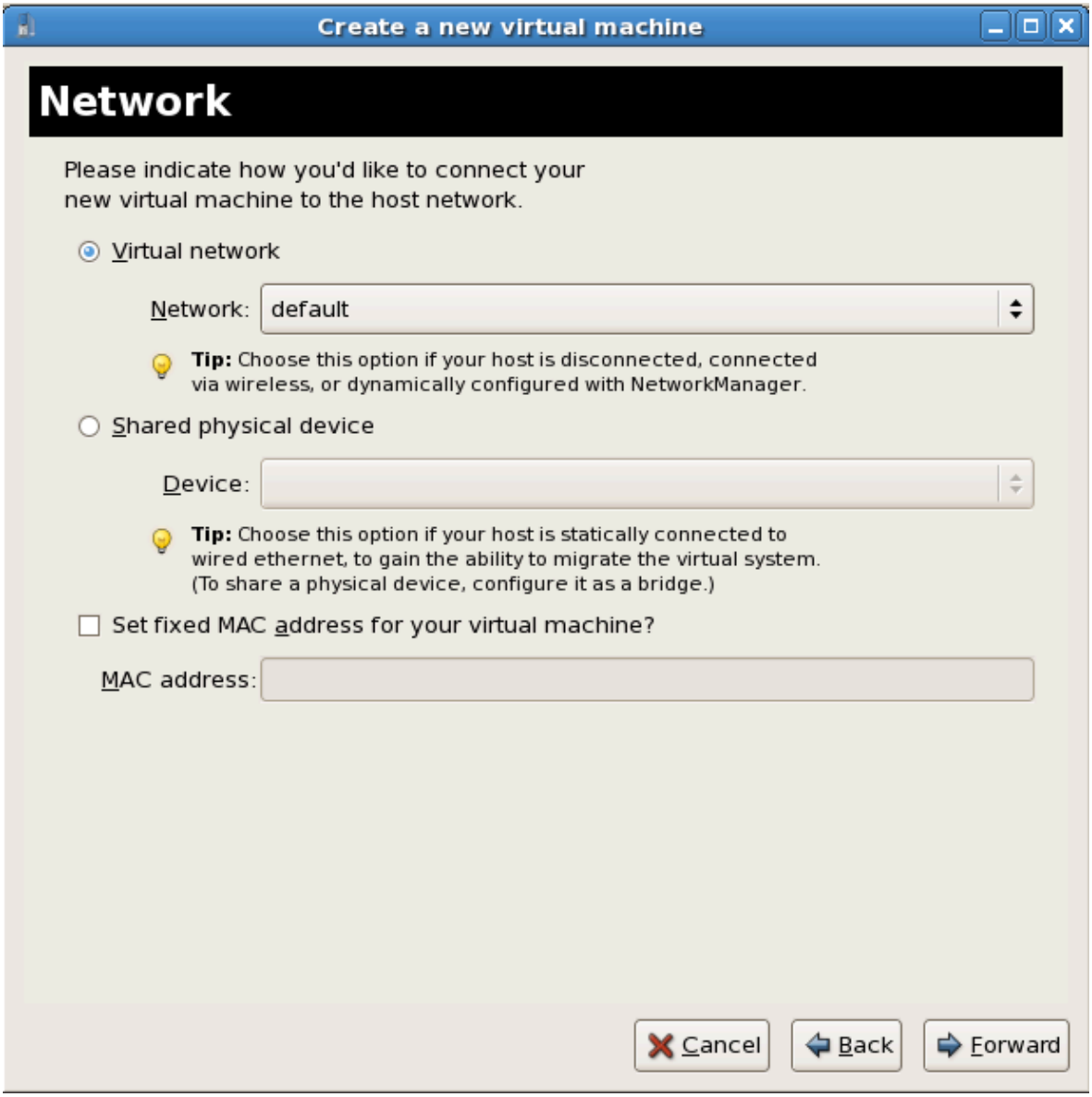
Для хранения образов виртуальных машин рекомендуется использовать стандартный каталог `/var/lib/libvirt/images/` . Если же вы хотите изменить каталог (например, на `/images/`), прежде чем приступить к установке,

потребуется его добавить в политику SELinux. Позднее будет описано, как изменить политику SELinux.

6. **Network setup**
   Select either **Virtual network** or **Shared physical device**.

   The virtual network option uses Network Address Translation (NAT) to share the default network device with the virtualized guest. Use the virtual network option for wireless networks.

   The shared physical device option uses a network bond to give the virtualized guest full access to a network device.

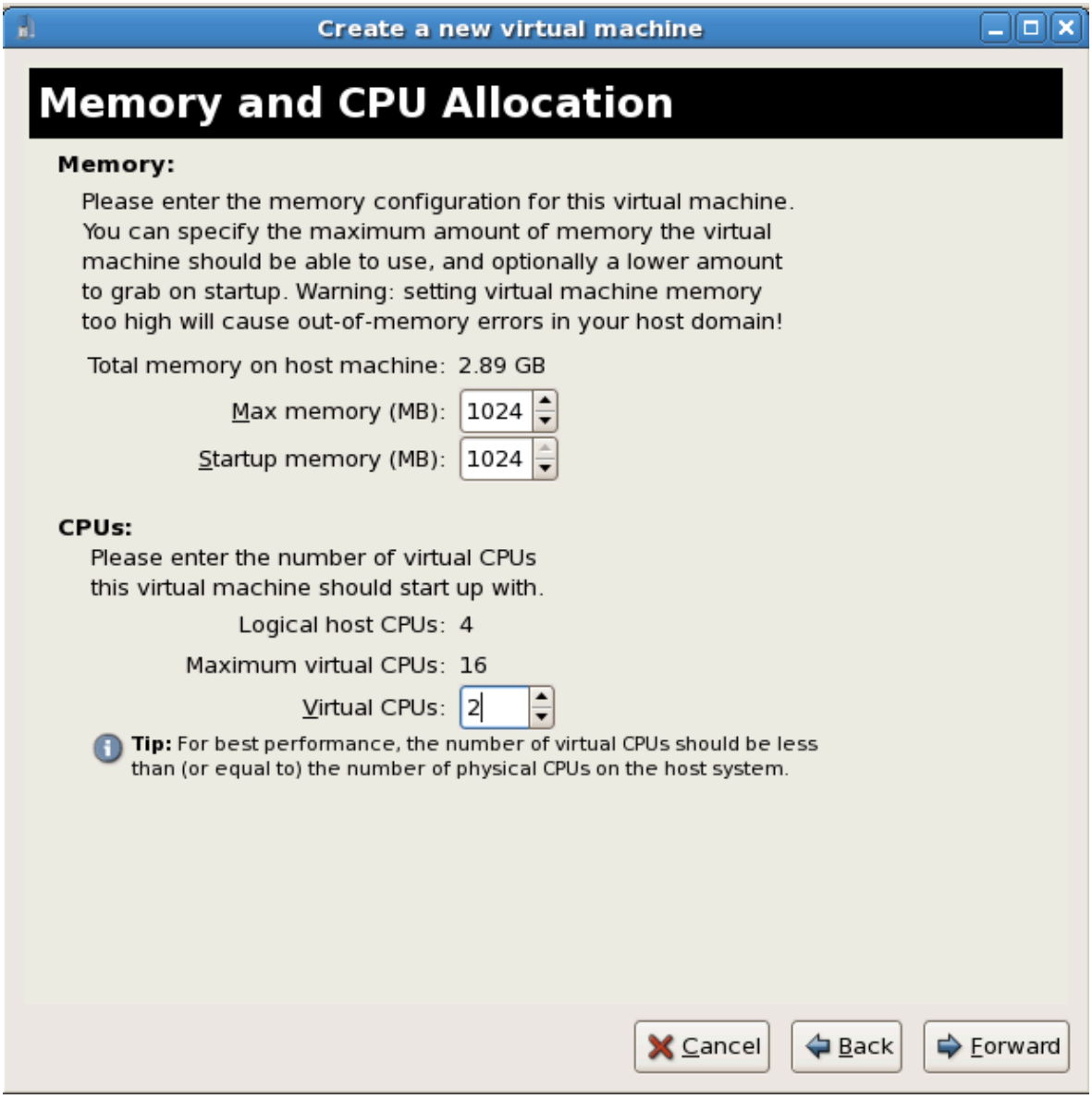


   Press **Forward** to continue.

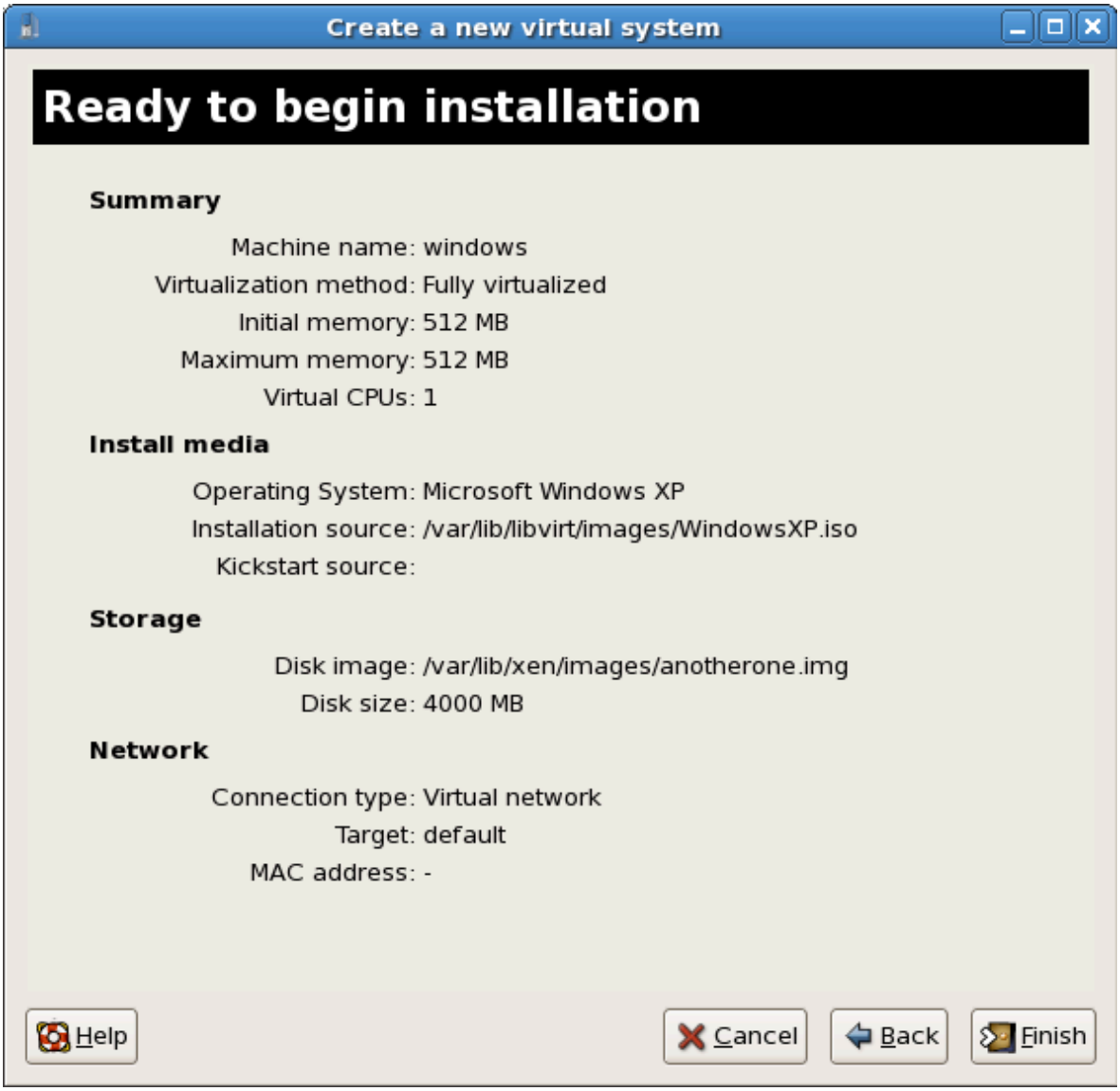
7. The Allocate memory and CPU window displays. Choose appropriate values for the virtualized CPUs and RAM allocation. These values affect the host's and guest's performance.

Виртуальной машине понадобится достаточный для ее работы объем оперативной памяти (как минимум 512 Мбайт). Стоит помнить, что они используют физическую память. Выполнение слишком большого числа гостей или предоставление размещающей системе недостаточного объема памяти может привести к повышенному использованию виртуальной памяти и области подкачки. Как известно, виртуальная память значительно медленнее физической, как следствие, работа системы существенно замедлится. Этого можно избежать, выделив гостевым системам достаточный объем памяти.

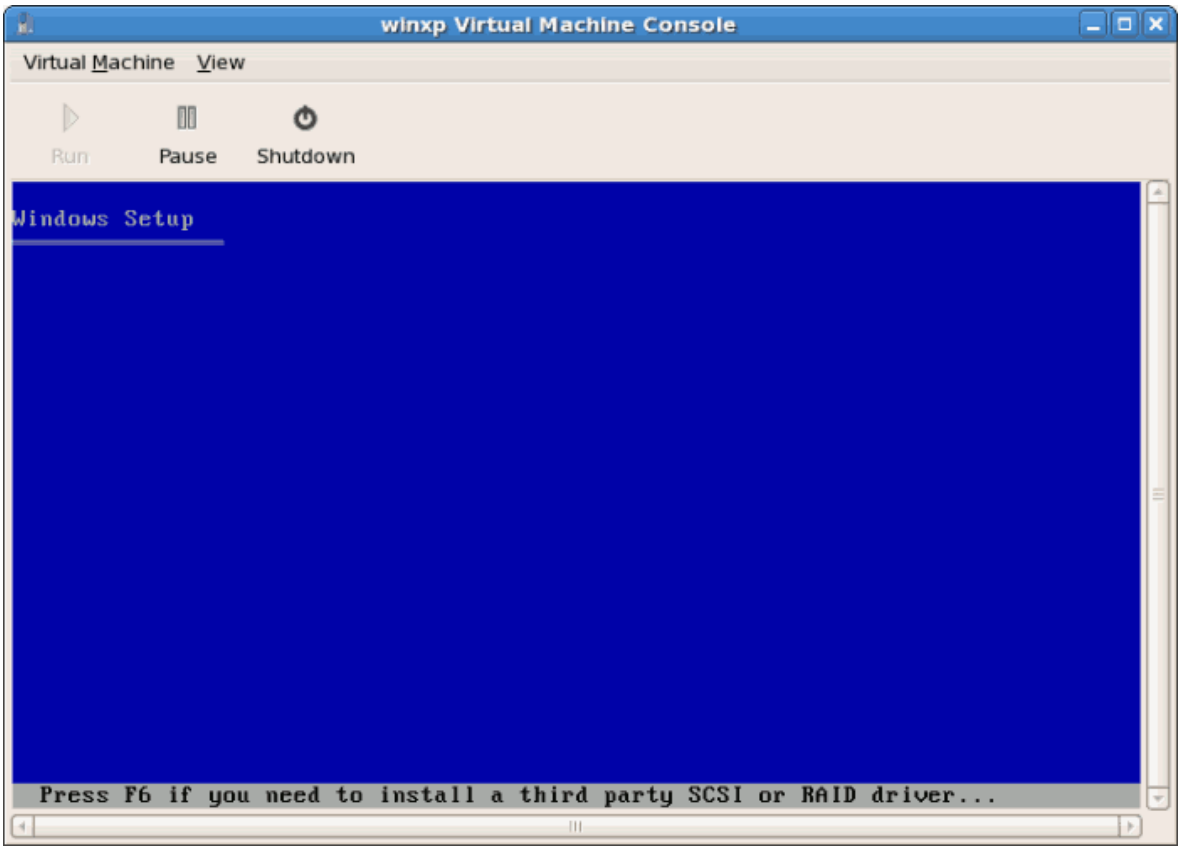
Assign sufficient virtual CPUs for the virtualized guest. If the guest runs a multithreaded application assign the number of virtualized CPUs it requires to run most efficiently. Do not assign more virtual CPUs than there are physical processors (or hyper-threads) available on the host system. It is possible to over allocate virtual processors, however, over allocating has a significant, negative affect on guest and host performance due to processor context switching overheads.



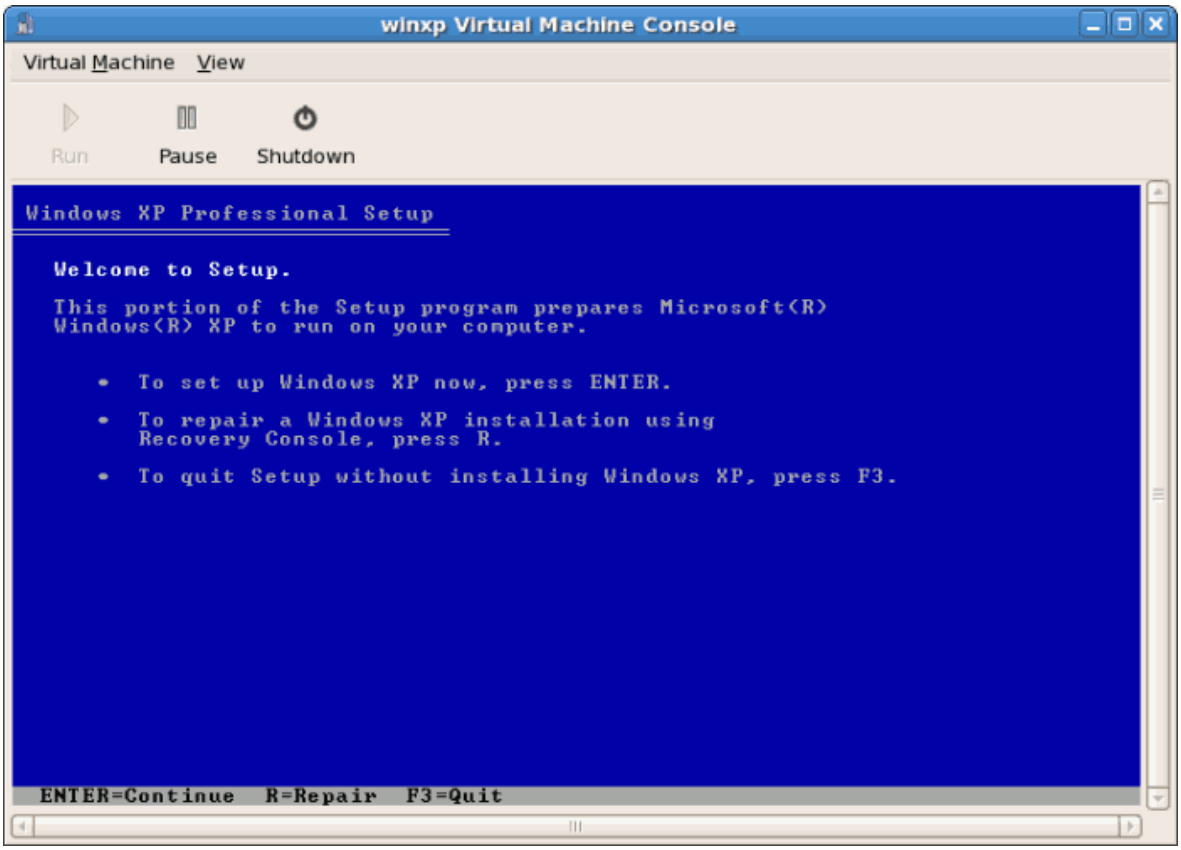
8. На следующем экране будут показаны введенные вами данные. Чтобы начать установку, нажмите кнопку завершения.

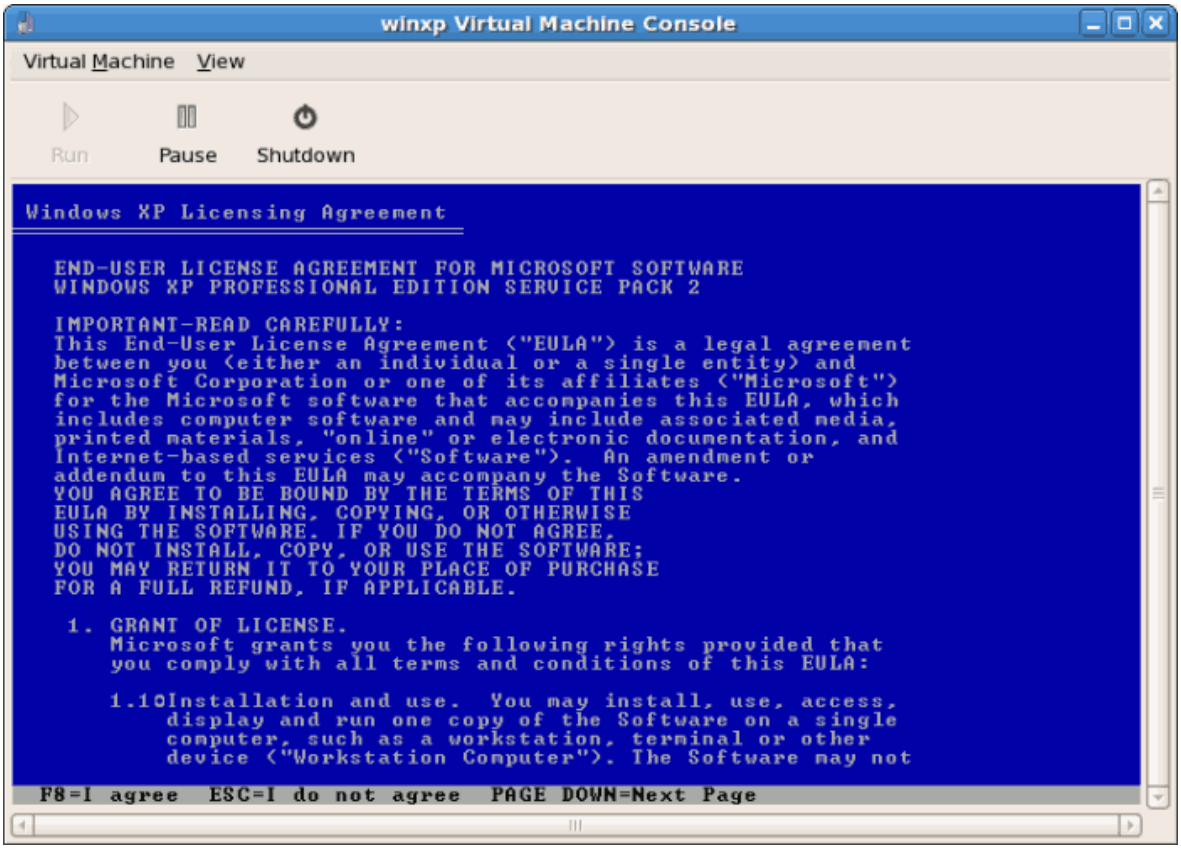


9. Начнется установка Windows. Так как потребуется выбрать оборудование, сразу после начала установки надо быстро открыть окно консоли. Нажав кнопку завершения, перейдите в окно просмотра **virt-manager** и выберите созданную гостевую систему Windows. Двойной щелчок мыши на имени системы откроет окно консоли. Быстро нажмите **F5** для выбора HAL, в появившемся окне диалога Windows-установки выберите вариант '`Generic i486 Platform`' (для перехода между пунктами используйте кнопки «Вверх» и «Вниз»).

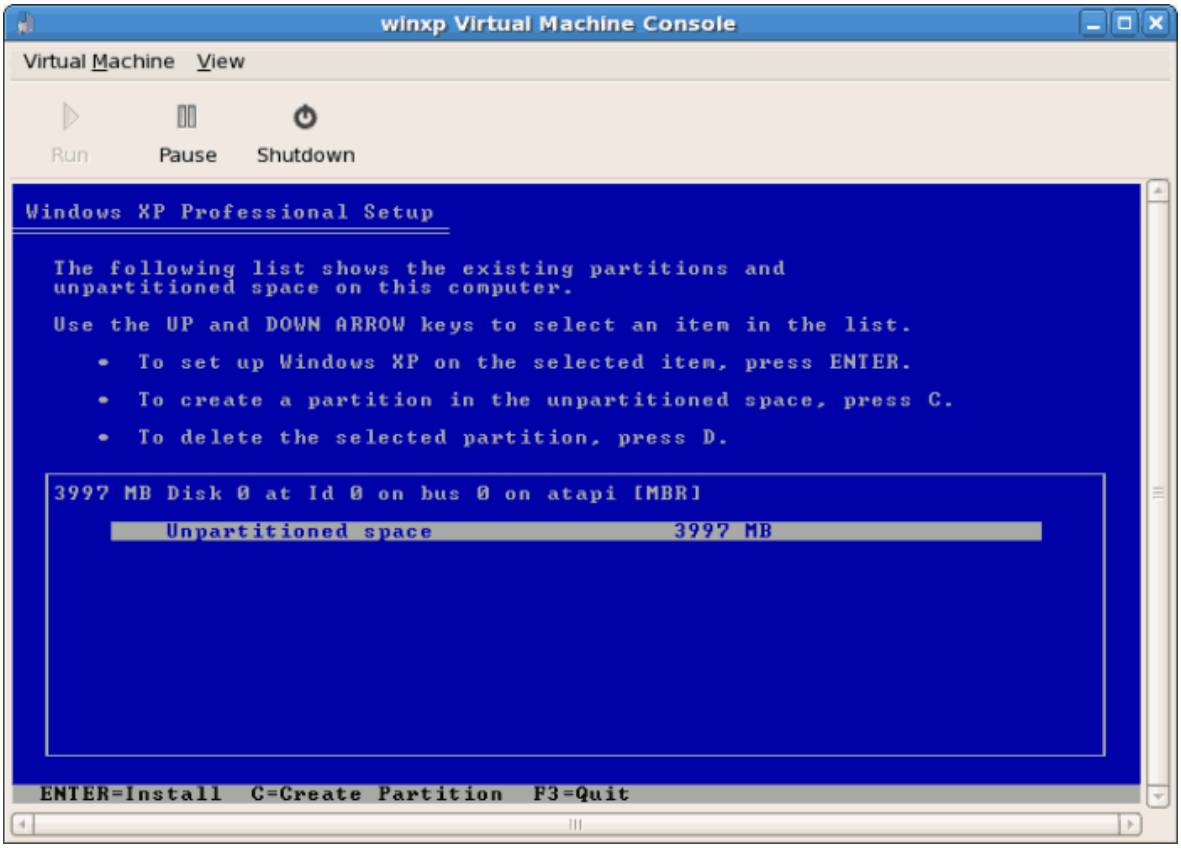


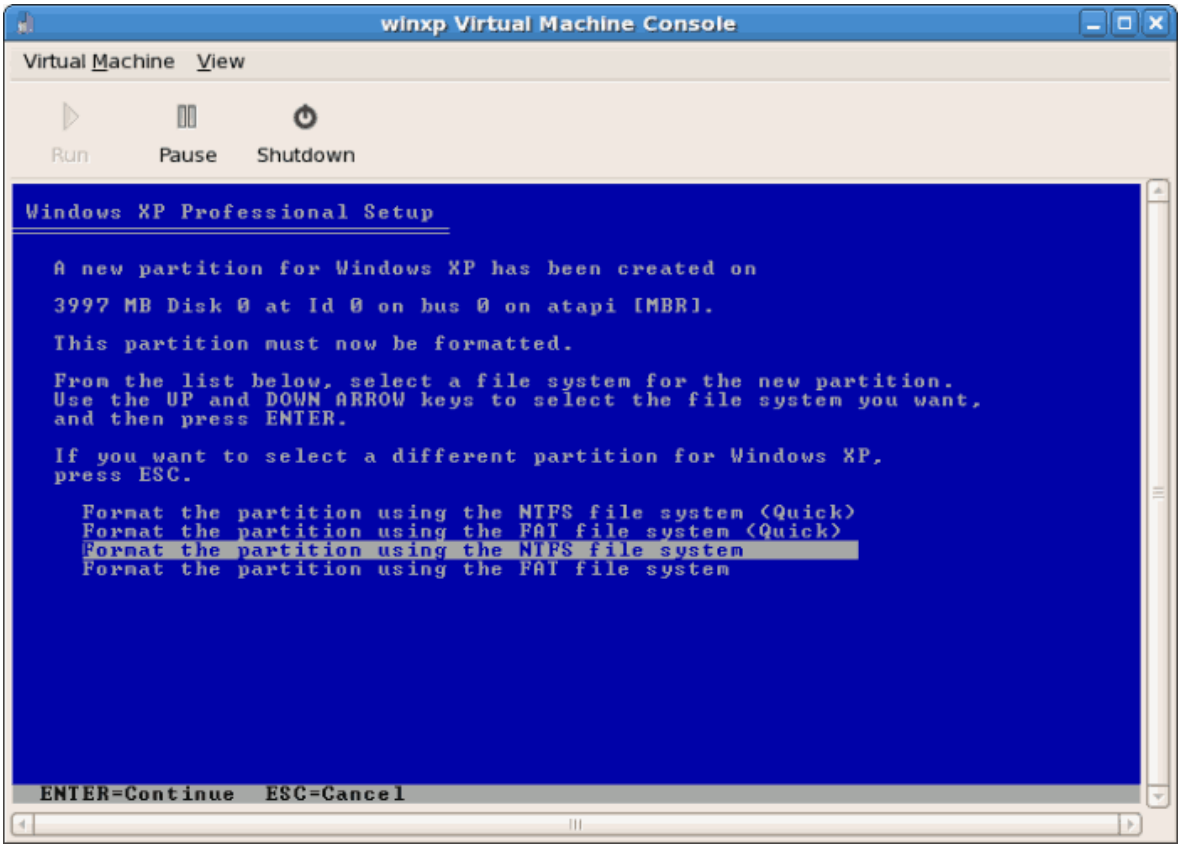
10. Установка Windows продолжится в обычном режиме.


winxp Virtual Machine Console
Virtual Machine View
Run
Pause
Shutdown
Windows XP Professional Setup
Welcome to Setup.
This portion of the Setup program prepares Microsoft(R) Windows(R) XP to run on your computer.
• To set up Windows XP now, press ENTER.
• To repair a Windows XP installation using Recovery Console, press R.
• To quit Setup without installing Windows XP, press F3.
ENTER=Continue R=Repair F3=Quit

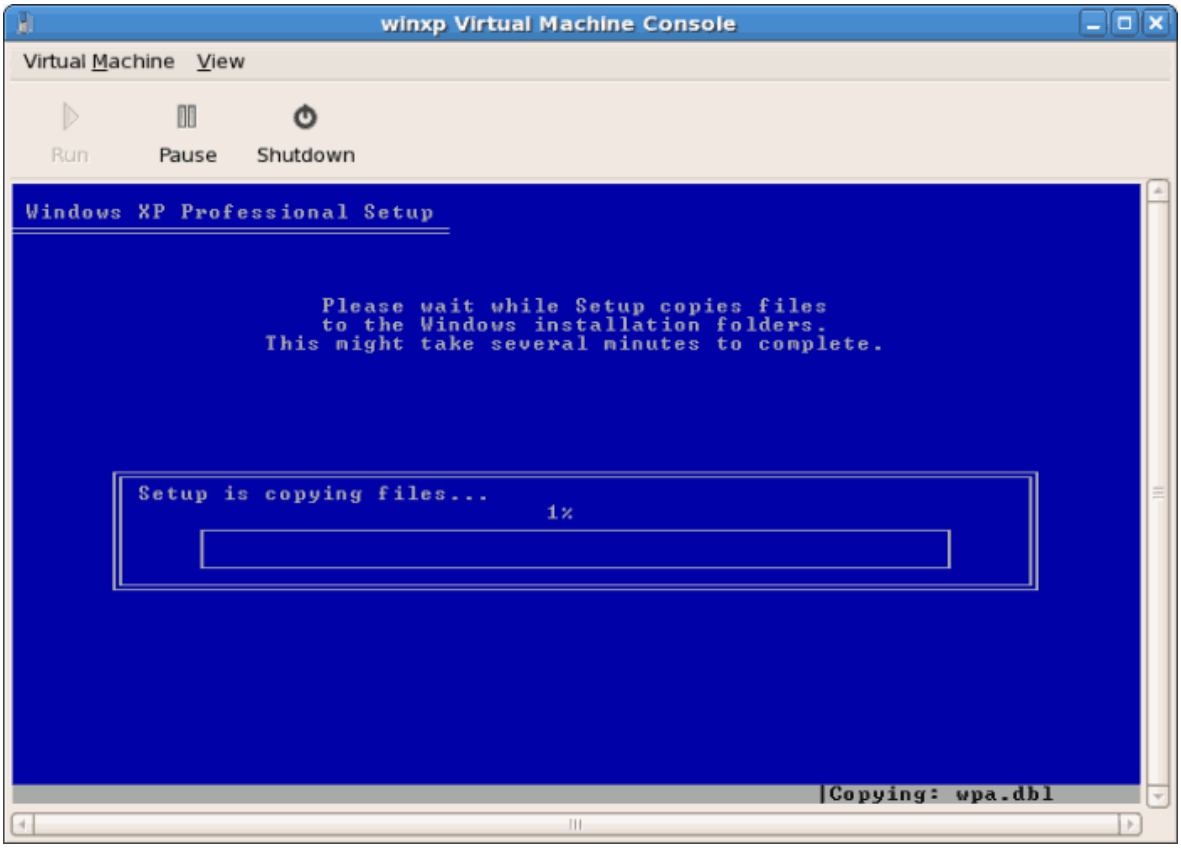

winxp Virtual Machine Console
Virtual Machine View
Run
Pause
Shutdown
Windows XP Licensing Agreement
END-USER LICENSE AGREEMENT FOR MICROSOFT SOFTWARE
WINDOWS XP PROFESSIONAL EDITION SERVICE PACK 2
IMPORTANT-READ CAREFULLY:
This End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Microsoft Corporation or one of its affiliates ("Microsoft") for the Microsoft software that accompanies this EULA, which includes computer software and may include associated media, printed materials, "online" or electronic documentation, and Internet-based services ("Software"). An amendment or addendum to this EULA may accompany the Software.
YOU AGREE TO BE BOUND BY THE TERMS OF THIS EULA BY INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE. IF YOU DO NOT AGREE, DO NOT INSTALL, COPY, OR USE THE SOFTWARE; YOU MAY RETURN IT TO YOUR PLACE OF PURCHASE FOR A FULL REFUND, IF APPLICABLE.
1. GRANT OF LICENSE.
Microsoft grants you the following rights provided that you comply with all terms and conditions of this EULA:
1.1 Installation and use. You may install, use, access, display and run one copy of the Software on a single computer, such as a workstation, terminal or other device ("Workstation Computer"). The Software may not
F8=I agree ESC=I do not agree PAGE DOWN=Next Page

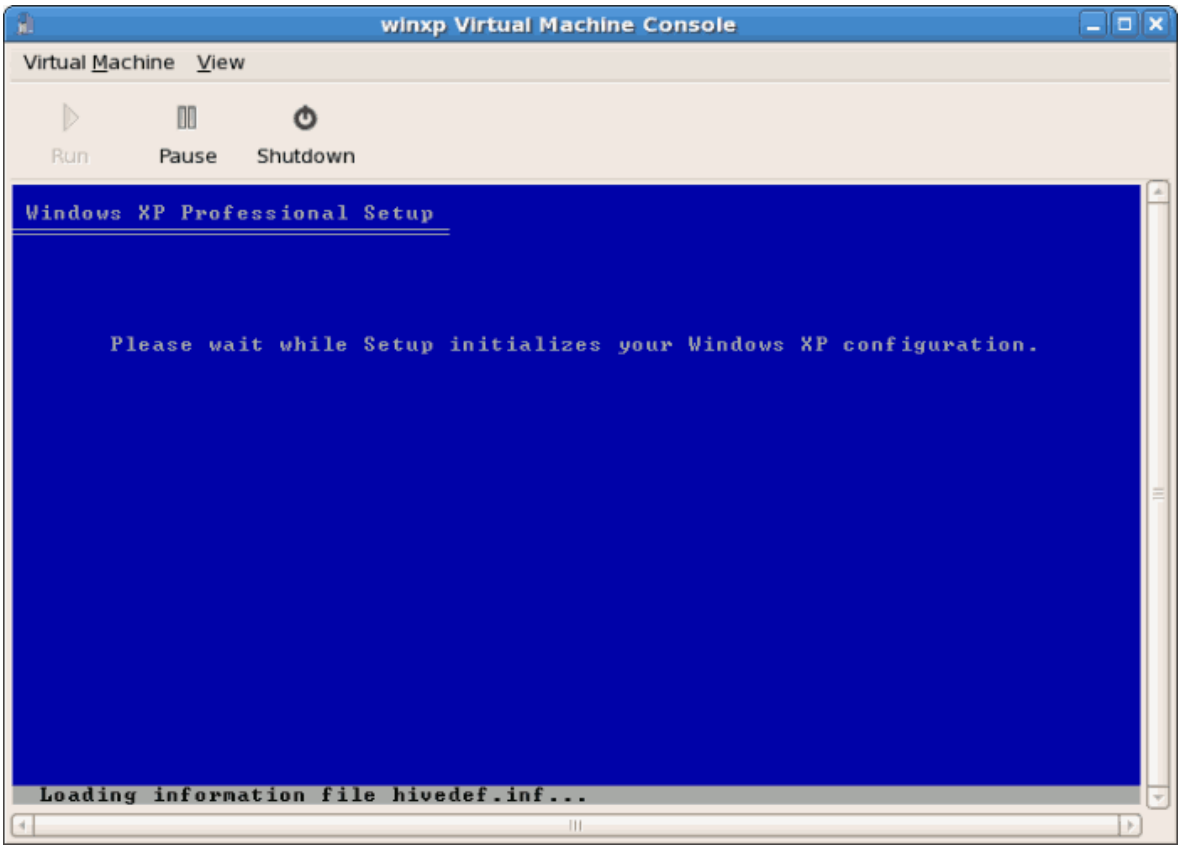
11. Создайте разделы на жестком диске.



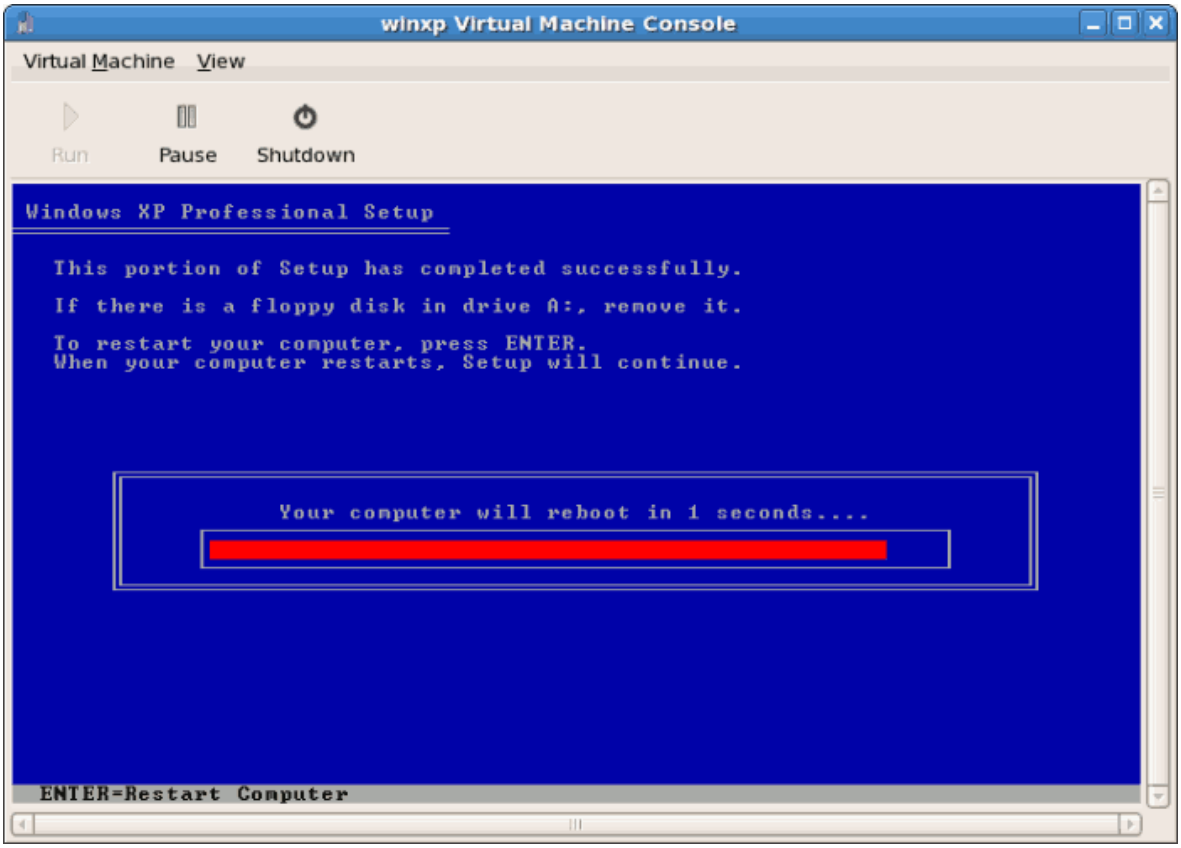


12. После форматирования диска Windows приступит к копированию файлов.





13. После завершения копирования будет выполнена перезагрузка Windows.

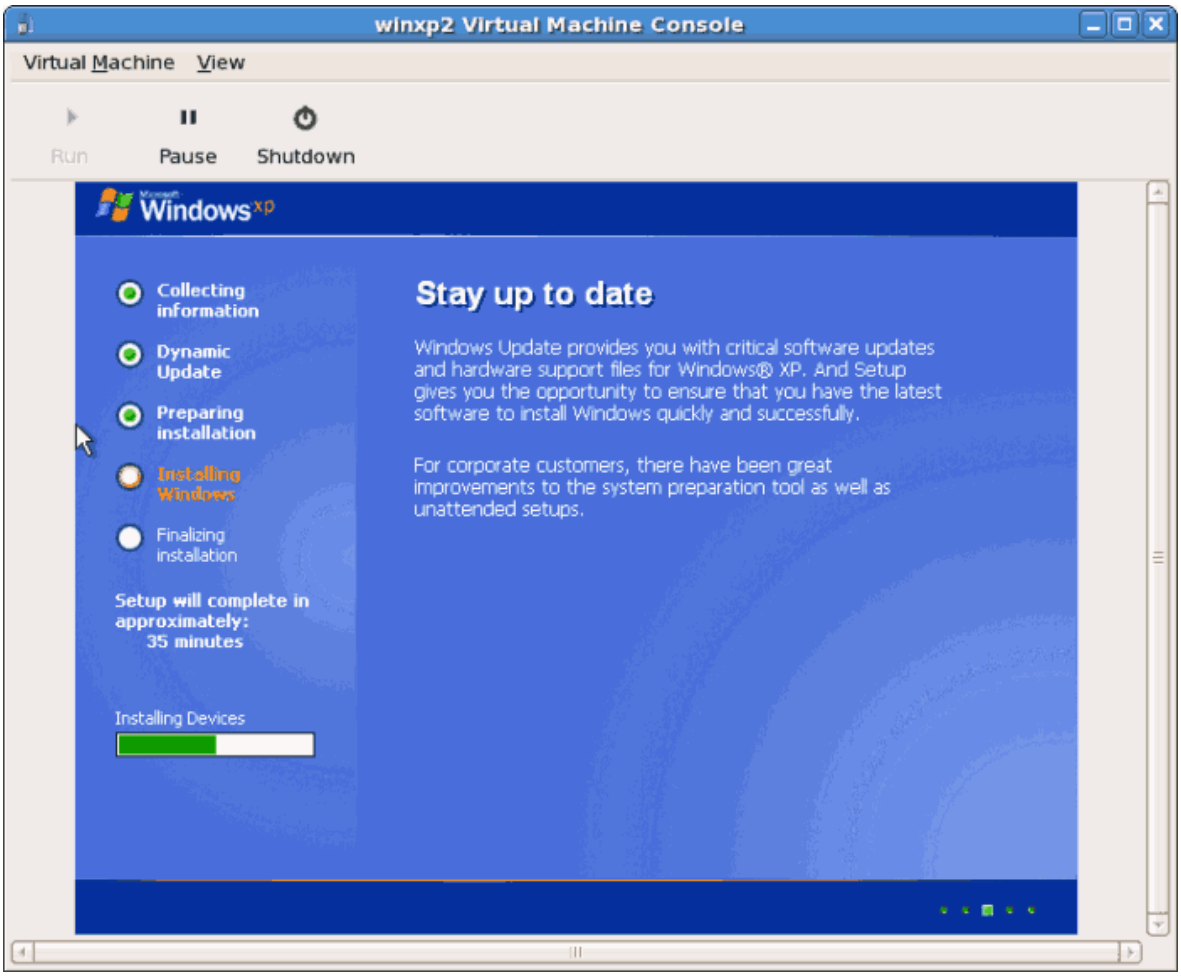


14. Перезапустите гостевую систему Windows

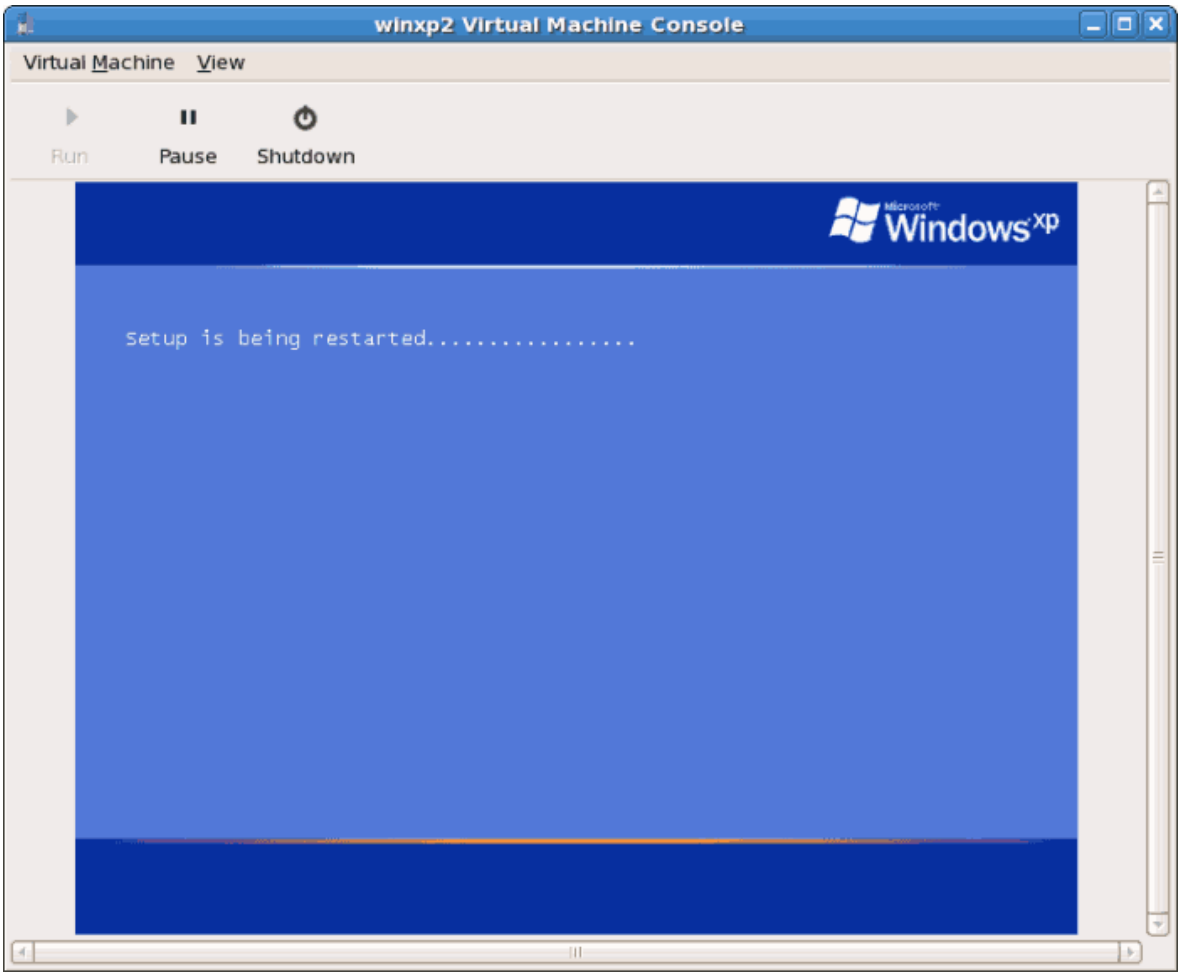
```
# virsh start имя
```

Замените *имя* именем созданной виртуальной машины.

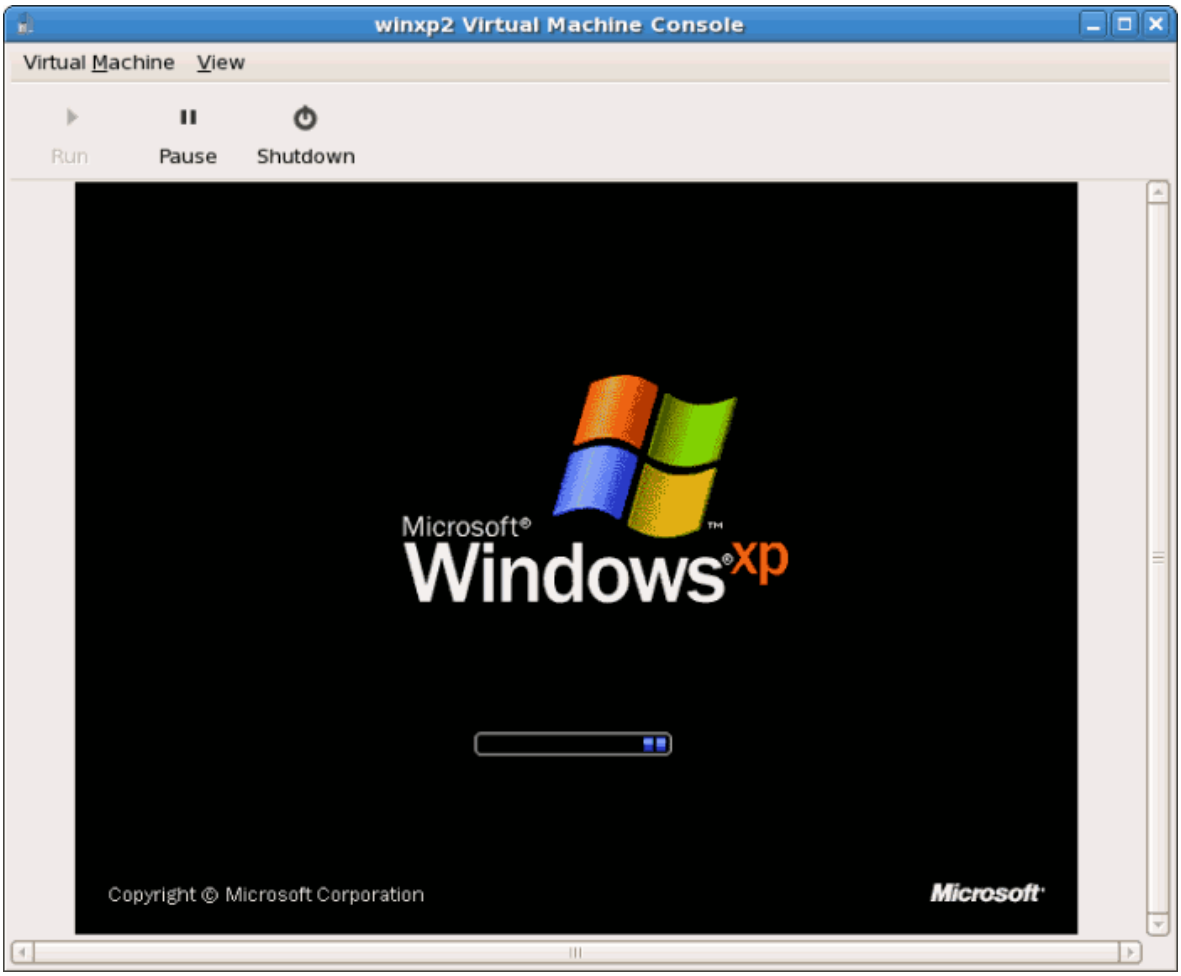
15. В открывшемся окне консоли вы увидите этап настройки установки Windows.



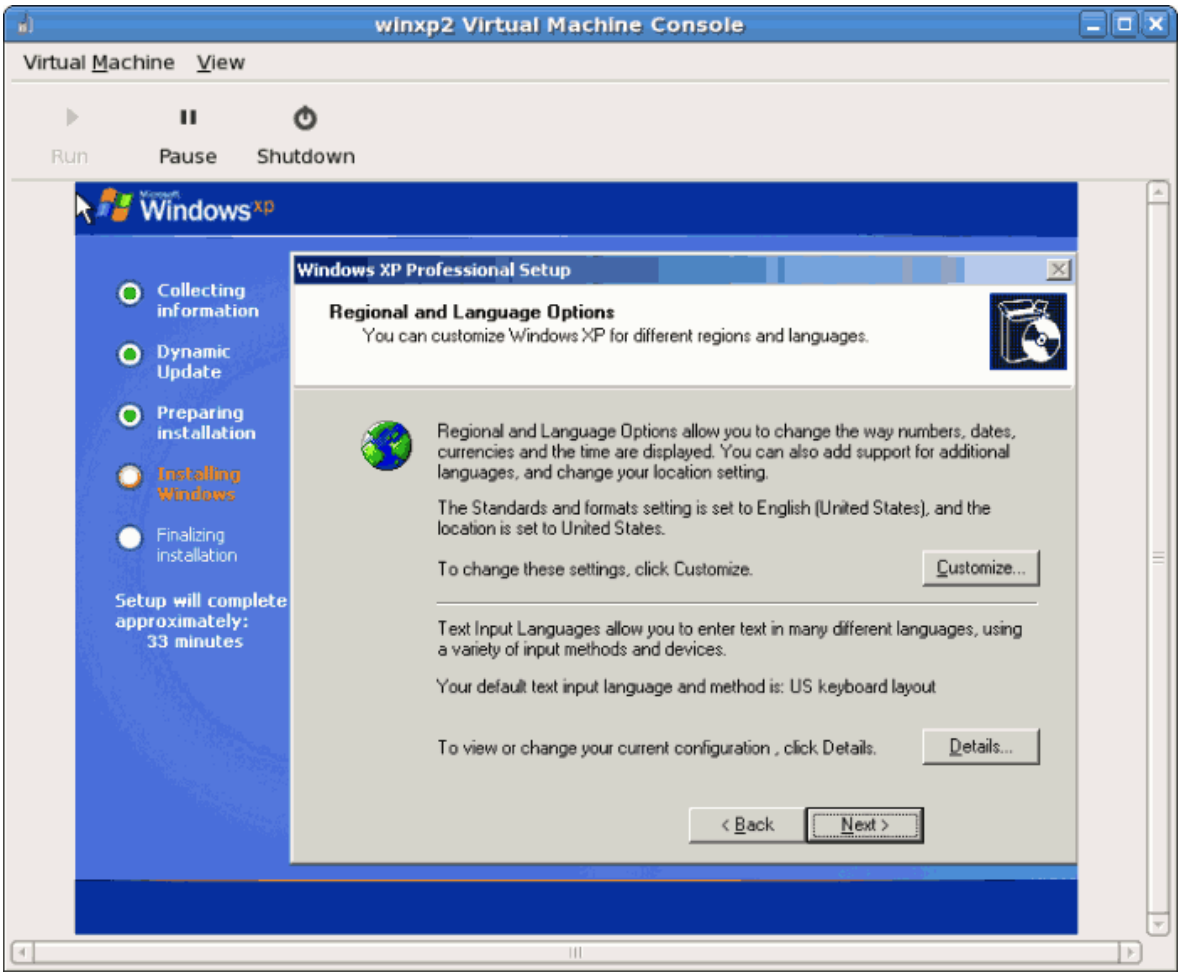
16. Если оказалось, что процесс установки завис на этом этапе, попробуйте перезапустить виртуальную машину еще раз, выполнив команду **# `virsh reboot` *имя***. Вы должны увидеть сообщение о повторном запуске процесса настройки.



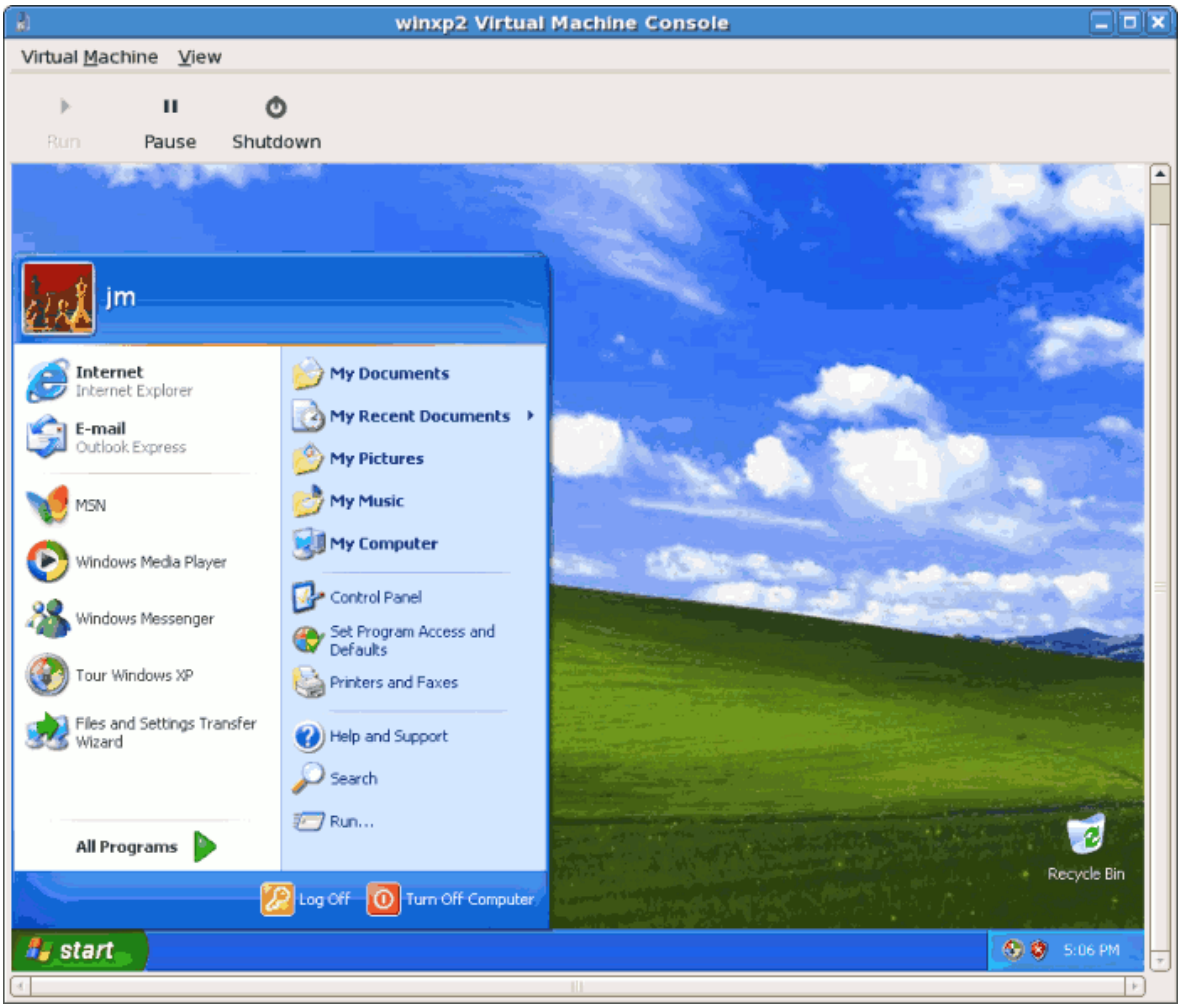
17. Наконец, появится экран загрузки Windows.



18. Теперь можно продолжить стандартную настройку установки Windows.



19. По завершению настройки появится рабочий стол Windows.



3.4. Установка Windows Server 2003 в качестве полностью виртуализированного гостя

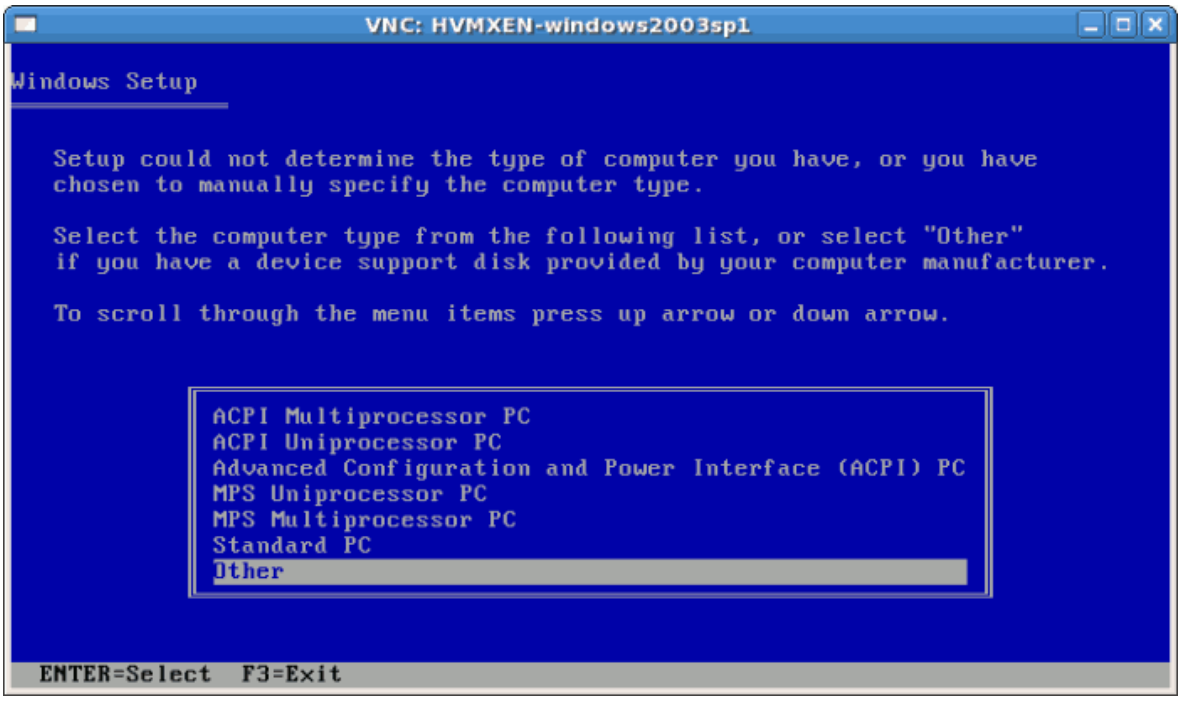
Дальше будет описан процесс установки Windows Server 2003 в качестве полностью виртуализированного гостя с помощью команды **`virt-install`**, которую можно использовать вместо **`virt-manager`**. Процесс установки аналогичен рассмотренному ранее (см. *Раздел 3.3, «Установка Windows XP в качестве полностью виртуализированного гостя»*).

1. При запуске команды **`virt-install`** для установки Windows Server 2003 откроется окно virt-viewer.

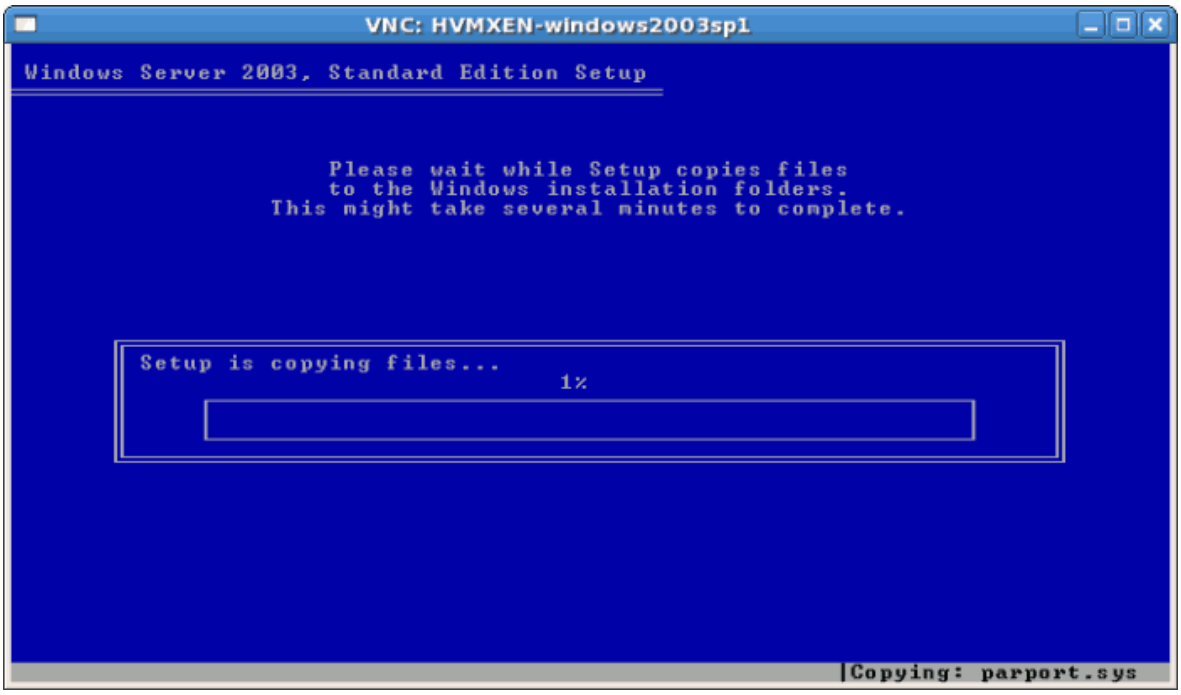
   Начните установку:

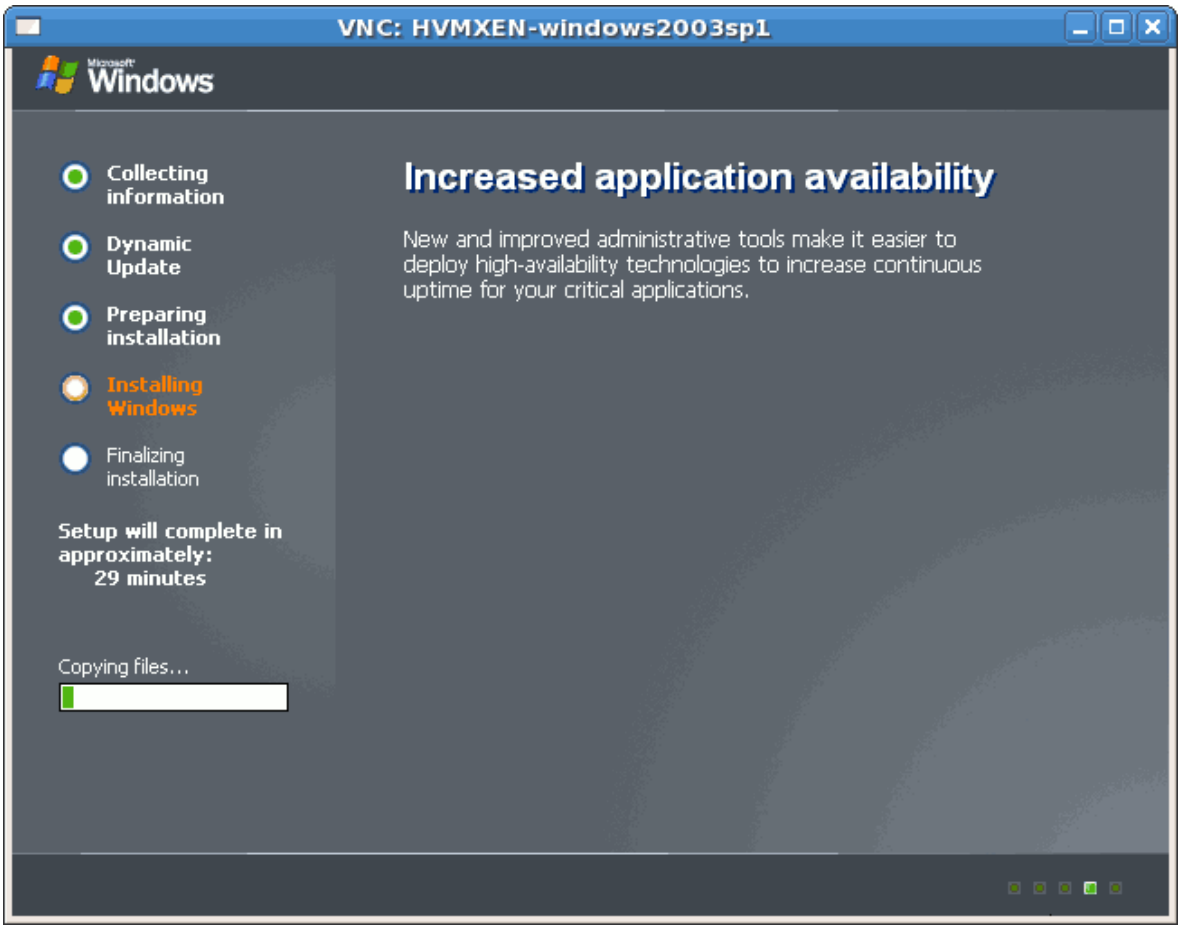
```
# virt-install -hvm -s 5 -f /var/lib/libvirt/images/windows2003spi1.dsk
 \
-n windows2003sp1 -cdrom=/ISOs/WIN/en_windows_server_2003_sp1.iso  \
-vnc -r 1024
```

2. После начала установки быстро нажмите **F5**, чтобы открыть окно выбора HAL или типа компьютера. Выберите стандартный компьютер. Если вы не успели нажать F5, придется начать установку заново.



3. Остальные этапы установки не отличаются от уже рассмотренных в предыдущей секции.





4. Вы успешно установили полностью виртуализированную гостевую систему Windows Server 2003.

3.5. Установка Windows XP Server 2008 в качестве полностью виртуализированного гостя

В этой секции рассматривается установка полностью виртуализированной системы Windows Server 2008.

Процедура 3.4. Установка Windows Server 2008 с помощью virt-manager

1. **Open virt-manager**
   Start **`virt-manager`**. Launch the **Virtual Machine Manager** application from the **Applications** menu and **System Tools** submenu. Alternatively, run the **`virt-manager`** command as root.

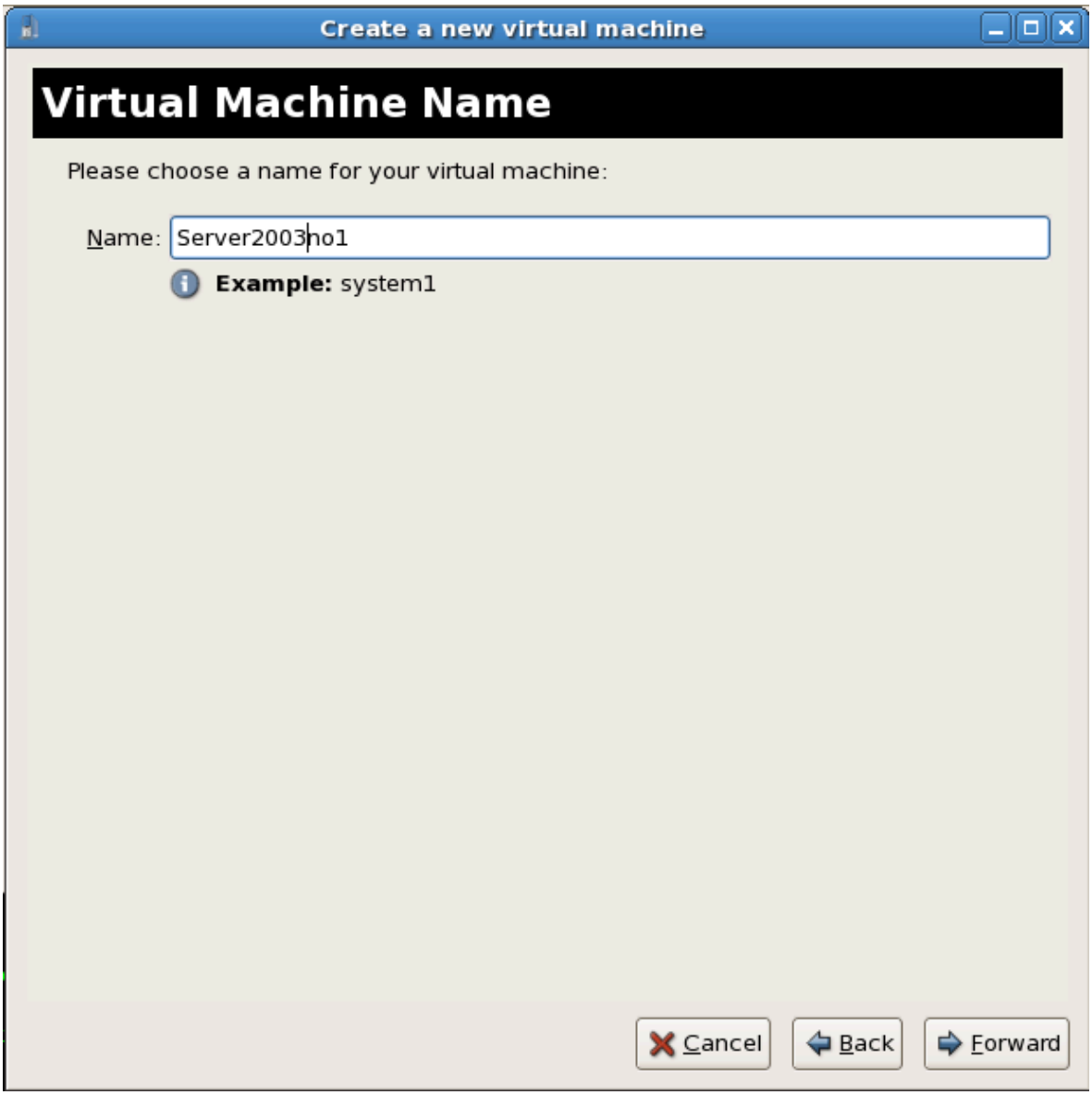
2. **Select the hypervisor**
   Select the hypervisor. If installed, select Xen or KVM. For this example, select KVM. Note that presently KVM is named `qemu`.

   После этого кнопка создания новой виртуальной машины станет доступна.

3. **Start the new virtual machine wizard**
   Pressing the **New** button starts the virtual machine creation wizard.

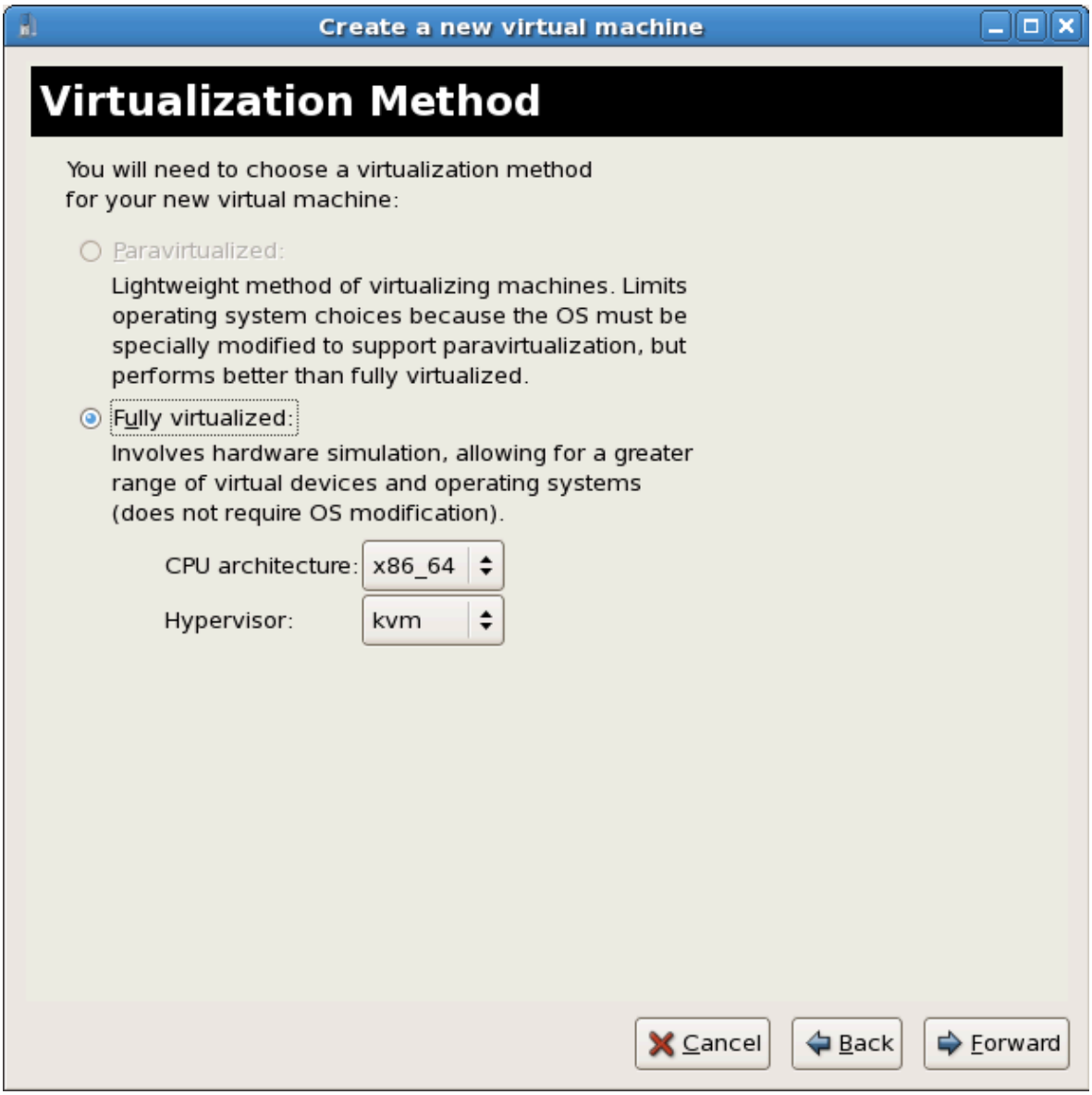
Press **Forward** to continue.

4. **Name the virtual machine**
   Введите имя для виртуализированного гостя. Обратите внимание, что имя не может содержать пробелы и знаки пунктуации.



Нажмите **Далее**.

5. **Choose a virtualization method**
   На этом этапе можно выбрать тип виртуализации. При этом в качестве гипервизора необходимо указать тот гипервизор, который был выбран в пункте 2. В этом примере будет снова выбран KVM.



Нажмите **Далее**.

6. **Select the installation method**
   Для любых версий Windows надо выбрать локальный установочный носитель.

   PXE можно выбрать, если есть сервер PXE, специально настроенный для выполнения сетевой установки Windows. В этом руководстве сетевая установка Windows не рассматривается.

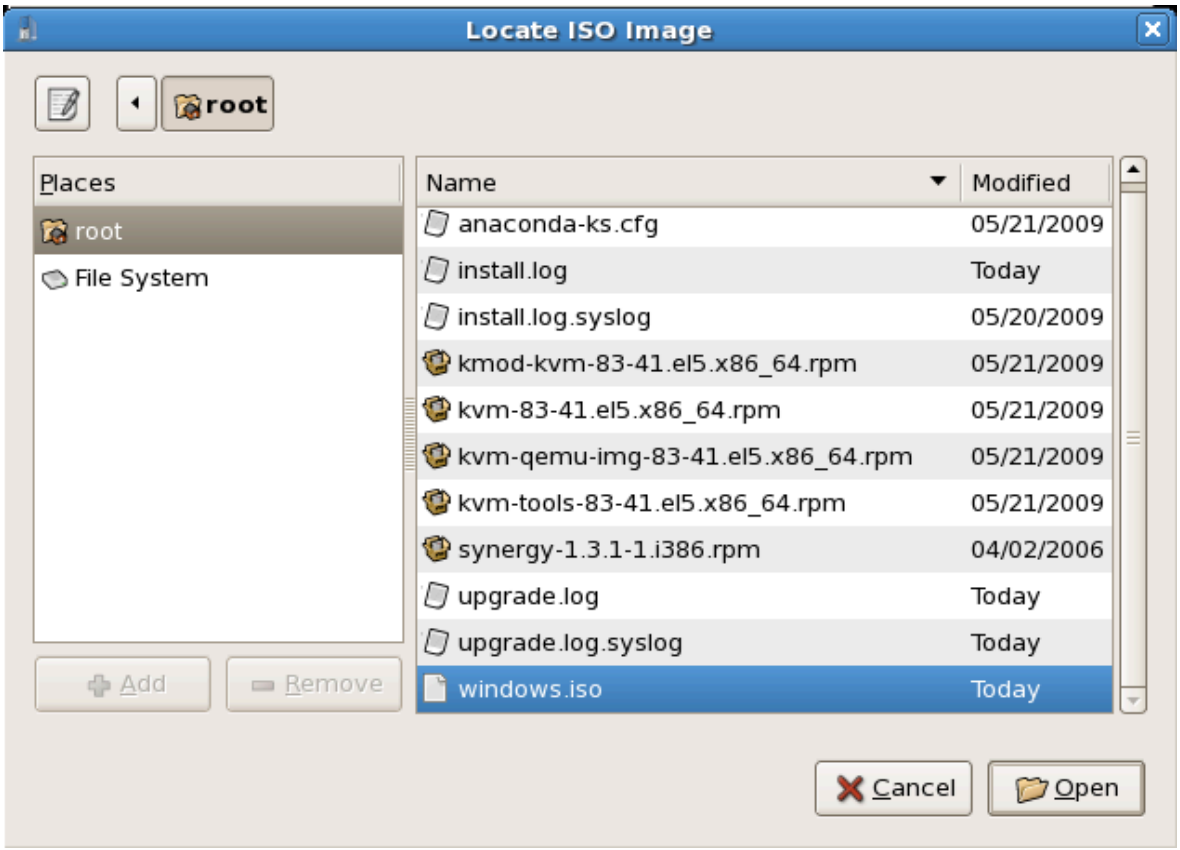
   В выпадающем списке **Тип ОС** выберите **Windows**, а в качестве самой системы укажите **Microsoft Windows 2008**.

Нажмите **Далее**.

7. **Locate installation media**
   Вы можете указать путь к ISO-образу или устройству чтения дисков. В приведенном примере будет указан путь к образу установочного компакт-диска Windows Server 2008.

   a. Press the **Browse** button.

   b. Выберите файл ISO.



Нажмите **Открыть** для подтверждения выбора.

c. Выбранный файл будет служить источником установки.

Нажмите **Далее**.

> **Image files and SELinux**
> Для хранения ISO-файлов и образов хранилищ рекомендуется использовать каталог **`/var/lib/libvirt/images/`**, так как другие каталоги могут требовать дополнительной настройки SELinux (см. *Раздел 7.1, «Виртуализация и SELinux»*).

8. **Storage setup**
   Можно выбрать физическое блочное устройство или файловый образ. Образы должны храниться в каталоге **`/var/lib/libvirt/images/`**. Убедитесь, что виртуализированному гостю предоставлено достаточно пространства.

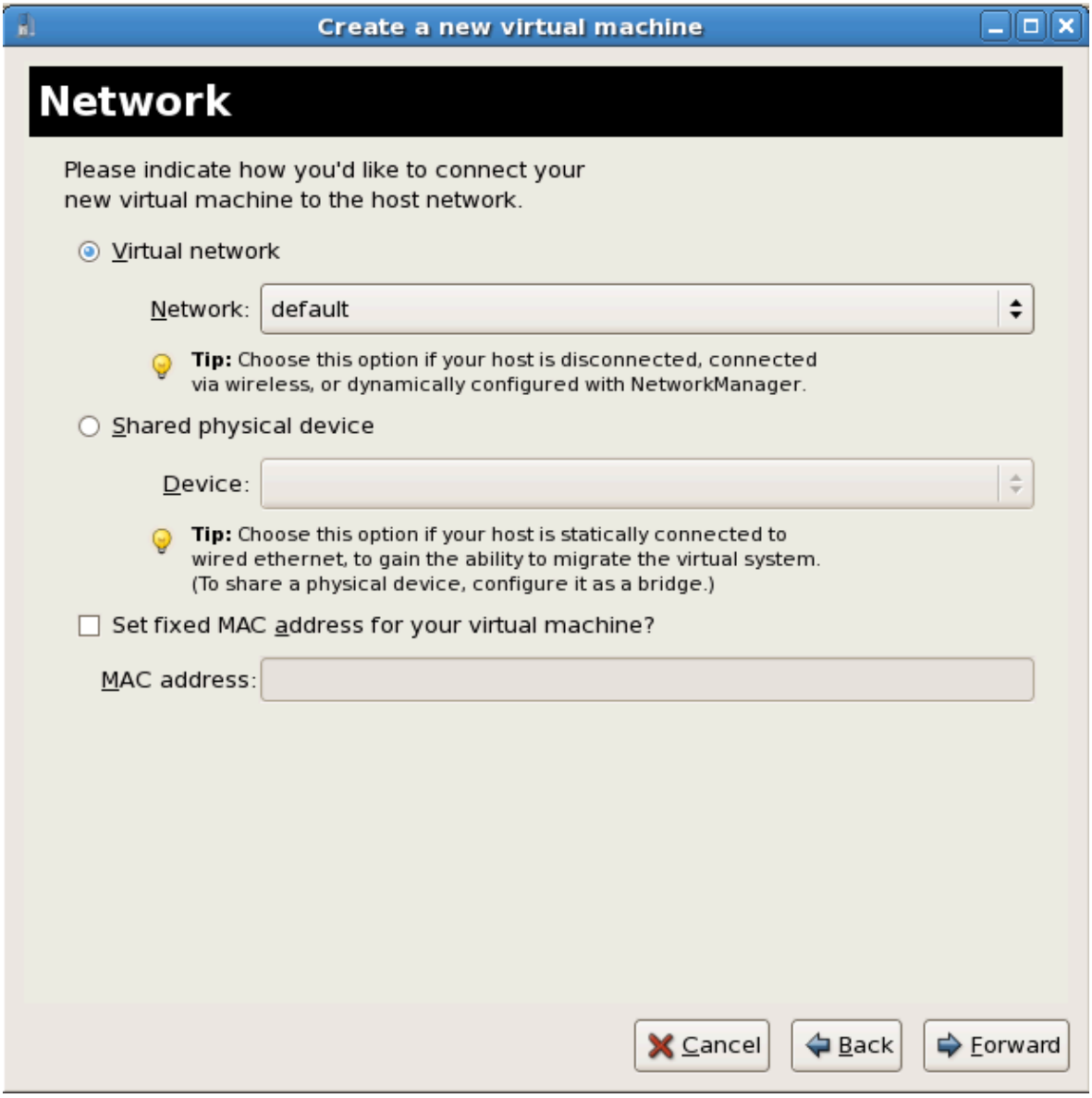
Намжите **Далее**.

9. **Network setup**

   Select either **Virtual network** or **Shared physical device**.

   The virtual network option uses Network Address Translation (NAT) to share the default network device with the virtualized guest. Use the virtual network option for wireless networks.

   The shared physical device option uses a network bond to give the virtualized guest full access to a network device.



Press **Forward** to continue.

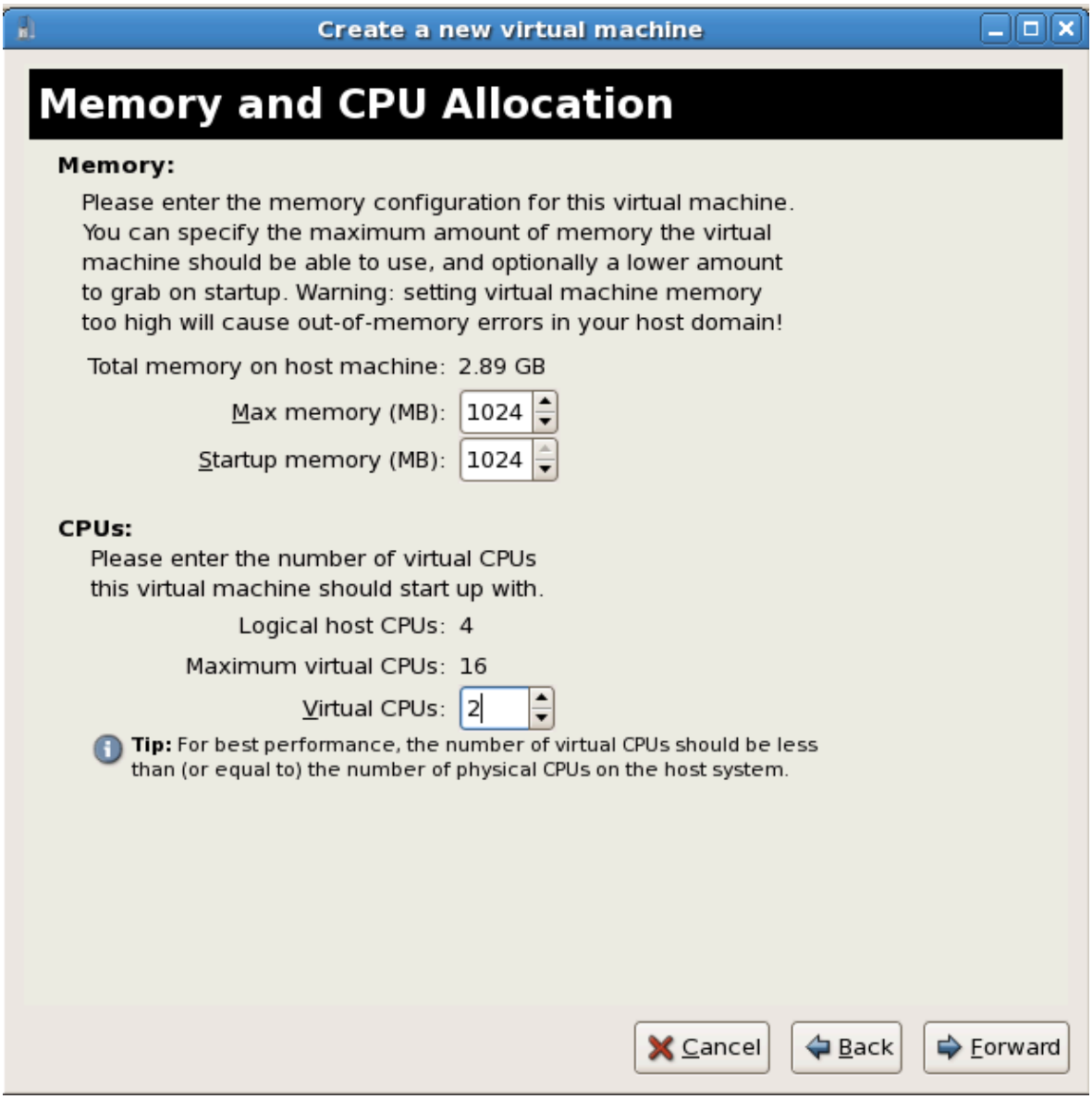
10. **Memory and CPU allocation**

    The Allocate memory and CPU window displays. Choose appropriate values for the virtualized CPUs and RAM allocation. These values affect the host's and guest's performance.

    Virtualized guests require sufficient physical memory (RAM) to run efficiently and effectively. Choose a memory value which suits your guest operating system and application requirements. Windows Server 2008. Remember, guests use physical RAM. Running too many guests or leaving insufficient memory for the host system results in significant usage of virtual memory and swapping. Virtual memory is significantly slower causing degraded system performance and responsiveness. Ensure to allocate sufficient memory for all guests and the host to operate effectively.

    Assign sufficient virtual CPUs for the virtualized guest. If the guest runs a multithreaded application assign the number of virtualized CPUs it requires to run most efficiently. Do not assign more virtual CPUs than there are physical processors (or hyper-threads) available on

the host system. It is possible to over allocate virtual processors, however, over allocating has a significant, negative affect on guest and host performance due to processor context switching overheads.

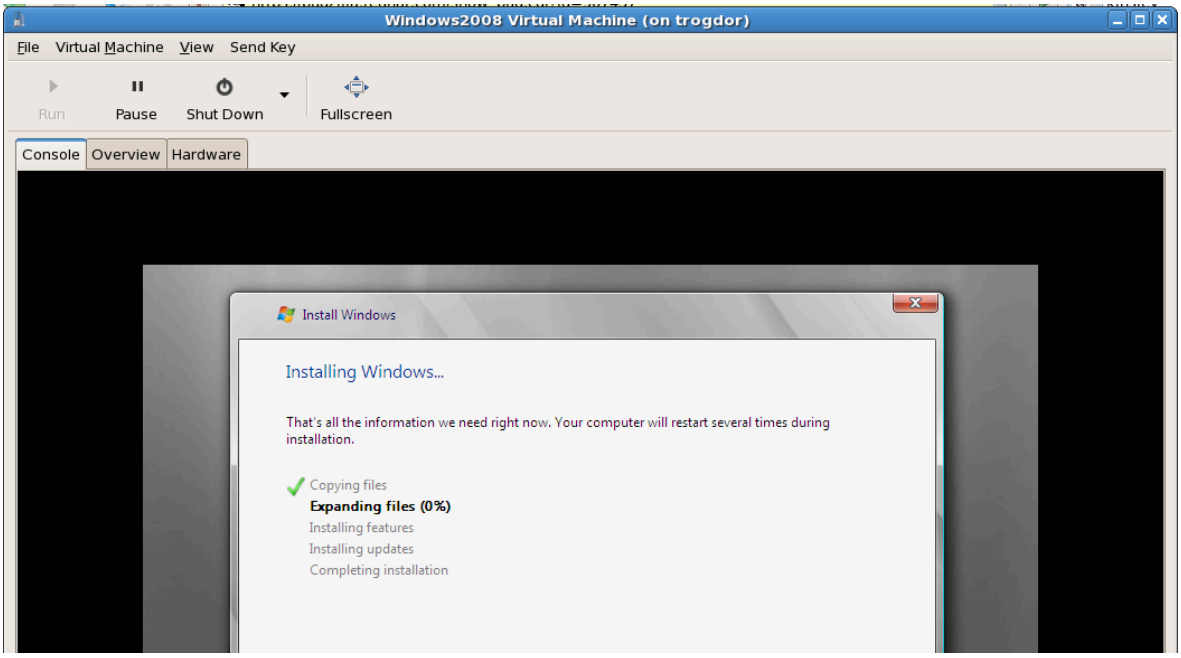


Press **Forward** to continue.

11. **Verify and start guest installation**
    В следующем окне проверьте настройки.

Нажмите **Продолжить**.

12. **Установка Windows**



Дождитесь завершения процесса установки Windows Server 2008. Сам процесс здесь не рассматривается, его описание можно найти в *документации Microsoft*[2].

Часть II. Configuration

Настройка виртуализации в Fedora

В последующих главах будет рассмотрена настройка различных задач виртуализации: добавление накопителей и сетевых устройств, усиление защиты, повышение производительности, использование паравиртуализированных драйверов в полностью виртуализированных гостевых системах.

Виртуализированные блочные устройства

В этой главе будет рассмотрен процесс установки и настройки блочных устройств в виртуализированных гостевых системах. Термин «блочные устройства» объединяет различные виды устройств хранения.

4.1. Создание контроллера виртуализированного дисковода

В контроллерах дисководов все еще есть необходимость в старых операционных системах, в частности, для установки драйверов. В настоящее время обращение к физическим дисководам из виртуализированных систем невозможно, но поддерживается создание и обращение к образам дискет из виртуализированных дисководов.

Для этого потребуется файл образа дискеты, создать который можно с помощью команды **dd**. В приведенном ниже примере замените /dev/fd0 именем дисковода и присвойте диску соответствующее имя.

```
# dd if=/dev/fd0 of=~/legacydrivers.img
```

Замечание о паравиртуализированных драйверах

Паравиртуализированные драйверы позволяют сопоставить физический дисковод полностью виртуализированным системам.

В этой главе в качестве примера будет рассмотрена виртуальная машина, созданная с помощью **virt-manager**, в которой выполняется полностью виртуализированная установка Linux, образ которой образ расположен в **/var/lib/libvirt/images/rhel5FV.img**. В приведенном примере будет использоваться гипервизор Xen.

1. В работающей гостевой системе создайте файл конфигурации в формате XML для гостевого образа:

   ```
   # virsh dumpxml rhel5FV > rhel5FV.xml
   ```

   Стандартные настройки в созданном XML-файле можно будет изменить в соответствии с вашими требованиями. *Глава 18, Создание специализированных сценариев libvirt* содержит подробную информацию о создании файлов конфигурации с помощью virsh.

2. Создайте образ дискеты для гостевой системы.

   ```
   # dd if=/dev/zero of=/var/lib/libvirt/images/rhel5FV-floppy.img bs=512
    count=2880
   ```

3. Добавьте приведенный текст, указав путь к вашему файлу конфигурации. В этом примере будет создан гость с дисководом в виде виртуального устройства на основе файла.

```
<disk type='file' device='floppy'>
        <source file='/var/lib/libvirt/images/rhel5FV-floppy.img'/>
        <target dev='fda'/>
</disk>
```

4. Остановите гостевую систему.

```
# virsh stop rhel5FV
```

5. Перезапустите ее с использованием нового файла конфигурации.

```
# virsh create rhel5FV.xml
```

Теперь дисковод должен быть доступен в гостевой системе и сохранен в размещающей системе в виде файла образа.

4.2. Добавление устройств хранения в гостевую систему

В этой секции будет рассмотрен процесс добавления устройств хранения в виртуальную машину. Дополнительные накопители могут быть добавлены только после создания гостевых систем. Типы поддерживаемых накопителей и протоколов:

- разделы на локальных жестких дисках;
- логические тома;
- Fibre Channel или iSCSI, подключенные напрямую;
- файловые контейнеры, расположенные в файловой системе размещающей системы;
- файловые системы **NFS**, напрямую подключенные виртуальной машиной;
- накопители iSCSI, к которым гостевые системы обращаются напрямую.
- Кластерные файловые системы (**GFS**).

Добавление в гостевую систему накопителя на основе файла

Накопители на основе файлов представляют собой файлы, расположенные в файловой системе размещающей операционной системы, которые функционируют как виртуализированные жесткие диски для виртуализированных гостей. Последовательность действий при добавлении таких устройств хранения:

1. Можно создать пустой файл контейнера или использовать существующий (например, файл ISO).

   a. Для создания разреженного файла выполните приведенную ниже команду (обратите внимание, что использование таких файлов не рекомендуется, так как при этом может быть нарушена целостность данных). Эти файлы создаются намного быстрее и обычно

используются при тестировании, но в производственной среде все же лучше к ним не прибегать.

```
# dd if=/dev/zero of=/var/lib/libvirt/images/FileName.img bs=1M
 seek=4096 count=0
```

b. Чтобы создать неразреженный файл, выполните

```
# dd if=/dev/zero of=/var/lib/libvirt/images/FileName.img bs=1M
 count=4096
```

Обе команды создадут файл размером 400 мегабайт, который будет служить основой для дополнительного хранилища .

2. Сохраните существующую конфигурацию гостя в отдельный файл. В приведенном примере файл конфигурации гостевой системы *Guest1* будет сохранен в домашний каталог.

```
# virsh dumpxml Guest1 > ~/Guest1.xml
```

3. Теперь откройте этот файл конфигурации в текстовом редакторе и найдите записи, которые начинаются с "disk=". Они выглядят примерно так:

```
>disk type='file' device='disk'<
        >driver name='tap' type='aio'/<
        >source file='/var/lib/libvirt/images/Guest1.img'/<
        >target dev='xvda'/<
>/disk<
```

4. Добавьте дополнительное хранилище, добавив соответствующую запись в конец секции disk=. Не забудьте указать имя виртуального блочного устройства, которое еще не упоминается в файле конфигурации. Пример добавления образа **FileName.img**:

```
>disk type='file' device='disk'<
        >driver name='tap' type='aio'/<
        >source file='/var/lib/libvirt/images/Guest1.img'/<
        >target dev='xvda'/<
>/disk<
>disk type='file' device='disk'<
        >driver name='tap' type='aio'/<
        >source file='/var/lib/libvirt/images/FileName.img'/<
        >target dev='hda'/<
>/disk<
```

5. Перезапустите гостевую систему с использованием нового файла конфигурации.

```
# virsh create Guest1.xml
```

6. Приведенная ниже информация применима лишь к гостевым системам Linux. Другие операционные системы могут отличаться в плане работы с новыми устройствами хранения. Поэтому за информацией следует обратиться к документации операционной системы.

   Гость теперь использует файл **`FileName.img`** как устройство **`/dev/hdb`**. Это устройство надо отформатировать. Так, в гостевой системе создайте один основной раздел, занимающий все устройство, и отформатируйте его.

   a. Нажмите *n*, чтобы создать новый раздел.

   ```
   # fdisk /dev/hdb
   Command (m for help):
   ```

   b. Нажмите *p*, чтобы определить этот раздел как основной.

   ```
   Command action
      e   extended
      p   primary partition (1-4)
   ```

   c. Выберите доступный номер раздела. В этом примере будет выбран первый раздел.

   ```
   Partition number (1-4): 1
   ```

   d. Нажмите *`Enter`* и введите номер первого цилиндра.

   ```
   First cylinder (1-400, default 1):
   ```

   e. Выберите размер раздела. В этом примере раздел будет занимать весь диск. Нажмите *`Enter`*.

   ```
   Last cylinder or +size or +sizeM or +sizeK (2-400, default 400):
   ```

   f. Нажмите *`t`*, чтобы указать тип раздела.

   ```
   Command (m for help): t
   ```

   g. Выберите номер созданного раздела.

   ```
   Partition number (1-4): 1
   ```

   h. Введите *83* (раздел Linux).

   ```
   Hex code (type L to list codes): 83
   ```

   i. сохраните изменения и нажмите *q* для выхода.

   ```
   Command (m for help): w
   ```

```
Command (m for help): q
```

j. Создайте в разделе файловую систему `ext3`.

```
# mke2fs -j /dev/hdb
```

7. Подключите диск.

```
# mount /dev/hdb1 /myfiles
```

Теперь гостевая система обладает дополнительным виртуализированным файловым устройством хранения.

Добавление жестких дисков и других блочных устройств в гостевую систему

Системные администраторы обычно используют дополнительные жесткие диски для увеличения пространства для хранения информации или для разделения системных и пользовательских данных. *Процедура 4.1, «Добавление физических блочных устройств в виртуализированную гостевую систему»* описывает процесс добавления жесткого диска узла в виртуализированную гостевую систему.

Процесс добавления аналогичен для всех типов физических блочных устройств, будь то CD-ROM, DVD или дисковод.

Процедура 4.1. Добавление физических блочных устройств в виртуализированную гостевую систему

1. Физически подключите жесткий диск. Настройте к нему доступ.

2. При необходимости настройте режим **`multipath`** и сохранение постоянства в размещающей системе.

3. Выполните приведенную ниже команду **`virsh attach`**. Замените *`myguest`* именем вашей гостевой системы, *`/dev/hdb1`* добавляемым устройством, а вместо *`hdc`* укажите расположение устройства в гостевой системе. Обратите внимание, что на месте *`hdc`* должно быть незанятое имя устройства. Для гостевых систем Windows укажите *`hd*`*.

   Для устройств CD-ROM и DVD добавьте параметр *`--type hdd`*.

   Для дисководов добавьте параметр *`--type floppy`*.

```
# virsh attach-disk myguest /dev/hdb1 hdc --driver tap --mode readonly
```

4. Гостевая система теперь обладает дополнительным жестким диском с именем **`/dev/hdb`** (в Linux) или **`D: drive`** (в Windows). Возможно, это устройство потребуется заново отформатировать.

4.3. Настройка постоянного хранилища

В окружениях с внешними накопителями (например, на основе Fibre Channel или iSCSI) рекомендуется настроить постоянные имена устройств, что облегчит выполнение живой миграции, так как в разных системах будут использоваться одни и те же имена.

Уникальные идентификаторы UUID (Universally Unique Identifier) — стандарт идентификации компьютеров и устройств в распределенных компьютерных окружениях. В этой секции идентификаторы UUID будут использоваться для идентификации LUN iSCSI и Fibre Channel. UUID аналогичен метке устройства и сохраняется между перезагрузками, отключениями и сменой устройств.

Настройка одного пути должна использоваться в системах без **multipath**. *Многопутевая настройка* может использоваться в системах с **multipath**.

Настройка одного пути

В этой секции будет рассмотрено, как обеспечить постоянство *LUN* с помощью **udev**. Используйте этот метод только, если в размещающей системе не настроены многопутевые возможности.

1. Внесите изменения в файл **/etc/scsi_id.config**.

   a. Убедитесь, что строка **options=-b** отмечена как комментарий.

   ```
   # options=-b
   ```

   b. Добавьте строку

   ```
   options=-g
   ```

   При этом **udev** будет подразумевать, что все подключенные устройства SCSI возвращают UUID.

2. Чтобы отобразить UUID для конкретного устройства, выполните команду **scsi_id -g -s /block/sd***. Пример:

   ```
   # scsi_id -g -s /block/sd*
   3600a0b8000132751000000015427b625e
   ```

   Вывод команды содержит UUID устройства `/dev/sdc`.

3. Убедитесь, что UUID, полученный в результате выполнения команды **scsi_id -g -s /block/sd***, совпадает с идентификатором, получаемым компьютером, который обращается к устройству.

4. Далее следует создать правило для назначения имени устройству. В каталоге **/etc/udev/rules.d** создайте файл **20-names.rules**, в который мы будем добавлять все новые правила. Формат правил:

   ```
   KERNEL="sd*", BUS="scsi", PROGRAM="/sbin/scsi_id -g -s", RESULT=UUID,
    NAME=имя_устройства
   ```

Замените *UUID* полученным ранее значением, а *имя_устройства* именем. Пример правила:

```
KERNEL="sd*", BUS="scsi", PROGRAM="/sbin/scsi_id -g -s",
 RESULT="3600a0b8000132751000000015427b625e", NAME="rack4row16"
```

Демон **udev** теперь будет искать в правиле все устройства **/dev/sd*** для заданного UUID. После подключения найденного устройства в систему ему будет присвоено заданное правилом имя. Так, в нашем примере устройству с UUID равным 3600a0b8000132751000000015427b625e будет присвоено имя **/dev/rack4row16**.

5. В файл **/etc/rc.local** добавьте строку

```
/sbin/start_udev
```

6. Скопируйте изменения в файлы **/etc/scsi_id.config**, **/etc/udev/rules.d/20-names.rules**, **/etc/rc.local** на всех узлах.

```
/sbin/start_udev
```

Сетевые устройства хранения с настроенными правилами теперь будут использовать одинаковые имена на всех узлах, где вы применили изменения. Теперь при миграции гостевых систем между узлами можно использовать общее хранилище, а гостевые системы смогут обращаться к устройствам хранения с помощью своих файлов конфигурации.

Многопутевая настройка

В системах с несколькими физическими путями к устройствам хранения используется пакет **multipath**, обеспечивающий высокую отказоустойчивость и производительность сетевых устройств хранения, подключенных к системам Linux.

Чтобы обеспечить сохранение постоянства LUN в окружении **multipath**, необходимо присвоить псевдонимы многопутевым устройствам. Каждому устройству хранения соответствует UUID, который выполняет функции ключа для создаваемых имен. Определить UUID устройства можно с помощью команды **scsi_id**.

```
# scsi_id -g -s /block/sdc
```

Многопутевые устройства создаются в каталоге **/dev/mpath**. В приведенном ниже примере будет определено 4 устройства в файле **/etc/multipath.conf**:

```
multipaths {
        multipath {
        wwid                    3600805f300159870000000000768a0019
        alias                    oramp1
        }
        multipath {
        wwid                    3600805f300159870000000000d643001a
        alias                    oramp2
```

```
        }
        mulitpath {
        wwid                 3600805f3001598700000000086fc001b
        alias                 oramp3
        }
        mulitpath {
        wwid                 3600805f3001598700000000000984001c
        alias                 oramp4
        }
}
```

В результате будет создано четыре LUN с именами **/dev/mpath/oramp1**, **/dev/mpath/oramp2**, **/dev/mpath/oramp3** и **/dev/mpath/oramp4**. Теперь сопоставление идентификаторов именам будет сохраняться между перезагрузками.

4.4. Добавление виртуализированного устройства CD-ROM или DVD в гостевую систему

Для подключения ISO-файла гостевую систему, в то время пока она в онлайне, используйте команду **virsh** с параметром *attach-disk*.

```
# virsh attach-disk [ID_домена] [источник] [цель] --driver file --type
 cdrom --mode readonly
```

Параметры *источник* и *цель* представляют собой пути к файлам и устройствам в размещающей и гостевой системах (соответственно). Параметр *цель* может представлять собой путь к ISO-файлу или устройству из каталога **/dev**.

Виртуализация и общие хранилища данных

В этой главе рассматривается использование общего сетевого хранилища в окружении виртуализации Fedora.

Общее хранилище может быть организовано следующими способами:

- Fibre Channel
- iSCSI
- NFS
- GFS2

Наличие сетевого хранилища является обязательным требованием для выполнения автономной и живой миграции.

5.1. Использование iSCSI для хранения гостей

Здесь рассматривается размещение виртуальных машин на iSCSI-устройствах.

5.2. Использование NFS для хранения гостей

Здесь рассматривается размещение виртуальных машин в подключенных с помощью NFS ресурсах.

5.3. Использование GFS2 для хранения гостей

Здесь рассматривается размещение виртуальных машин в файловой системе GFS2 (Global File System 2).

Рекомендации для сервера

Приведенные далее советы и рекомендации помогут обеспечить безопасность и надежность сервера Fedora (dom0).

- Включите принудительный режим SELinux, выполнив команду

```
# setenforce 1
```

- Удалите или отключите службы, в которых нет необходимости, такие как **`AutoFS`**, **`NFS`**, **`FTP`**, **`HTTP`**, **`NIS`**, **`telnetd`**, **`sendmail`** и пр.
- Добавьте лишь минимально число учетных записей, которое требуется для управления платформой, а также удалите ненужные записи.
- Постарайтесь не выполнять посторонние приложения на узле, так как это может негативно сказаться на производительности виртуальной машины и подвергнуть риску стабильность работы сервера.
- Образы виртуальных машин следует сохранить в **`/var/lib/libvirt/images/`**. Если же вы используете другой каталог, убедитесь, что это отражено в настройках политики SELinux.
- Источники установки, иерархии каталогов и установочные образы должны быть сохранены в одном месте -- обычно там, где расположен сервер `vsftpd`.

Виртуализация и безопасность

При реализации технологии виртуализации в корпоративной инфраструктуре необходимо обеспечить безопасность размещающей системы. Домен 0 является привилегированным доменом, выполняющим функции управления системой и виртуальными машинами. Если он не защищен, все остальные домены в системе подвергаются риску. Существует несколько способов усиления защиты систем, утилизирующих виртуализацию. Сначала разработайте комплексный *план по развертыванию*, содержащий описание спецификаций и служб, обязательных для виртуальных машин и размещающих серверов, а также информацию о том, что необходимо для поддержки этих служб. Ниже перечислены наиболее важные принципы защиты, которые следует принять во внимание при разработке такого плана.

- Убедитесь, что на главном узле выполняется минимально необходимое число служб. Чем меньше заданий и служб выполняется в домене 0, тем меньше он подвергается риску.
- Включите *SELinux* для гипервизора (см. *Раздел 7.1, «Виртуализация и SELinux»*).
- Используйте межсетевой экран для ограничения трафика к dom0. Межсетевой экран может быть настроен на автоматический отказ для усиления защиты. Также рекомендуется ограничить число служб, использующих сетевое подключение.
- Не открывайте доступ к dom0 для обычных пользователей. Помните, домен 0 является привилегированным, а разрешение доступа непривилегированных пользователей может подвергнуть его безопасность неоправданному риску.

7.1. Виртуализация и SELinux

Механизм SELinux (Security Enhanced Linux) изначально был разработан Агентством национальной безопасности США при содействии сообщества Linux. SELinux ограничивает доступ процессов к ресурсам в зависимости от прав доступа пользователя и позволяет предотвратить большинство угроз безопасности, в том числе переполнение буфера и эскалации привилегий. Поэтому Fedora рекомендует использование строгого режима SELinux во всех системах Linux.

SELinux не позволит загрузить образы виртуальных машин, если они не расположены в каталоге **`/var/lib/libvirt/images`**.

Добавление хранилища LVM при включенном SELinux

Дальше будет рассмотрен пример добавления логического тома в виртуальную машину, где SELinux работает в строгом режиме. Приведенные инструкции также подходят для разделов жестких дисков.

Процедура 7.1. Создание и монтирование логического тома

1. Создайте логический том. Команда создания тома размером 5 гигабайт в группе томов *`группа_томов`* будет выглядеть так:

   ```
   # lvcreate -n новый_том -L 5G группа_томов
   ```

2. Отформатируйте *`новый_том`* и создайте файловую систему, поддерживающую расширенные атрибуты, например, ext3.

```
# mke2fs -j /dev/volumegroup/новый_том
```

3. Создайте новый каталог, куда будет смонтирован новый логический том. Этот каталог может расположен в любом месте, хотя не рекомендуется его размещать в системных каталогах (**/etc**, **/var**, **/sys**) и домашних каталогах (**/home** или **/root**).

```
# mkdir /virtstorage
```

4. Смонтируйте созданный том.

```
# mount /dev/volumegroup/новый_том /virtstorage
```

5. Определите тип SELinux для каталога Xen:

```
semanage fcontext -a -t xen_image_t "/virtualization(/.*)?"
```

Или для каталога KVM:

```
semanage fcontext -a -t virt_image_t "/virtualization(/.*)?"
```

При использовании целевой политики (целевая политика используется по умолчанию) команда добавит дополнительную строку в файл **/etc/selinux/targeted/contexts/files/file_contexts.local**. Добавленное выражение обеспечит постоянство этих изменений.

```
/virtstorage(/.*)?    system_u:object_r:xen_image_t:s0
```

6. Измените тип точки монтирования (**/virtstorage**) и всех файлов в этом каталоге на **xen_image_t** (**restorecon** и **setfiles** позволяют обращаться к файлам в **/etc/selinux/targeted/contexts/files/**).

```
# restorecon -R -v /virtualization
```

7.2. Замечания о SELinux

Эта секция содержит информацию, которую важно помнить при реализации SELinux в окружении виртуализации. Не забывайте обновлять политику SELinux, если вы внесли изменения в систему или добавили новые устройства. Чтобы настроить том LVM для виртуальной машины, необходимо изменить контекст SELinux для соответствующего блочного устройства и группы томов.

```
# semanage fcontext -a -t xen_image _t -f -b /dev/sda2
# restorecon /dev/sda2
```

Логический параметр **xend_disable_t** переводит **xend** в незащищенный режим после его перезапуска. Если вы решили отключить защиту, лучше это сделать отдельно для демона, а

не для всей системы. Не рекомендуется задавать метку **xen_image_t** для каталогов, которые планируется использовать для других целей.

Настройка сетевого окружения

В этой главе рассматриваются варианты конфигурации сетевого окружения, используемые приложениями на основе libvirt. Приведенная здесь информация применима ко всем гипервизорам, в том числе Xen и KVM. Дополнительно можно обратиться к документации libvirt.

Две типичные конфигурации позволяют настроить виртуальную сеть или общее физическое устройство. Конфигурация виртуальной сети будет одинакова для всех дистрибутивов, ее можно сразу применять. А общее физическое устройство будет нуждаться в дополнительной настройке для разных дистрибутивов.

8.1. Преобразование сетевых адресов с помощью libvirt

Преобразование сетевых адресов (NAT, Network Address Translation) используется довольно часто для организации общих сетевых соединений (виртуальных сетей).

Настройка размещающей системы

Любая стандартная установка `libvirt` включает возможности подключения к виртуальным машинам на основе NAT. Чтобы в этом убедиться, выполните команду **`virsh net-list --all`**.

```
# virsh net-list --all
Name                 State      Autostart
-----------------------------------------
default              active     yes
```

Если запись «default» отсутствует, попробуйте подключить следующий файл конфигурации:

```
# virsh net-define /usr/share/libvirt/networks/default.xml
```

`/usr/share/libvirt/networks/default.xml` содержит определения сетевого окружения, используемого по умолчанию.

Настройте автоматический запуск:

```
# virsh net-autostart default
Network default marked as autostarted
```

Запустите сеть:

```
# virsh net-start default
Network default started
```

Вы должны увидеть изолированный мост. К этому устройству *не* подключены физические интерфейсы, так как оно использует перенаправление IP и NAT для подключения к внешней сети. Не добавляйте новые интерфейсы.

```
# brctl show
```

```
bridge name     bridge id               STP enabled     interfaces
virbr0          8000.000000000000       yes
```

libvirt добавит правила **iptables**, разрешающие прохождение трафика гостевых систем, подключенных к устройству virbr0 в цепочках **INPUT**, **FORWARD**, **OUTPUT** и **POSTROUTING**. Затем **libvirt** попытается включить параметр **ip_forward**. Но другие программы могут его отключить, поэтому в файл **/etc/sysctl.conf** стоит добавить выражение

```
net.ipv4.ip_forward = 1
```

Настройка гостевой системы

После завершения настройки размещающей системы можно подключить гостевую систему к виртуальной сети. Подключить гостевую систему к сети «default» можно с помощью следующего файла:

```
<interface type='network'>
  <source network='default'/>
</interface>
```

Note

Можно дополнительно определить MAC-адрес, иногда есть смысл задать его вручную. Если этого не сделать, он просто будет сгенерирован автоматически.

```
<interface type='network'>
  <source network='default'/>
  <mac address='00:16:3e:1a:b3:4a'/>
  </interface>
```

8.2. Мостовое соединение с помощью libvirt

Мостовое соединение используется для выделения физического устройства виртуальной машине и часто применяется для более тонкой настройки серверов с многочисленными сетевыми интерфейсами.

Отключите сетевые сценарии Xen

Если ваша система использует мост Xen, рекомендуется его отключить. Для этого в файле **/etc/xen/xend-config.sxp** измените строку

```
(network-script network-bridge)
```

на

```
(network-script /bin/true)
```

Отключите NetworkManager

NetworkManager не поддерживает мостовое подключение и должен быть отключен, если вы планируете использовать сетевое окружение со старыми сетевыми сценариями.

```
# chkconfig NetworkManager off
# chkconfig network on
# service NetworkManager stop
# service network start
```

Note

Вместо отключения NetworkManager можно добавить параметр «*NM_CONTROLLED=no*» в сценарии **ifcfg-***.

Создание сценариев инициализации сети

Создайте или отредактируйте указанные ниже файлы конфигурации сети. Повторите для каждого дополнительного сетевого моста (изменив имя).

Перейдите в каталог **/etc/sysconfig/network-scripts**.

```
# cd /etc/sysconfig/network-scripts
```

Откройте сценарий для добавляемого устройства. В приведенном примере **ifcfg-eth0** содержит определение физического сетевого интерфейса, входящего в состав моста:

```
DEVICE=eth0
# измените аппаратный адрес, чтобы он соответствовал адресу сетевой карты
HWADDR=00:16:76:D6:C9:45
ONBOOT=yes
BRIDGE=br0
```

Подсказка

Можно настроить максимальный размер блока передачи (MTU, Maximum Transfer Unit), добавив переменную *MTU* в конец файла конфигурации.

```
MTU=9000
```

Создайте новый сценарий с именем **ifcfg-br0** (или аналогичным названием) в каталоге **/etc/sysconfig/network-scripts**. *br0* обозначает имя моста и может иметь любую длину, главное — чтобы эта часть имени файла совпадала с параметром DEVICE.

```
DEVICE=br0
TYPE=Bridge
BOOTPROTO=dhcp
ONBOOT=yes
```

```
DELAY=0
```

Warning

The line, *`TYPE=Bridge`*, is case-sensitive. It must have uppercase 'B' and lower case 'ridge'.

Завершив настройку, перезапустите службу сети или перезагрузите компьютер.

```
# service network restart
```

Configure **`iptables`** to allow all traffic to be forwarded across the bridge.

```
# iptables -I FORWARD -m physdev --physdev-is-bridged -j ACCEPT
# service iptables save
# service iptables restart
```

Disable iptables on bridges

Alternatively, prevent bridged traffic from being processed by **`iptables`** rules. In **`/etc/sysctl.conf`** append the following lines:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

Reload the kernel parameters configured with **`sysctl`**

```
# sysctl -p /etc/sysctl.conf
```

Restart the **`libvirt`** daemon.

```
# service libvirtd reload
```

Теперь должно быть доступно общее физическое устройство, которое гости могут подключить. Проверьте его наличие:

```
# brctl show
bridge name     bridge id               STP enabled     interfaces
virbr0          8000.000000000000       yes
br0             8000.000e0cb30550       no              eth0
```

Обратите внимание, что созданный мост совершенно не зависит от **`virbr0`**. *НЕ* пытайтесь подключить физическое устройство к **`virbr0`**. Мост **`virbr0`** используется исключительно для NAT.

Паравиртуализированные драйверы KVM

Виртуализированным гостевым системам Windows на узлах KVM доступны паравиртуализированные драйверы в составе пакета virtio. Пакет virtio поддерживает блочные устройства хранения и контроллеры сетевых интерфейсов.

Паравиртуализированные драйверы позволяют повысить производительность полностью виртуализированных гостевых систем, снизить задержки операций ввода-вывода и значительно увеличить скорость обработки.

Паравиртуализированные драйверы KVM загружаются и устанавливаются автоматически в поздних версиях Fedora. Никаких дополнительных действий предпринимать не требуется.

Аналогично модулю KVM, драйверы virtio доступны на узлах с поздними версиями Fedora.

Note

Каждому гостю доступно 28 гнезд PCI для дополнительных устройств. Каждая паравиртуализированная сеть или блочное устройство будет занимать отдельное гнездо. Таким образом, гостевая система может использовать максимум 28 дополнительных устройств, любую комбинацию паравиртуализированных сетевых и дисковых устройств, а также других PCI-устройств, использующих VTd.

Системы Microsoft Windows, которые поддерживают паравиртуализированные драйверы KVM:

- Windows XP,
- Windows Server 2003,
- Windows Vista,
- Windows Server 2008.

9.1. Установка паравиртуализированных драйверов Windows

В этой секции будет рассмотрен процесс установки паравиртуализированных драйверов KVM для работы Windows. Их можно загрузить во время установки Windows и установить после установки гостевой системы.

Паравиртуализированные драйверы можно установить несколькими способами:

- можно разместить установочные файлы в сети для сетевой установки,
- можно выполнить установку с виртуализированного устройства CD-ROM или ISO-образа,
- можно установить драйверы во время загрузки из виртуализированного дисковода (подходит для гостевых систем Windows).

В этом руководстве будет описан процесс установки с паравиртуализированного установочного диска, функционирующего как CD-ROM.

1. **Загрузите драйверы**
   Драйверы также можно найти на сайте *windowsservercatalog.com*[1].

   Пакет *virtio-win* установит образ **`virtio-win.iso`** в каталог **`/usr/share/virtio-win/`**.

2. **Установите паравиртуализированные драйверы**
   Рекомендуется сначала установить драйверы в гостевой системе, а уже после этого подключать или изменять устройства с целью использования паравиртуализированных драйверов.

   Для блочных устройств, на которых расположены корневые файловые системы или другие блочные устройства, необходимые для загрузки гостя, потребуется установить драйверы, прежде чем приступить к изменению настроек устройства.

Монтирование образа с помощью virt-manager

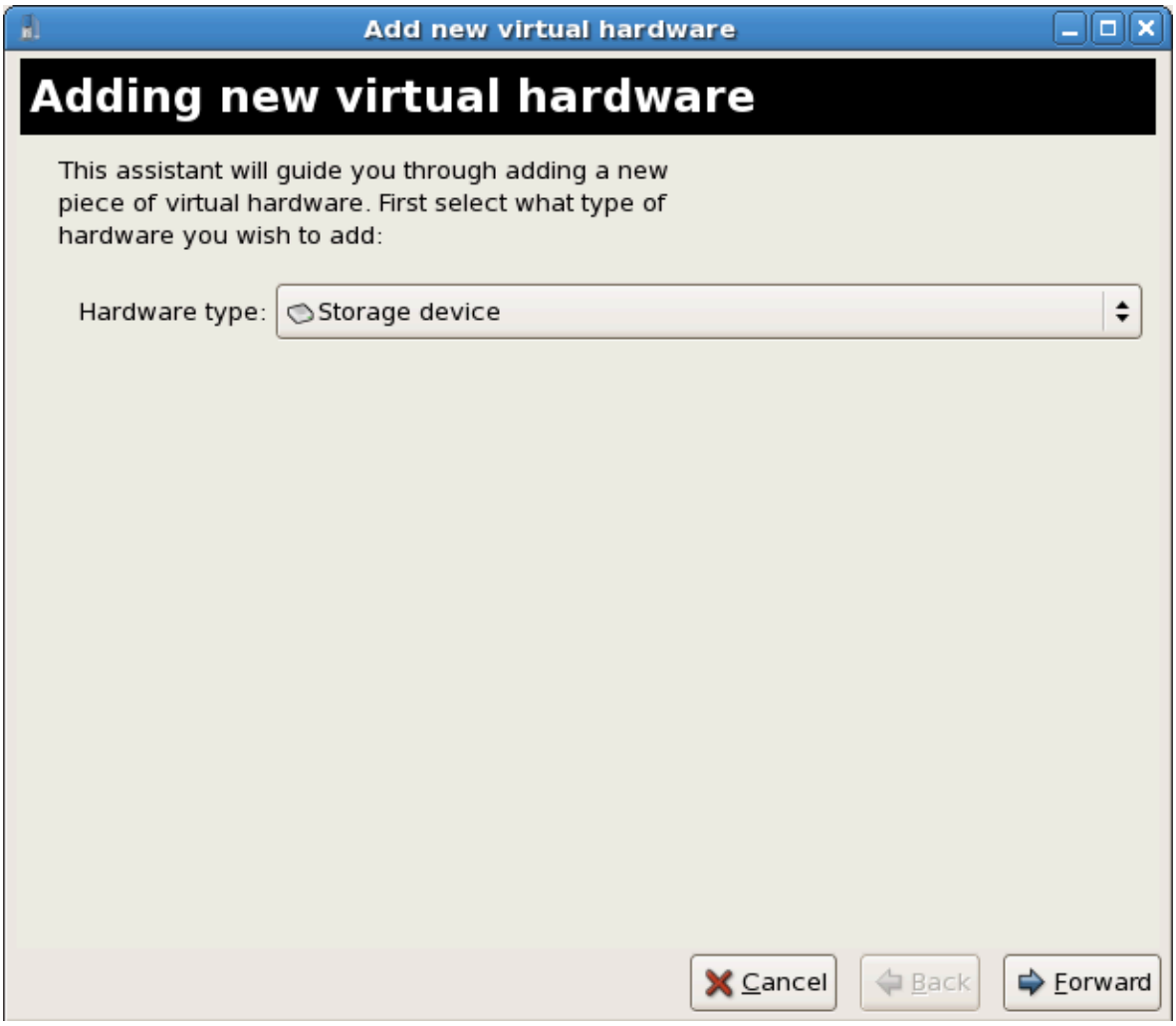
*Процедура 9.1, «Монтирование образа CD-ROM для гостя Windows с помощью **`virt-manager`** »* содержит инструкции по добавлению образа CD-ROM с помощью **`virt-manager`**.

Процедура 9.1. Монтирование образа CD-ROM для гостя Windows с помощью `virt-manager`

1. Запустите **`virt-manager`**, выберите гостевую систему из списка и нажмите кнопку **Сведения**.

2. Нажмите кнопку добавления на открывшейся панели.

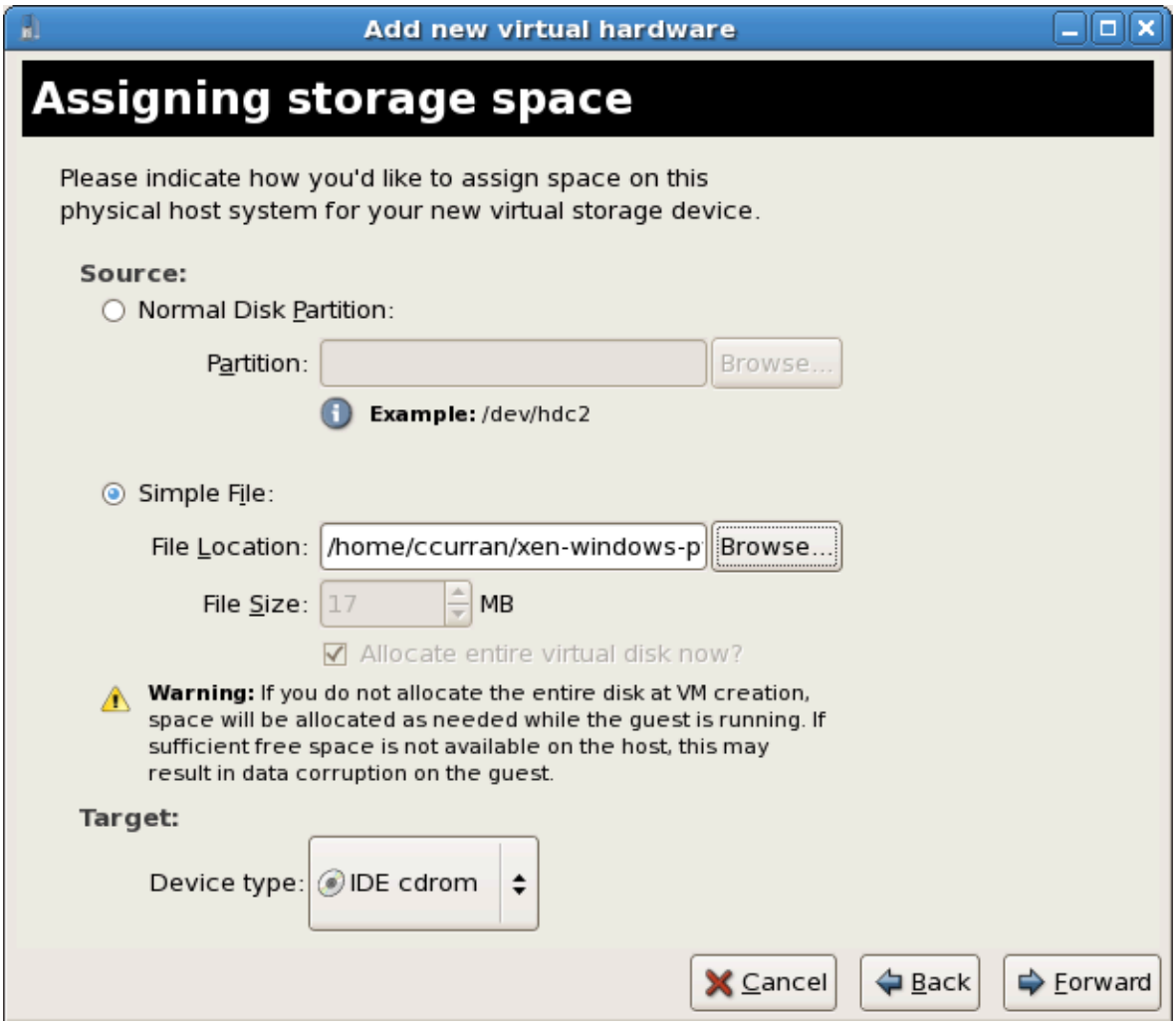
3. Откроется окно мастера добавления устройств. Выберите устройство хранения из выпадающего списка и нажмите кнопку продолжения.



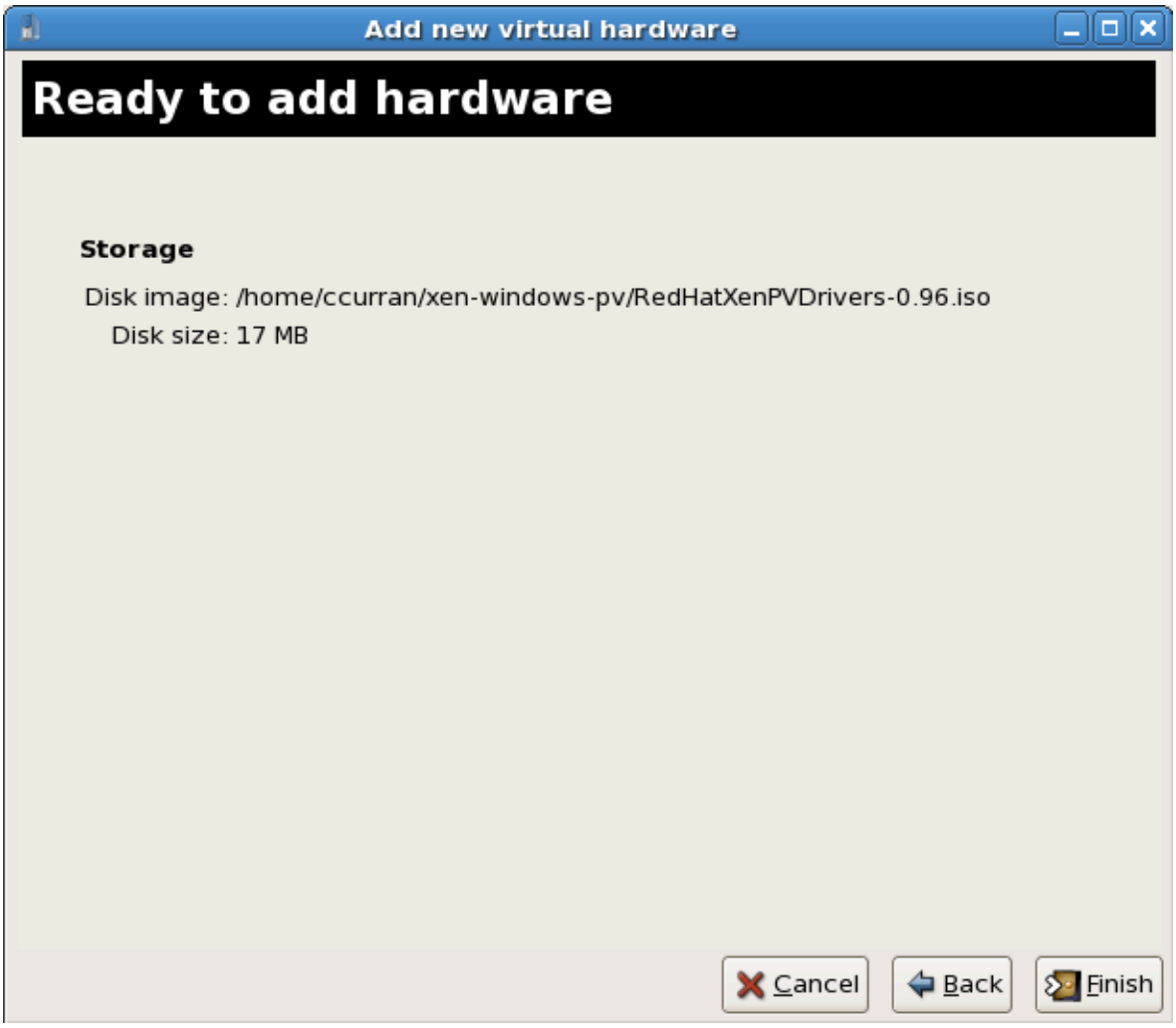
4. Отметьте **Файл (образ диска)** и выберите файл *.iso, нажав кнопку **Обзор...**. Если паравиртуализированные драйверы были установлены с помощью **`yum`**, то нужный файл будет расположен в **`/usr/share/xenpv-win`**.

   Если драйверы хранятся на обычном компакт-диске, выберите **Обычный дисковый раздел**.

   В списке **Тип устройства** выберите **IDE cdrom** и нажмите кнопку продолжения.



5. Диск будет доступен гостевой системе в момент ее запуска. Закройте окно мастера, нажав кнопку завершения.



Установка из виртуализированного дисковода

Здесь рассматривается установка паравиртуализированных драйверов в процессе установки Windows.

- Во время первой установки виртуальной машины Windows подключите **`viostor.vfd`** как дисковод.

  a. **Windows Server 2003**
  Когда Windows предложит нажать F6, чтобы задать сторонние драйверы, следуйте инструкциям.

  b. **Windows Server 2008**
  При запросе драйверов нажмите «Загрузить драйвер», перейдите к диску A: и выберите подходящий для вашей операционной системы драйвер.

Использование паравиртуализированных драйверов KVM для существующих устройств

Измените существующий жесткий диск, подключенный к гостю, чтобы вместо виртуализированного драйвера IDE использовался драйвер **`virtio`**. Ниже будут

внесены изменения в файлы конфигурации libvirt. Новое устройство с использованием паравиртуализированных драйверов также можно добавить с помощью **`virt-manager`** или команд **`virsh attach-disk`** и **`virsh attach-interface`** (см. *Использование паравиртуализированных драйверов KVM для новых устройств*).

1. Типичная запись для виртуализированного гостя, не использующего паравиртуализированные драйверы, которая определяет блочное устройство на основе файла выглядит примерно так:

```
<disk type='file' device='disk'>
   <source file='/var/lib/libvirt/images/disk1.img'/>
   <target dev='hda' bus='ide'/>
</disk>
```

2. Присвойте **`bus=`** значение **`virtio`**, чтобы использовать паравиртуализированное устройство.

```
<disk type='file' device='disk'>
   <source file='/var/lib/libvirt/images/disk1.img'/>
   <target dev='hda' bus='virtio'/>
</disk>
```

Использование паравиртуализированных драйверов KVM для новых устройств

Здесь рассматривается процесс создания новых устройств, использующих паравиртуализированные драйверы KVM, с помощью **`virt-manager`**.

Новое устройство с использованием паравиртуализированных драйверов также можно добавить с помощью команд **`virsh attach-disk`** и **`virsh attach-interface`**.

Сначала установите драйверы

Прежде чем приступить к установке новых устройств, убедитесь, что в гостевой системе Windows уже установлены драйверы. В противном случае устройство не будет обнаружено и, как следствие, работать не будет.

1. Открыть виртуализированную гостевую систему можно дважды щелкнув мышью на ее имени в **`virt-manager`**.

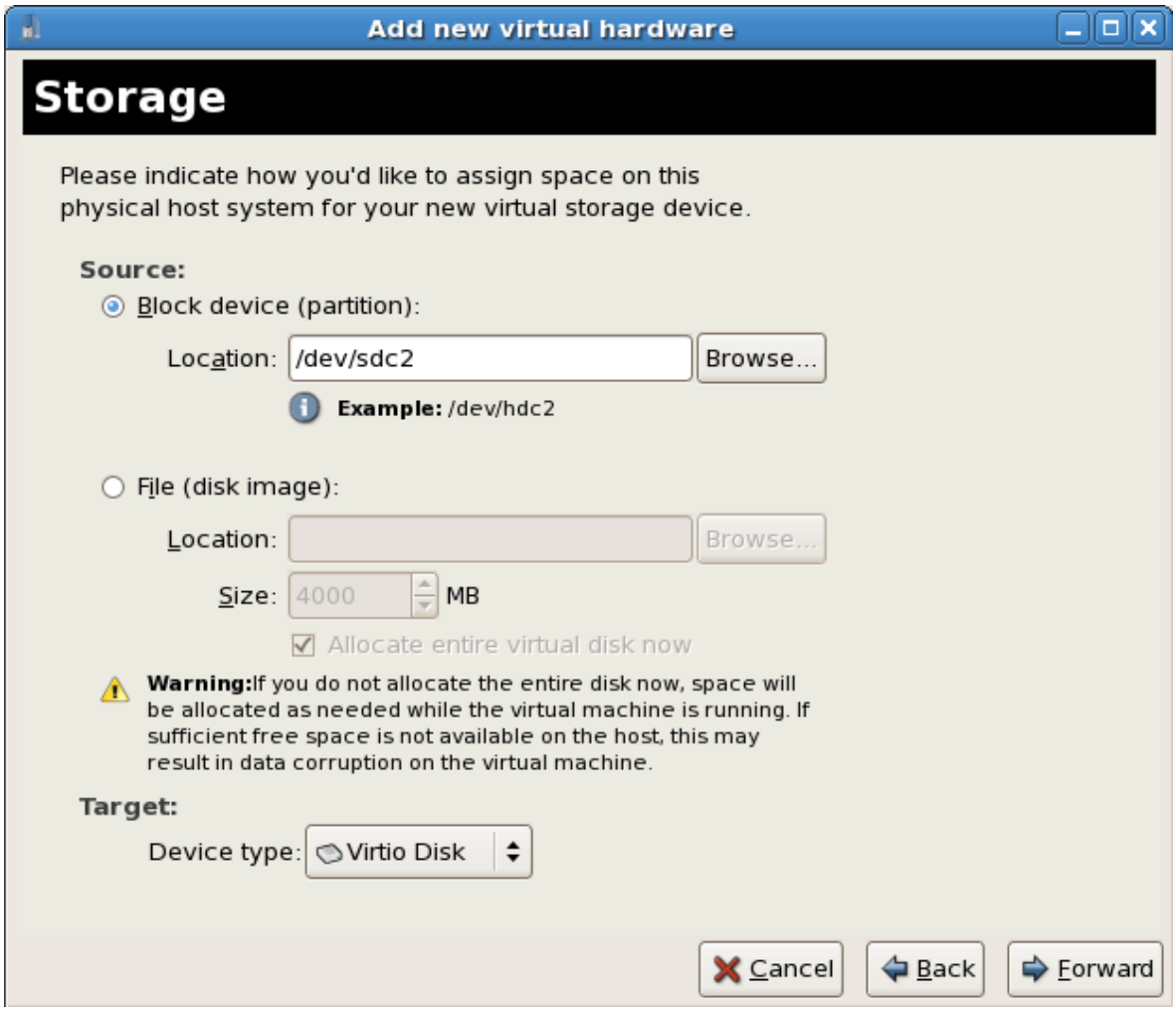
2. Перейдите на вкладку **Оборудование**.

3. Нажмите кнопку **Добавить оборудование**.

4. На открывшейся вкладке выберите **Накопители** или **Сеть**.

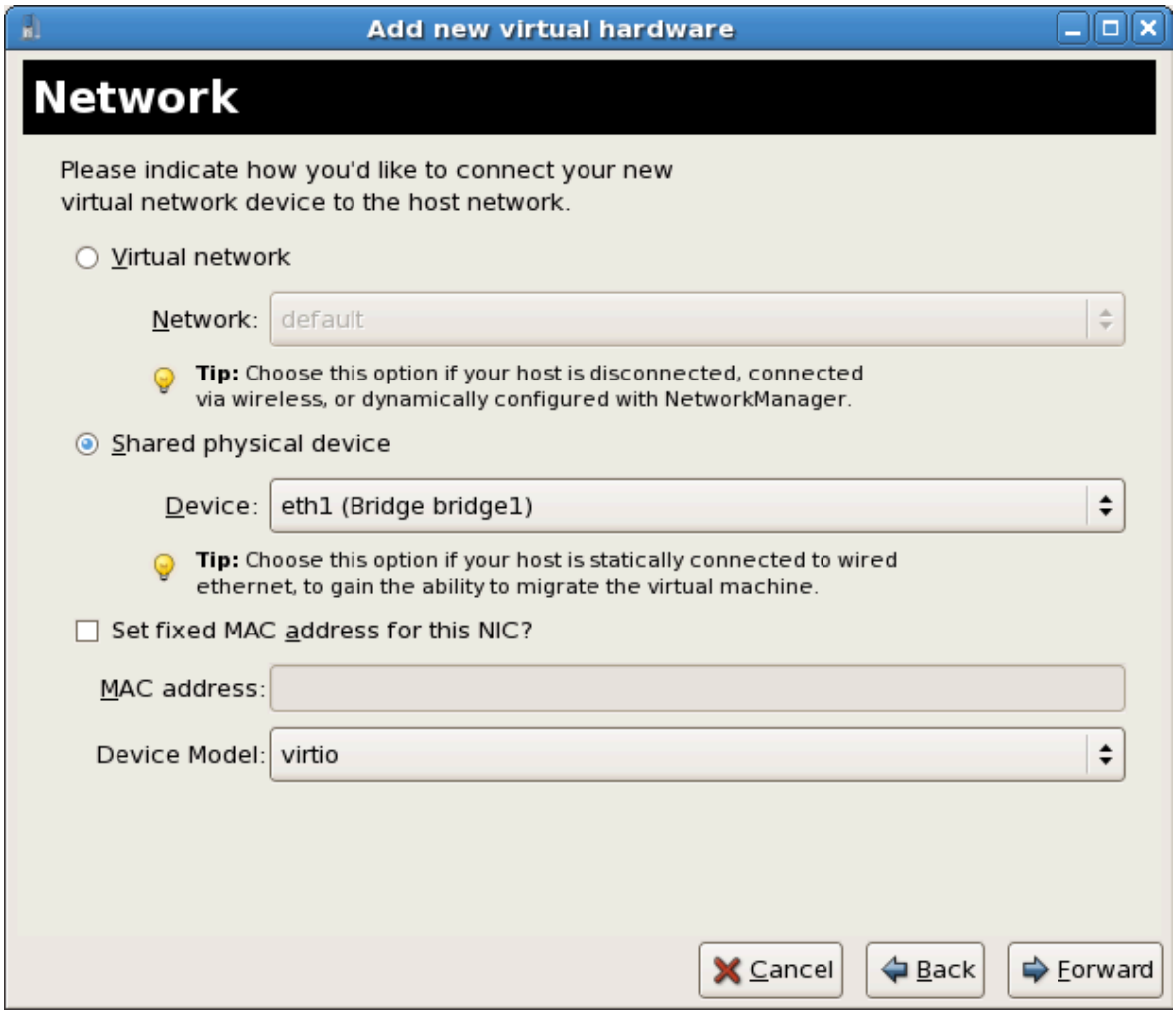
   1. **Новые дисковые устройства**

      Выберите диск или файл образа. В качестве типа устройства укажите диск virtio и нажмите кнопку продолжения.

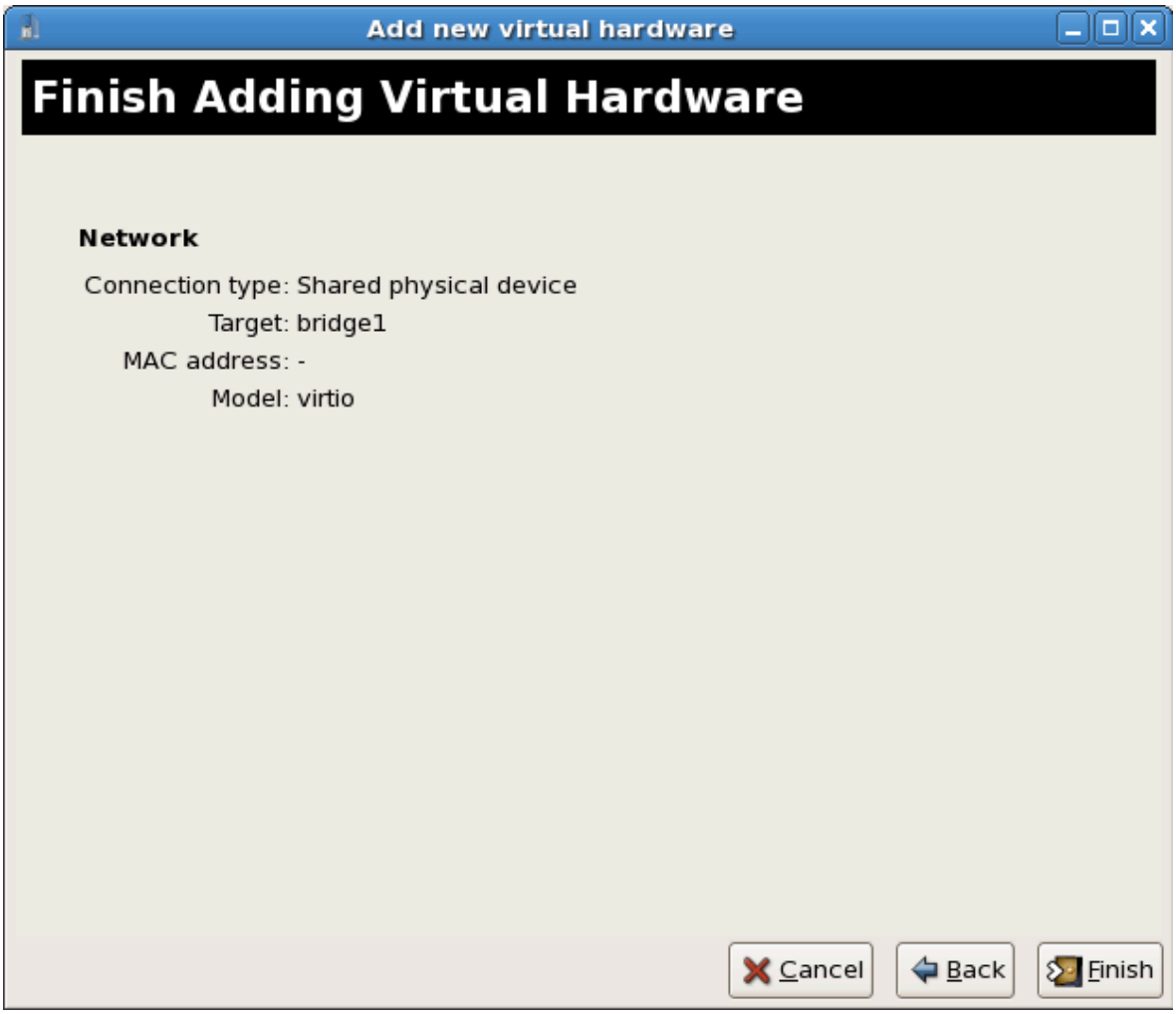


2. **Новые сетевые устройства**

   Выберите **Виртуальное устройство** или **Общее физическое устройство**. В качестве типа устройства укажите **virtio** и нажмите кнопку продолжения.



5. Чтобы сохранить устройство, нажмите **Завершить**.



6. Перезагрузите гостевую систему, чтобы убедиться, что Windows распознает устройство.

Часть III. Administration

Управление виртуализированными системами

Последующие главы предоставляют информацию по управлению размещающими и гостевыми системами с помощью стандартных инструментов виртуализации Fedora.

Управление гостевыми системами с помощью xend

Демон **xend** выполняет определенные функции по управлению виртуальными машинами, в том числе осуществляет контроль виртуализированных ресурсов. Для взаимодействия с виртуальными машинами необходимо, чтобы **xend** работал. Прежде чем вы запустите **xend**, задайте параметры в файле конфигурации **`/etc/xen/xend-config.sxp`**. Ниже приведены доступные параметры.

Параметр	Description
(console-limit)	Определяет лимит буфера памяти сервера консоли xend_unix_server и задает значение для каждого домена отдельно.
(min-mem)	Определяет минимальный объем памяти в мегабайтах, выделяемый домену 0. Если указано значение 0, объем не меняется.
(dom0-cpus)	Определяет число процессоров, которые домен 0 сможет использовать. По умолчанию назначается 1.
(enable-dump)	Определяет, выполнять ли дамп ядра при сбое. По умолчанию установлено в 0.
(external-migration-tool)	Задает имя приложения или сценария, отвечающего за миграцию внешних устройств. Сценарии должны располагаться в **`etc/xen/scripts/external-device-migrate`**.
(logfile)	Задает расположение файла журнала. По умолчанию журнал будет сохранен в **`/var/log/xend.log`**.
(loglevel)	Устанавливает уровень критичности сообщений, записываемых в журнал. Доступные значения: DEBUG, INFO, WARNING, ERROR, CRITICAL. По умолчанию используется DEBUG.
(network-script)	Задает сценарий, активирующий сетевое окружение. Сценарии должны располагаться в каталоге **`etc/xen/scripts`**.
(xend-http-server)	Определяет, активировать ли HTTP-сервер управления пакетами. По умолчанию сервер отключен.
(xend-unix-server)	Определяет, активировать ли сокет-сервер домена. Сокет-сервер представляет собой конечную точку, где обрабатываются сетевые соединения низкого уровня, которая разрешает или запрещает входящие подключения. Значение по умолчанию — yes.

Параметр	Description
(xend-relocation-server)	Активирует сервер перемещения для поддержки миграции между машинами. По умолчанию сервер отключен.
(xend-unix-path)	Задает расположение вывода данных команды **`xend-unix-server`**. По умолчанию вывод будет сохранен в **`var/lib/xend/xend-socket`**.
(xend-port)	Определяет порт, используемый HTTP-сервером. По умолчанию используется порт 8000.
(xend-relocation-port)	Определяет порт, используемый сервером перемещения. По умолчанию используется порт 8002.
(xend-relocation-address)	Определяет адреса узлов, которым разрешена миграция. По умолчанию используется значение параметра `xend-address`.
(xend-address)	Определяет адрес, которому сопоставлен сокет-сервер. По умолчанию разрешены все подключения.

Таблица 10.1. Параметры конфигурации **xend**

Установив параметры, запустите демон **xend**. Для его запуска выполните:

```
service xend start
```

Остановка **xend**:

```
service xend stop
```

Эта команда остановит xend, если он запущен.

Перезапуск **xend**:

```
service xend restart
```

Эта команда перезапустит xend, даже если он уже работает.

Также можно проверить состояние **xend**:

```
service xend status
```

Отображает состояние демона.

Активация xend во время загрузки

С помощью команды **`chkconfig`** добавьте xend в **`initscript`**.

```
chkconfig --level 345 xend
```

xend будет запущен на уровнях выполнения 3, 4, 5.

Управление временем виртуальных машин KVM

KVM использует счетчик тактов процессора TSC (Time Stamp Counter). Некоторые процессоры не поддерживают постоянную частоту счетчика, что негативно скажется на работе виртуальных машин на основе KVM. Неточности могут привести к замедлению или, наоборот, ускорению работы сетевых приложений, что изменить скорость работы самой виртуальной машины.

Потенциальные проблемы, связанные с неточностью часов и счетчиков процессора:

- Нарушение синхронизации часов может привести к несвоевременному завершению сеансов и сказаться на работе сети;
- Виртуальные машины с замедленными часами могут столкнуться с проблемами при миграции;
- Возможен сбой виртуальных машин.

Эти проблемы существуют и на других платформах виртуализации, поэтому настоятельно рекомендуется тестировать счетчики и часы.

NTP

В размещающей и гостевых системах должен быть активен сетевой протокол NTP (Network Time Protocol). Команда запуска:

```
# service ntpd start
```

Добавьте службу ntpd в последовательность загрузки:

```
# chkconfig ntpd on
```

В большинстве случаев служба `ntpd` поможет минимизировать последствия расхождения часов.

Определение наличия постоянного счетчика TSC

Наличие постоянного счетчика TSC подтверждается флагом `constant_tsc`. Чтобы узнать, есть ли флаг `constant_tsc`:

```
$ cat /proc/cpuinfo | grep constant_tsc
```

Непустой вывод команды подтверждает наличие бита `constant_tsc`. Если же вывод пуст, обратитесь к приведенным ниже инструкциям.

Настройка узлов без постоянного счетчика TSC

Для процессоров, не поддерживающих постоянную частоту счетчика TSC, потребуется дополнительная настройка. Возможности управления питанием для виртуальных машин

потребуется отключить, так как они отрицательно сказываются на синхронизации времени с KVM.

Note

Эти инструкции применимы только для процессоров AMD F.

В случае отсутствия бита `constant_tsc` отключите все возможности управления питанием (*BZ#513138*[1]). Каждая система обладает несколькими таймерами, которые используются для синхронизации. В размещающей системе частота счетчика TSC может колебаться вследствие изменений **`cpufreq`**, перехода в Deep C-состояние или миграции на узел с более быстрым TSC. Для отключения Deep C-состояния остановите счетчик, добавьте **`processor.max_cstate=1`** в строку параметров grub на узле:

```
term Fedora (vmlinuz-2.6.29.6-217.2.3.fc11)
        root (hd0,0)
        kernel /vmlinuz-vmlinuz-2.6.29.6-217.2.3.fc11 ro root=/dev/
VolGroup00/LogVol00 rhgb quiet processor.max_cstate=1
```

Отключите **`cpufreq`** (только если отсутствует бит **`constant_tsc`**). Для этого в файле **`/etc/sysconfig/cpuspeed`** измените значения переменных **`MIN_SPEED`** и **`MAX_SPEED`** на максимально возможное значение частоты. Диапазоны частот можно найти в файлах **`/sys/devices/system/cpu/cpu*/cpufreq/scaling_available_frequencies`**.

Использование паравиртуализированных часов в гостевых системах Red Hat Enterprise Linux

В некоторых гостевых системах Red Hat Enterprise Linux потребуется настроить дополнительные параметры ядра. Их можно добавить в конец строки **`/kernel`** файла **`/boot/grub/grub.conf`**, расположенного в гостевой системе.

Приведенная ниже таблица содержит перечень версий Red Hat Enterprise Linux и параметров ядра, необходимых для виртуальных машин в системах, не поддерживающих постоянную частоту счетчика TSC.

Red Hat Enterprise Linux	Дополнительные параметры ядра виртуальной машины
5.4 AMD64/Intel 64 с паравиртуализированными часами	Дополнительные параметры не нужны
5.4 AMD64/Intel 64 без паравиртуализированных часов	divider=10 notsc lpj=n
5.4 x86 с паравиртуализированными часами	Дополнительные параметры не нужны
5.4 x86 без паравиртуализированных часов	divider=10 clocksource=acpi_pm lpj=n

[1] https://bugzilla.redhat.com/show_bug.cgi?id=513138

Red Hat Enterprise Linux	Дополнительные параметры ядра виртуальной машины
5.3 AMD64/Intel 64	divider=10 notsc
5.3 x86	divider=10 clocksource=acpi_pm
4.8 AMD64/Intel 64	notsc divider=10
4.8 x86	clock=pmtmr divider=10
3.9 AMD64/Intel 64	Дополнительные параметры не нужны
3.9 x86	Дополнительные параметры не нужны

Использование паравиртуализированных часов в гостевых системах Windows

Параметры загрузки Windows расположены в файле boot.ini. Следующий параметр позволяет использовать таймер PM-TIMER вместо счетчика TSC.

```
/use pmtimer
```

Дальнейшую информацию о параметрах загрузки Windows можно найти на странице *Параметры, используемые в файле Boot.ini в Windows XP и Windows Server 2003*[2].

[2] http://support.microsoft.com/kb/833721

Живая миграция KVM

В этой главе будет рассмотрен процесс живой миграции гостевых систем с гипервизора KVM на другой узел KVM.

Под миграцией понимается процесс переноса виртуализированной гостевой системы с одного узла на другой. Миграция является основополагающим аспектом виртуализации, так как на этом уровне программное обеспечение совершенно не зависит от оборудования. Основное назначение миграции:

- Load balancing - guests can be moved to hosts with lower usage when a host becomes overloaded.
- Hardware failover - when hardware devices on the host start to fail, guests can be safely relocated so the host can be powered down and repaired.
- Energy saving - guests can be redistributed to other hosts and host systems powered off to save energy and cut costs in low usage periods.
- Geographic migration - guests can be moved to another location for lower latency or in serious circumstances.

Миграция может быть выполнена в автономном режиме или подключенном режиме (так называемая «живая» миграция). В процессе миграции память гостевой системы передается на целевой узел; при этом файловая система гостя будет сохранена в общем хранилище (она не будет передаваться целевому узлу по сети).

An offline migration suspends the guest then moves an image of the guests memory to the destination host. The guest is resumed on the destination host and the memory the guest used on the source host is freed.

Длительность автономной миграции зависит от полосы пропускания и сетевой задержки. Так, перенос гостевой системы с 2 Гбайт памяти по 1 гигабит Ethernet займет около 10 секунд.

Живая миграция характеризуется тем, что работа виртуальных машин не останавливается при переносе. Все изменяемые за это время страницы памяти отслеживаются и передаются целевому узлу после завершения передачи образа. Процесс продолжается до тех пор, пока не будут скопированы все страницы или пока не истечет заданный гипервизором KVM период времени. Если страницы источника изменяются слишком быстро, то работа гостя на исходном узле будет приостановлена и будет выполнена передача регистров и буферов. Регистры будут загружены на новом узле и гость возобновит работу на целевом узле. Если же синхронизация невозможна, что вероятно в случае большой нагрузки, то виртуальная машина будет приостановлена для выполнения миграции в автономном режиме.

Длительность такой миграции зависит от полосы пропускания, сетевой задержки и активности гостевой системы. Нагрузка на процессор и большие объемы операций ввода-вывода также могут сказаться на длительности процесса.

12.1. Требования живой миграции

Ниже перечислены требования для успешного выполнения миграции.

Требования миграции

- Виртуализированный гость на общем устройстве хранения, использующий один из следующих протоколов:
  - Fibre Channel
  - iSCSI
  - NFS
  - GFS2
- Как минимум две системы Fedora одной версии с одними и теми же обновлениями.
- Обе системы должны открыть соответствующие порты.
- Сетевая конфигурация обеих систем должна совпадать.
- В исходной и целевой системах общее хранилище должно быть смонтировано в одну и ту же точку. Путь к также должен совпадать.

Настройка сетевого хранилища

Настройте общее хранилище и установите в нем виртуальную машину. *Глава 5, Виртуализация и общие хранилища данных* содержит инструкции.

Или же можно попробовать использовать рассмотренный в этом руководстве пример NFS (см. *Раздел 12.2, «Пример общего хранилища: Упрощение миграции за счет NFS»*).

12.2. Пример общего хранилища: Упрощение миграции за счет NFS

В рассмотренном ниже примере совместный доступ узлов KVM к образам гостевых систем будет обеспечен за счет NFS. Этот пример не подходит для масштабных установок, его целью является лишь демонстрация процесса миграции, поэтому не стоит его использовать для миграции большого количества виртуализированных гостей.

Для этого обратитесь к инструкциям, приведенным здесь: *Глава 5, Виртуализация и общие хранилища данных*

1. **Экспортируйте каталог с образом libvirt**
   Добавьте каталог с образом в файл **`/etc/exports`**:

   ```
   /var/lib/libvirt/images *.bne.redhat.com(rw,no_root_squash,async)
   ```

   Замените *`*.bne.redhat.com`* необходимым именем узла.

2. **Запустите NFS**
   a. Если пакеты NFS еще не установлены, установите их:

   ```
   # yum install nfs
   ```

   b. В **`iptables`** откройте порты для NFS и добавьте NFS в файл **`/etc/hosts.allow`**.

c. Запустите службу:

```
# service nfs start
```

3. **Смонтируйте общее хранилище**
   Смонтируйте **/var/lib/libvirt/images** в целевой системе:

```
# mount URL_на_исходном_узле:/var/lib/libvirt/images /var/lib/libvirt/
images
```

> **Расположение каталога должно совпадать на обоих узлах**
> Независимо от выбранного каталога, он должен располагаться в одном и том же месте в отправляющей и получающей системах.

12.3. Живая миграция с помощью virsh

Гостевую систему можно перенести на другой узел с помощью команды **virsh**. Ее аргумент **migrate** принимает параметры в следующем формате:

```
# virsh migrate --live GuestName DestinationURL
```

The *GuestName* parameter represents the name of the guest which you want to migrate.

The *DestinationURL* parameter is the URL or hostname of the destination system. The destination system must run the same version of Fedora, be using the same hypervisor and have **libvirt** running.

Once the command is entered you will be prompted for the root password of the destination system.

Пример живой миграции с помощью virsh

Этот пример демонстрирует перенос виртуальной машины **CentOS4test** с узла `test1.bne.redhat.com` на `test2.bne.redhat.com`.

Подразумевается, что общее хранилище уже настроено и удовлетворяет всем требованиям (см. *Требования миграции*).

1. **Убедитесь, что гость работает**
   Убедитесь, что `CentOS4test` выполняется на `test1.bne.redhat.com`:

```
[root@test1 ~]# virsh list
Id Name                 State
----------------------------------
 10 CentOS4                 running
```

2. **Можно приступить к миграции**
   Выполните приведенную ниже команду, чтобы начать перенос гостя на `test2.bne.redhat.com`. В конец ссылки добавьте **`/system`**, чтобы сообщить libvirt о необходимости получения полного доступа.

   ```
   # virsh migrate --live CentOS4test qemu+ssh://test2.bne.redhat.com/
   system
   ```

   Once the command is entered you will be prompted for the root password of the destination system.

3. **Подождите**
   Процесс миграции может занять некоторое время в зависимости от нагрузки и размера гостя. **`virsh`** будет сообщать только об ошибках. Гость будет продолжать работу на исходном узле до завершения переноса.

4. **Проверьте результат переноса**
   Убедитесь, что `CentOS4test` выполняется на `test2.bne.redhat.com`:

   ```
   [root@test2 ~]# virsh list
   Id Name                 State
   ----------------------------------
    10 CentOS4                 running
   ```

Живая миграция успешно завершена.

Другие сетевые механизмы
libvirt поддерживает множество сетевых механизмов, включая TLS/SSL, сокеты unix, SSH, TCP без шифрования. *Глава 13, Удаленное управление виртуализированными гостевыми системами* содержит подробную информацию.

12.4. Миграция с помощью virt-manager

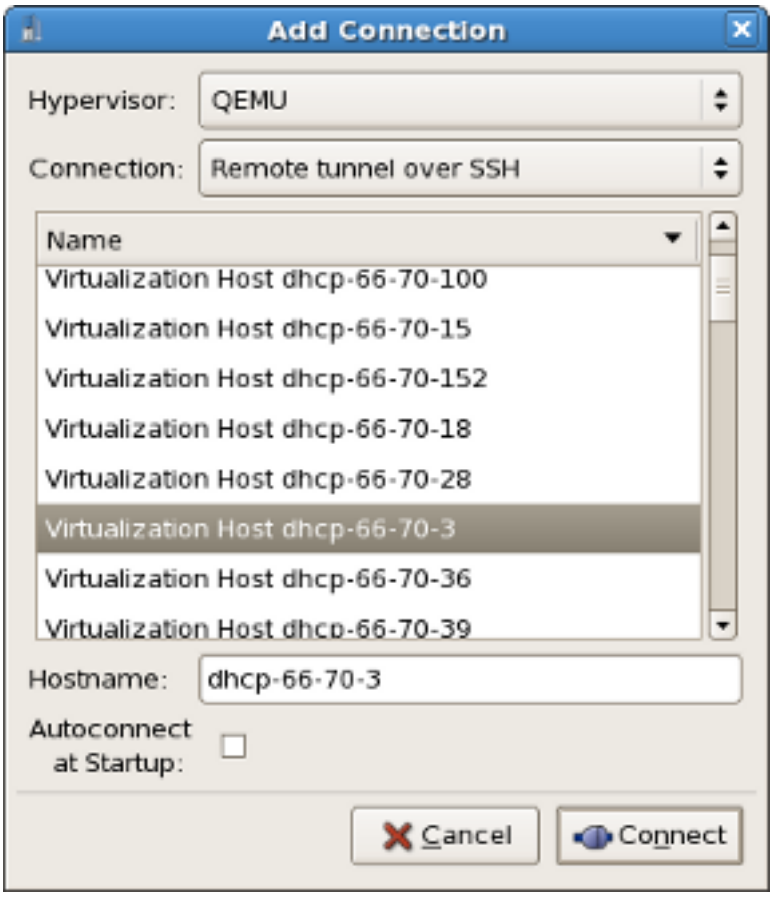
В этой секции рассматривается миграция гостей KVM с помощью **`virt-manager`**.

1. Подключитесь к отправляющей и получающей системам. В меню **Файл** выберите **Добавить соединение**.

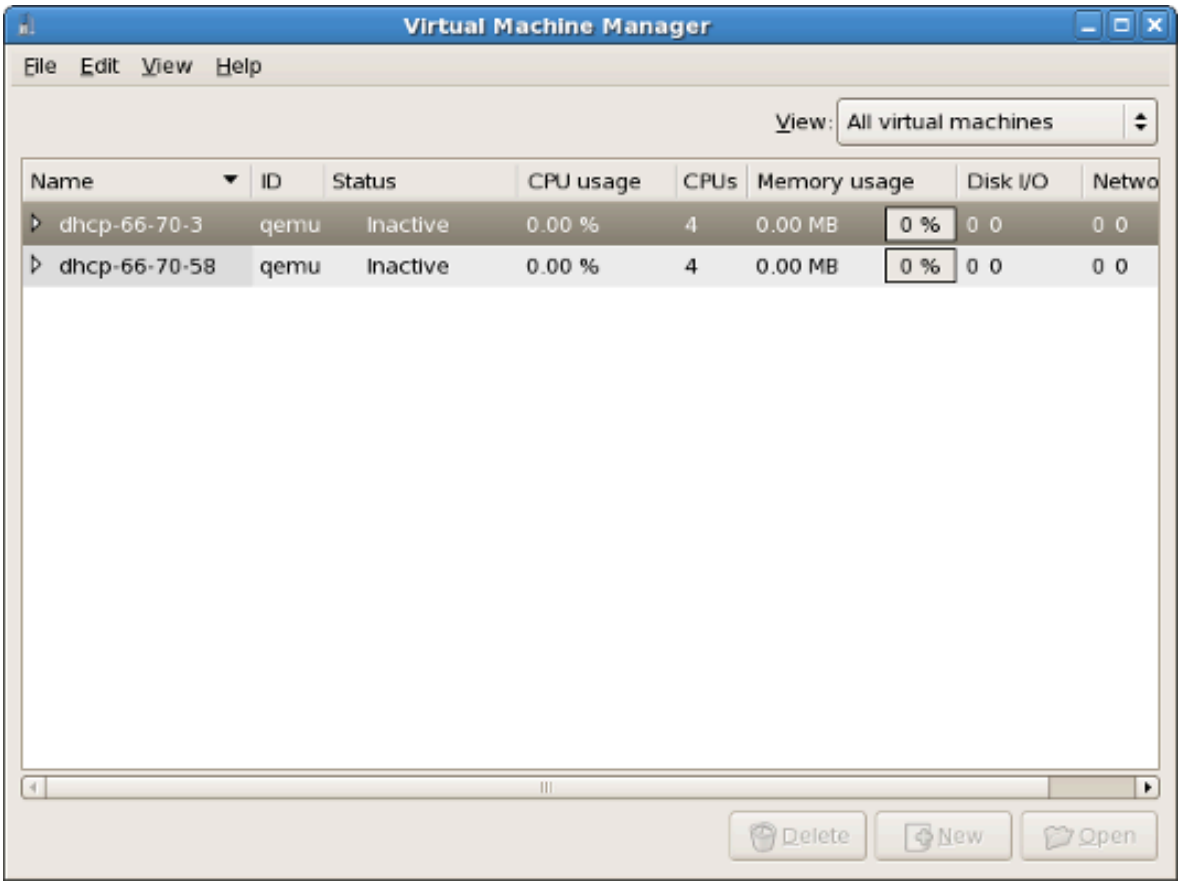
   В открывшемся окне измените следующее:

   - **Гипервизор:** Выберите **QEMU**.
   - **Соединение:** Выберите тип соединения.
   - **Имя узла:** Введите имя узла.

   Нажмите кнопку подключения.



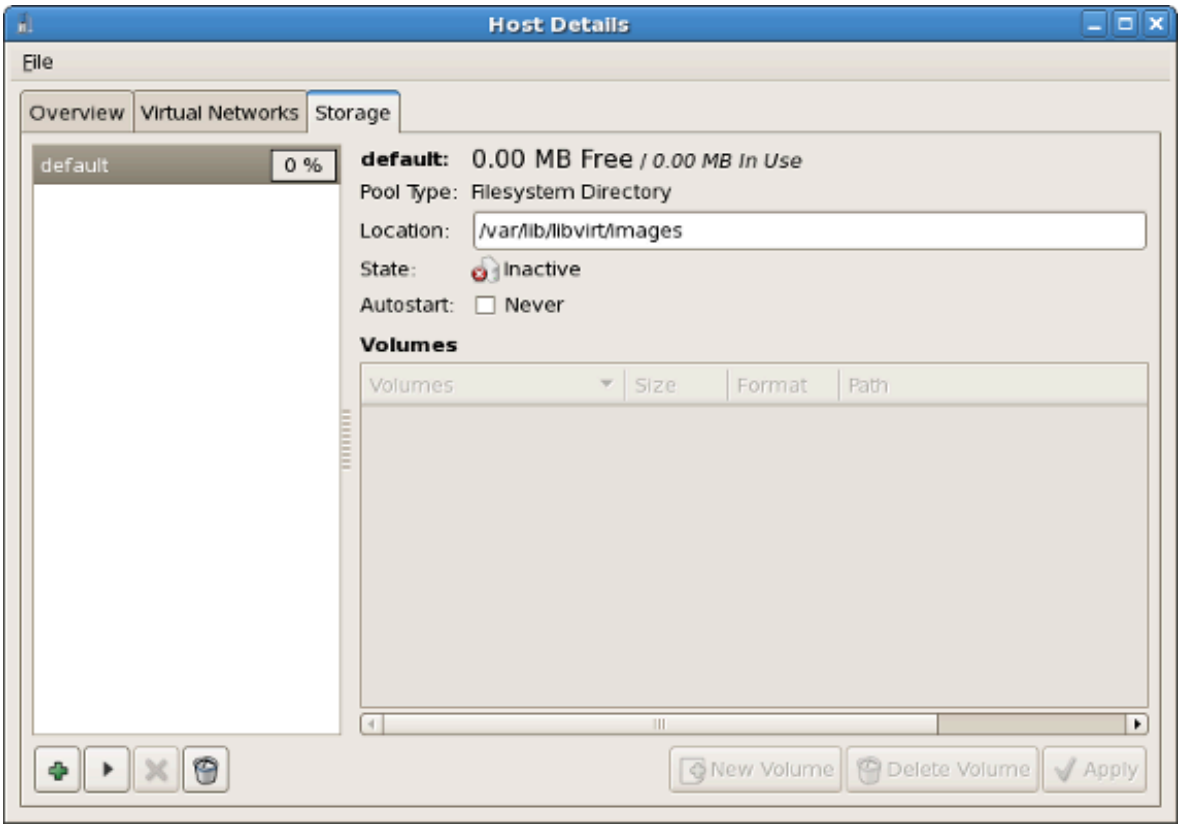
Менеджер виртуальных машин покажет список подключенных узлов.



2. Добавьте общий пул хранилищ.

   В меню правки выберите пункт сведений об узле.

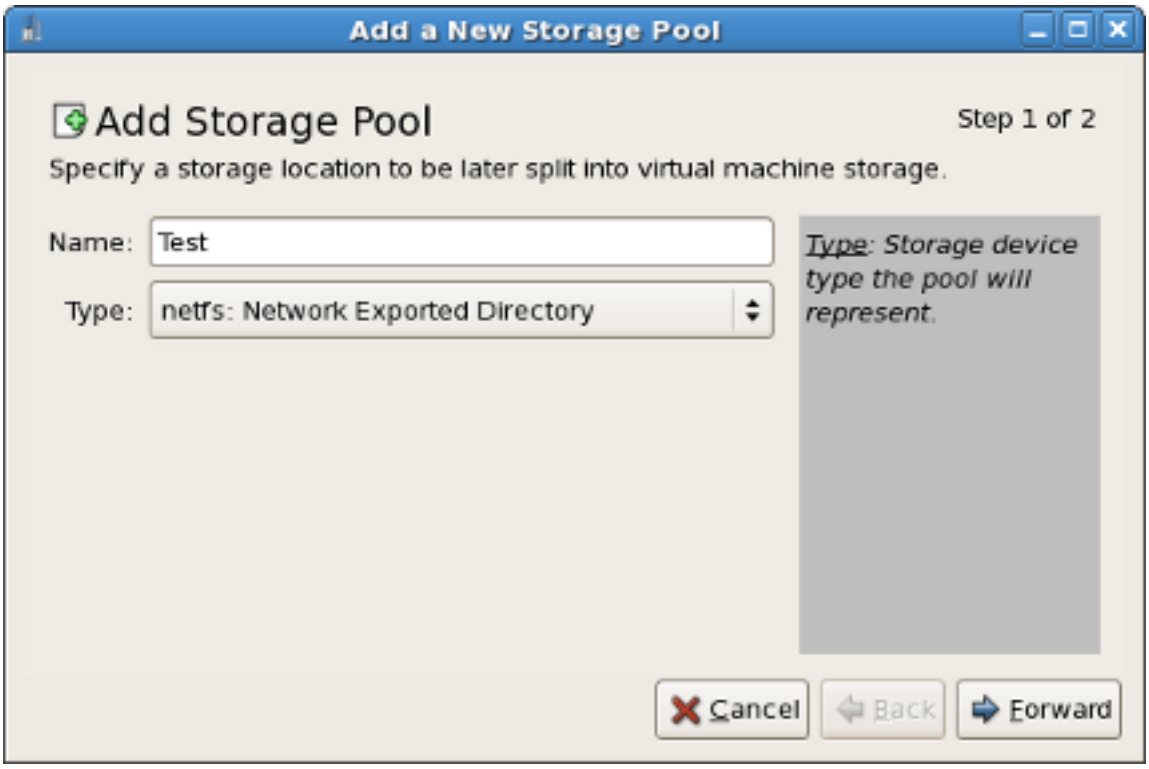
   Перейдите на вкладку **Хранилище**.



3. Чтобы добавить пул хранилищ, в левом нижнем углу нажмите кнопку **+**. В открывшемся окне укажите следующие параметры:

   В открывшемся окне измените следующее:

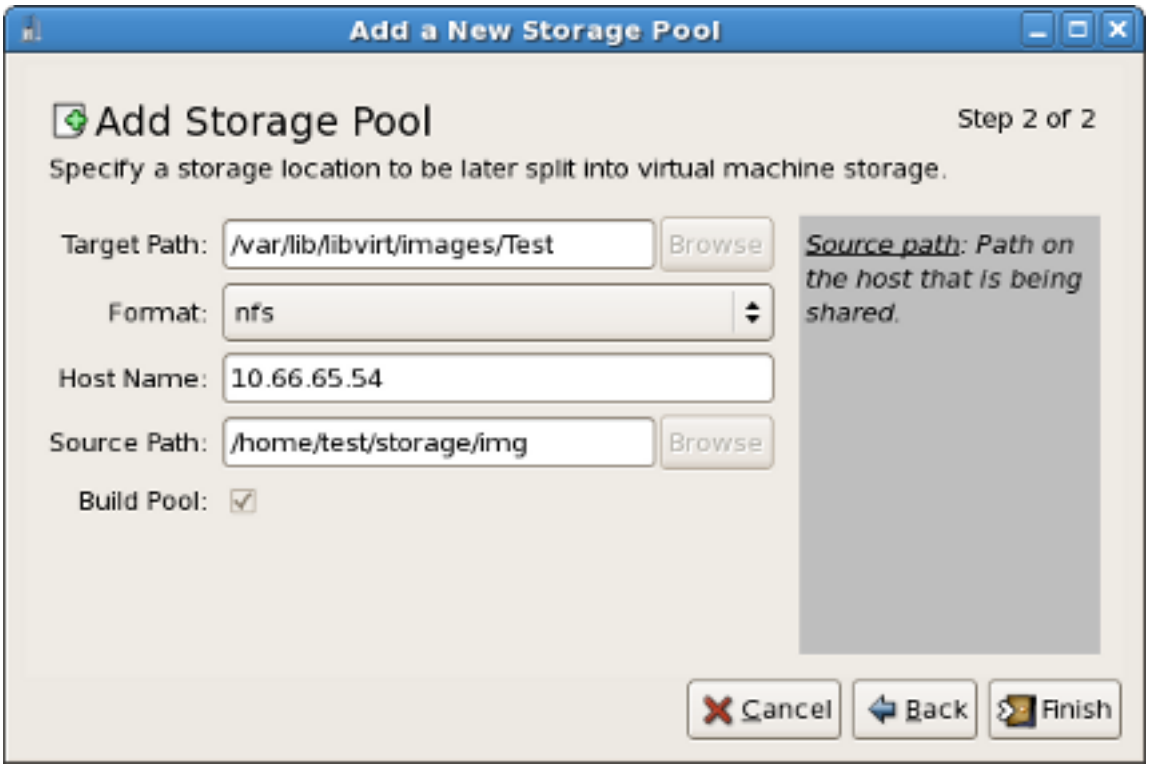
   - **Имя:** Введите имя для создаваемого пула.
   - **Тип:** Выберите **netfs: Network Exported Directory**.



Нажмите кнопку **Далее**.

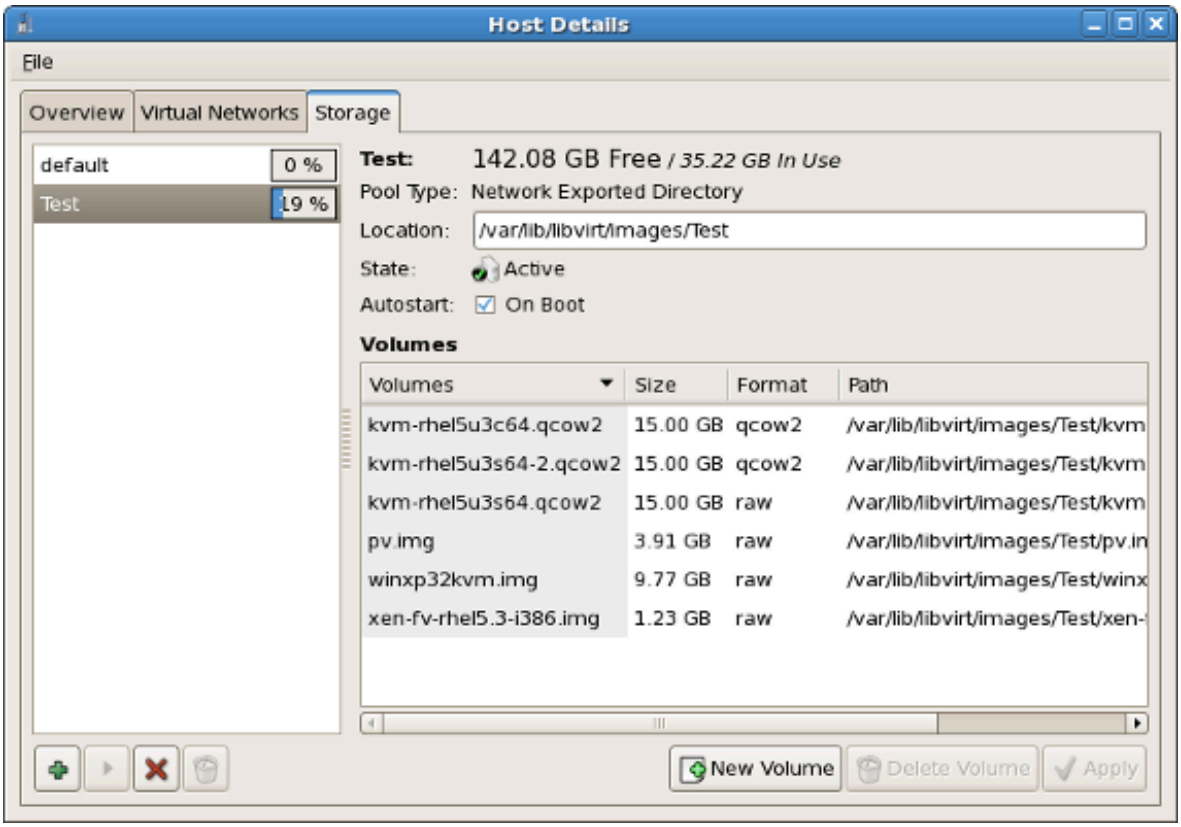
4. В открывшемся окне измените следующее:

   - **Формат:** Для живой миграции выберите NFS или iSCSI.
   - **Имя узла:** Введите адрес IP или полностью квалифицированное имя домена сервера хранилища.

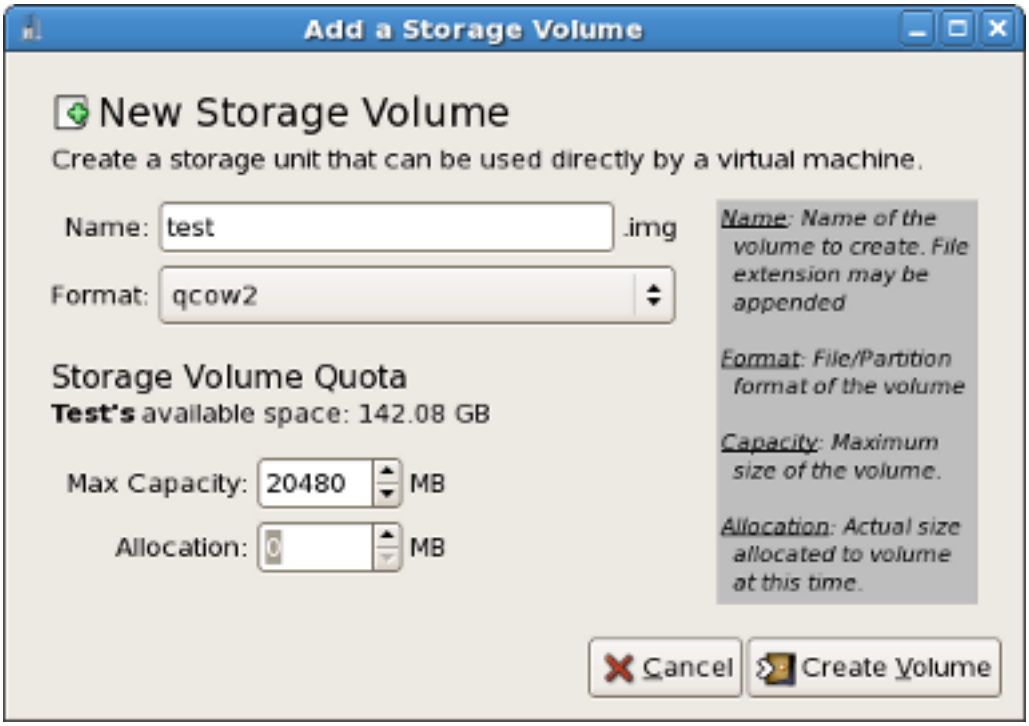


Нажмите кнопку **Готово**.

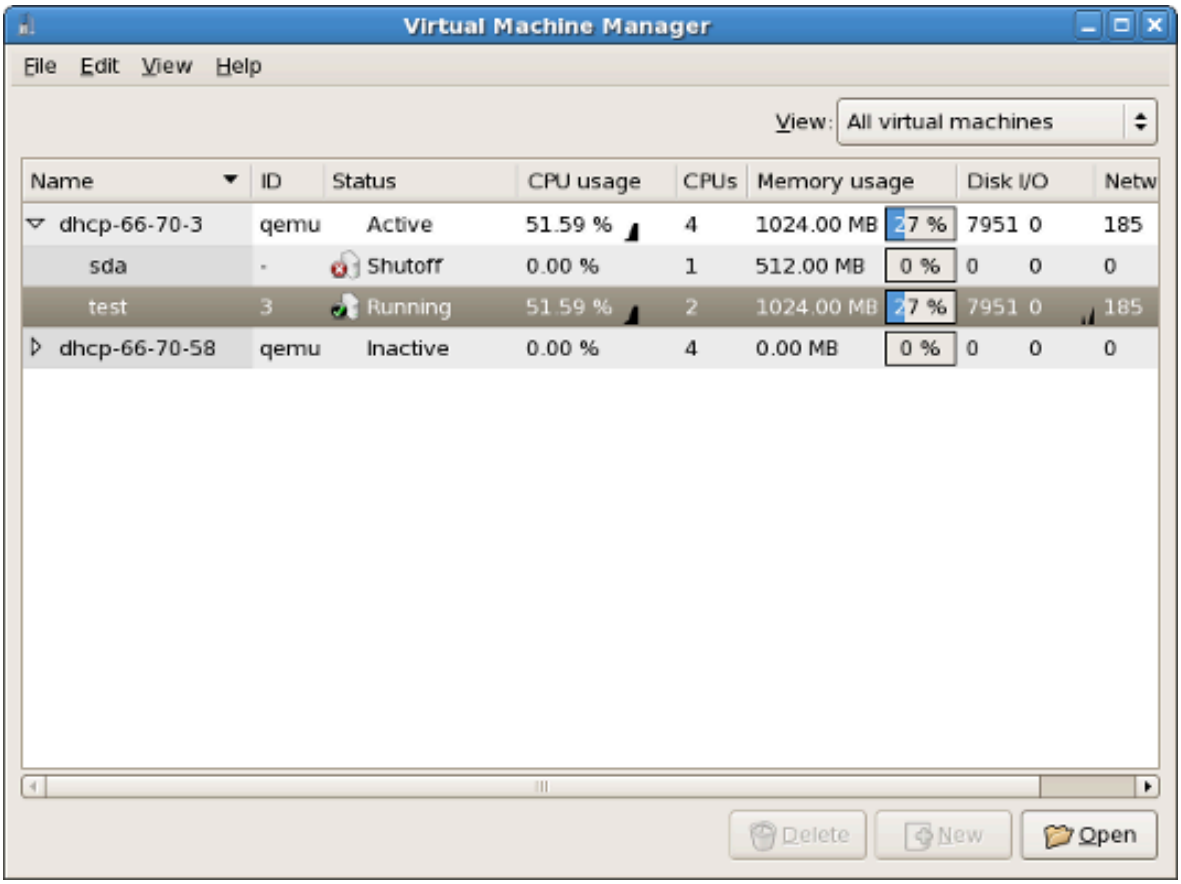
5. Нажмите кнопку **Новый том**, чтобы создать том в общем пуле.



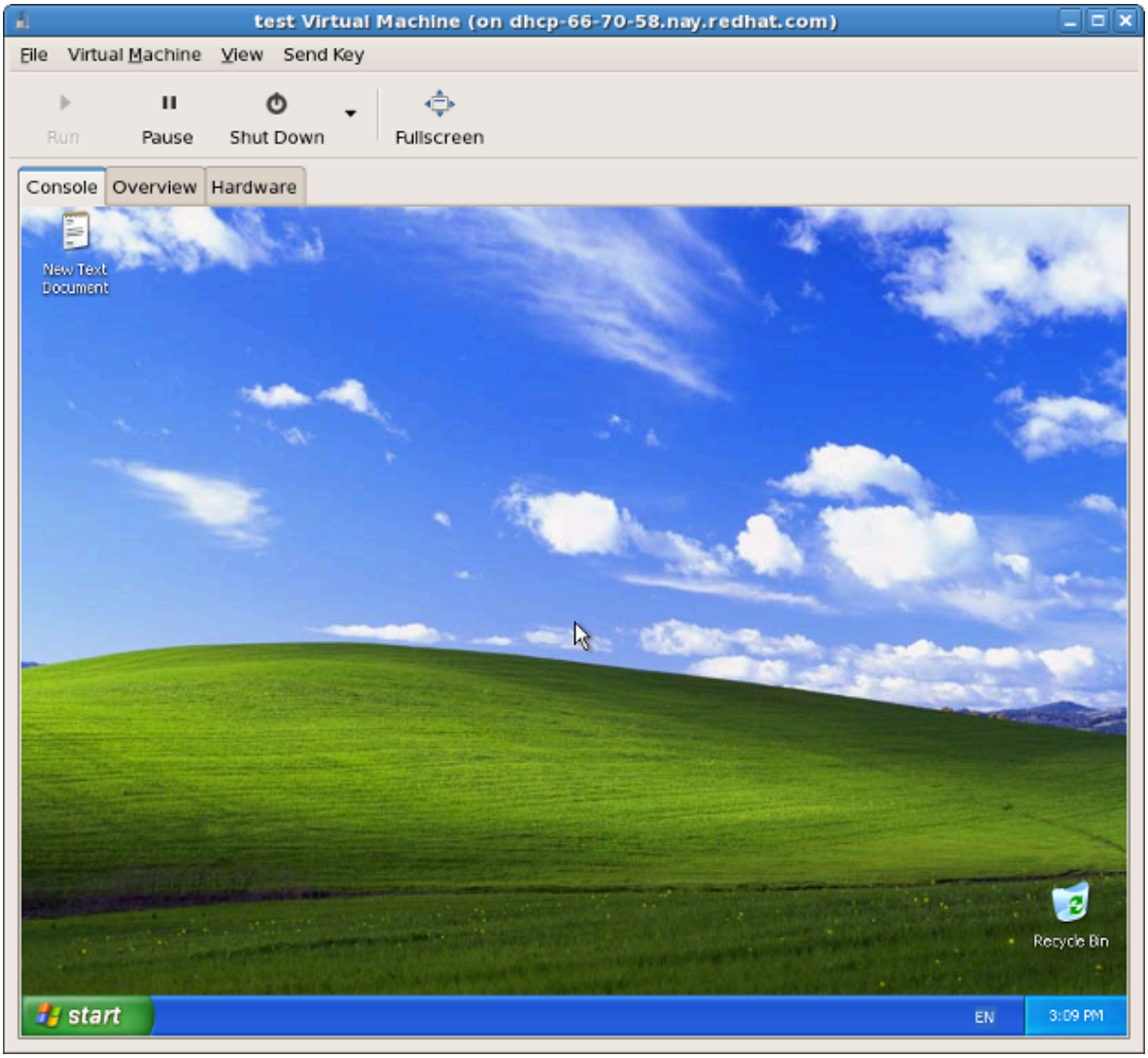
6. Заполните поля и нажмите **Создать том**.



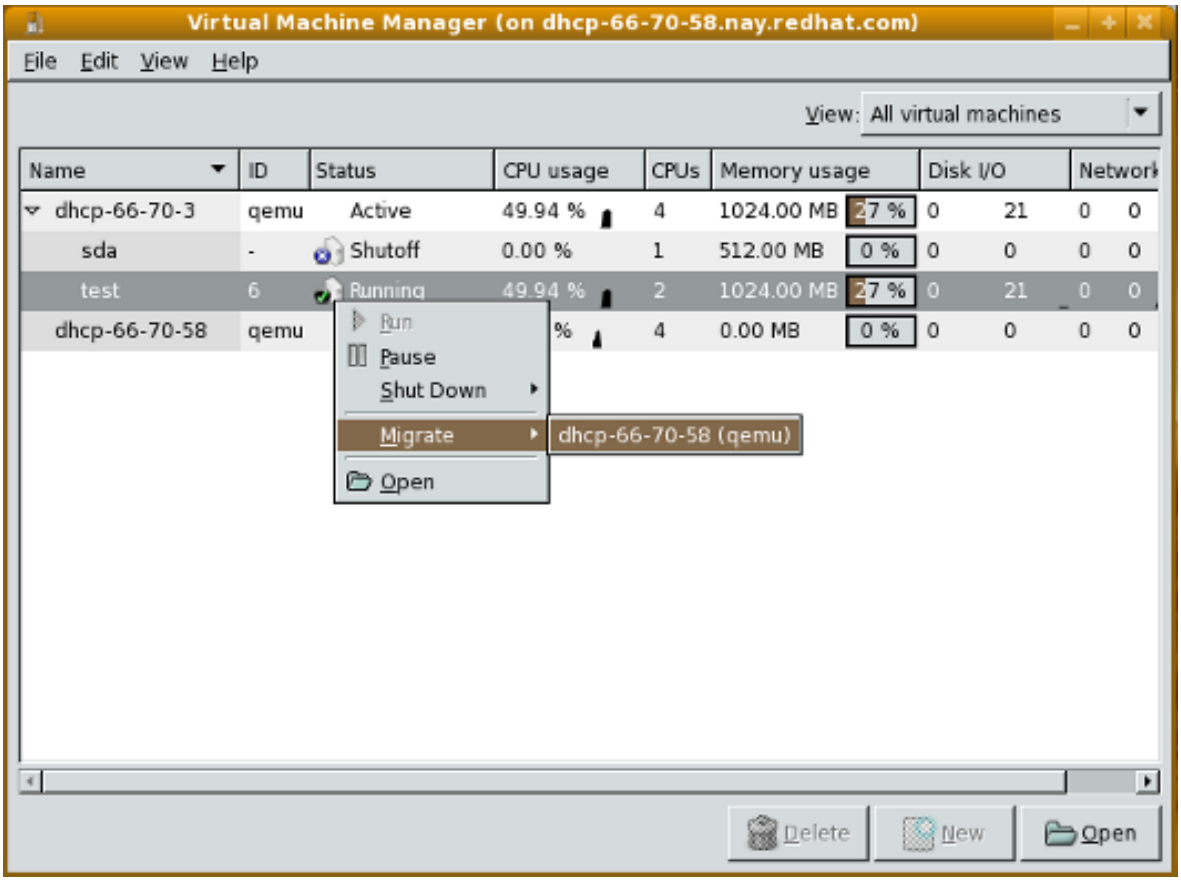
7. Создайте виртуальную машину, использующую этот том, и запустите ее.



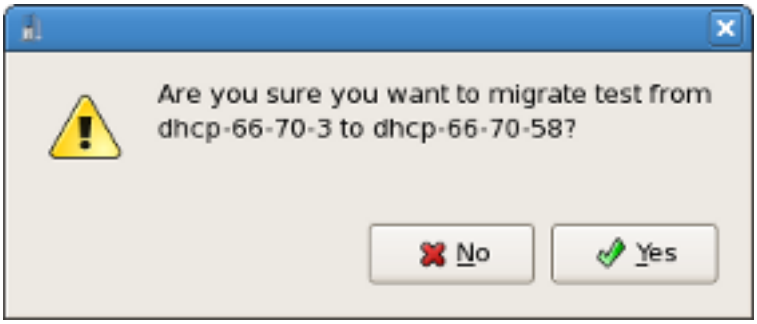
Появится окно виртуальной машины.



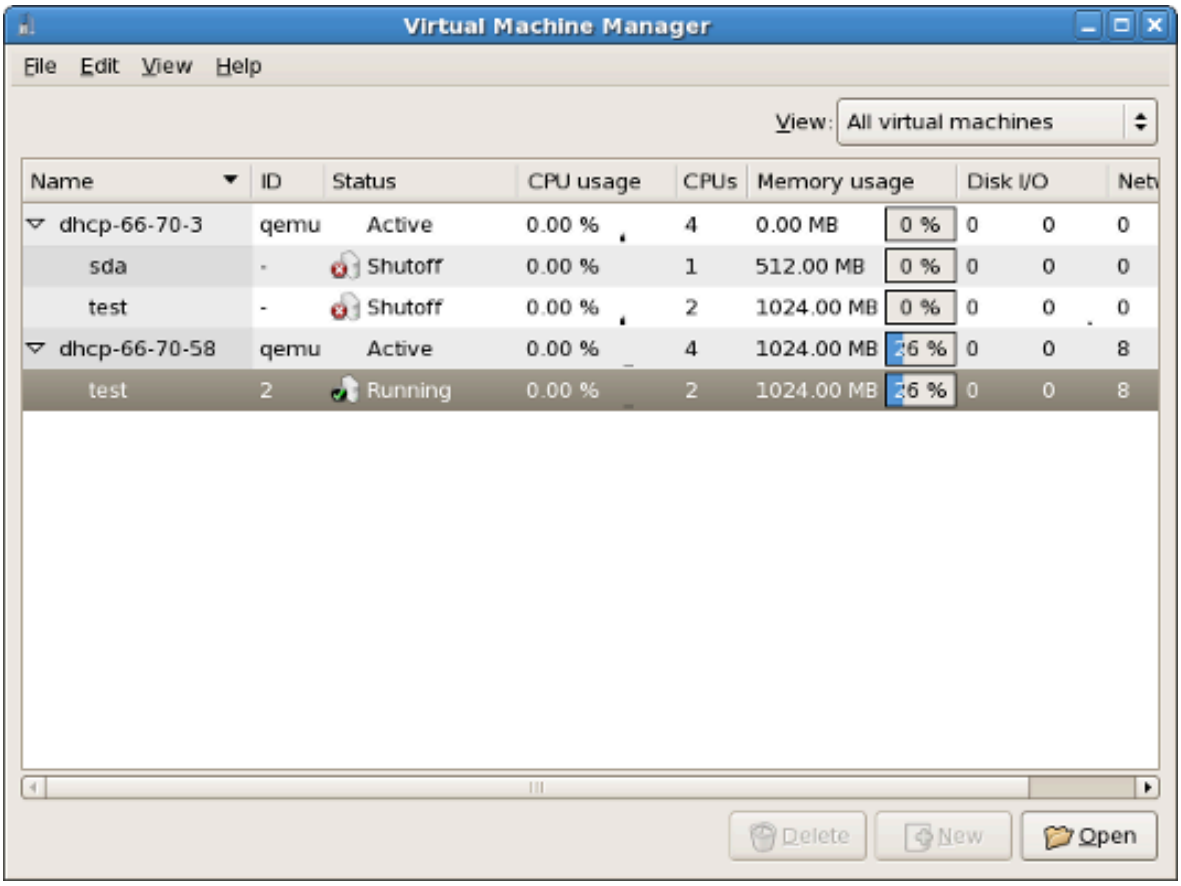
8. В окне менеджера виртуальных машин откройте контекстное меню созданной виртуальной машины и выберите пункт миграции, затем выберите целевой узел.



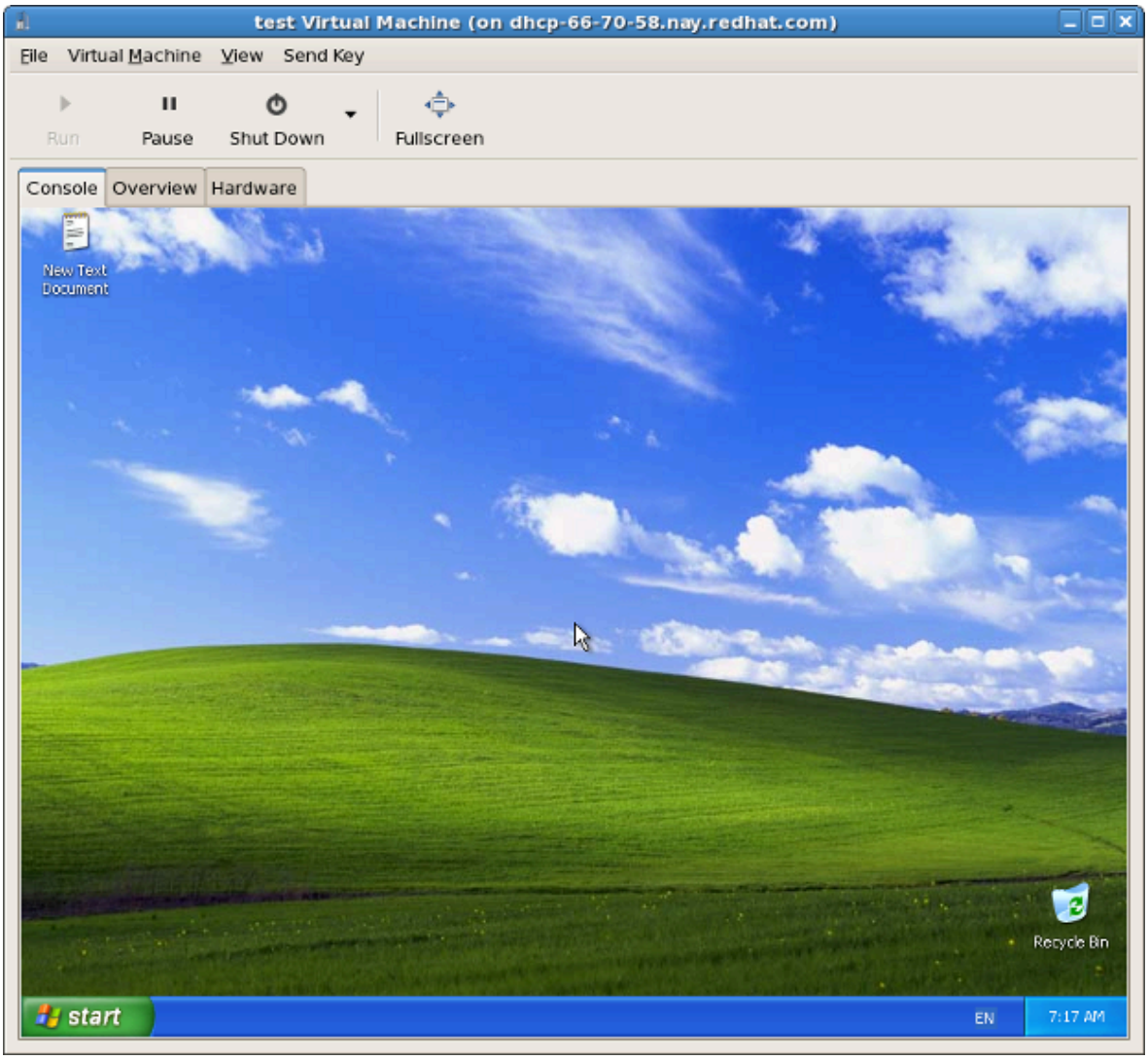
9. Нажмите **Да** для подтверждения.



Менеджер покажет виртуальную машину на целевом узле.



Окно виртуальной машины также отразит ее новое расположение.


test Virtual Machine (on dhcp-66-70-58.nay.redhat.com)
File
Virtual Machine
View
Send Key
Run
Pause
Shut Down
Fullscreen
Console
Overview
Hardware
New Text Document
Recycle Bin
start
EN
7:17 AM

Удаленное управление виртуализированными гостевыми системами

В этой главе рассматривается удаленное управление виртуализированными гостевыми системами с помощью **ssh**, TLS и SSL.

13.1. Удаленное управление с помощью SSH

Пакет *ssh* предоставляет зашифрованный сетевой протокол, который позволяет безопасно передавать управляющие функции удаленным серверам виртуализации. Ниже рассматривается управление удаленными машинами утилитой **libvirt** по SSH-туннелю.Аутентификация осуществляется за счет использования общих ключей **SSH** и паролей или проверочных фраз, получаемых локальным агентом **SSH**. Доступная гостевым виртуальным системам консоль **VNC** также защищена **SSH**.

Обычно используется стандартная конфигурация **SSH**, поэтому нет необходимости в создании дополнительных правил межсетевого экрана для обеспечения доступа к управляющей службе или консоли **VNC**.

При организации удаленного управления виртуальными машинами следует принять во внимание следующие моменты относительно **SSH**:

- потребуются права root для доступа к удаленной машине для управления ее виртуальными машинами;
- изначальный процесс настройки соединения может быть замедлен;
- не существует стандарта или однозначного способа получения ключа пользователя на всех узлах и в гостях;
- при значительном увеличении числа удаленных машин могут возникнуть сложности с использованием SSH.

Настройка доступа SSH для virt-manager

Приведенные далее инструкции подразумевают, что вы начинаете работу с нуля, то есть ключи **SSH** еще не настроены.

1. Потребуется пара общих ключей в системе, где будет выполняться **virt-manager**. Если **ssh** уже настроен, этот шаг можно пропустить.

   ```
   $ ssh-keygen -t rsa
   ```

2. Чтобы разрешить удаленную авторизацию, необходимо скопировать общий ключ на все удаленные машины, где выполняется **libvirt**. Скопируйте файл **$HOME/.ssh/id_rsa.pub** из системы, которая будет использоваться для удаленного управления:

   ```
   $ scp $HOME/.ssh/id_rsa.pub root@somehost:/root/key-dan.pub
   ```

3. После этого подключитесь к удаленной машине по **ssh** и в режиме root добавьте полученный файл в список авторизованных ключей. Если этот список не существует, проверьте разрешения файла.

```
$ ssh root@узел
# mkdir /root/.ssh
# chmod go-rwx /root/.ssh
# cat /root/key-dan.pub >> /root/.ssh/authorized_keys
# chmod go-rw /root/.ssh/authorized_keys
```

Демон libvirt (libvirtd)

libvirtd предоставляет интерфейс для управления виртуальными машинами. Он должен работать на каждом удаленном узле, которым вы планируете управлять. Дополнительно, пакет **kernel-xen** предъявляет отдельные требования к процессору и ядру.

```
$ ssh root@узел
# chkconfig libvirtd on
# service libvirtd start
```

После завершения настройки libvirtd и **SSH** вы сможете обращаться к удаленным виртуальным машинам, управлять ими и обращаться к гостевым системам через **VNC**.

13.2. Удаленное управление с помощью TLS и SSL

Виртуальными машинами можно управлять удаленно по TLS и SSL, что, с одной стороны, обеспечивает масштабируемость, а с другой — усложняет работу по сравнению с обычным использованием ssh (см. *Раздел 13.1, «Удаленное управление с помощью SSH»*). TLS и SSL используются веб-браузерами для установки безопасных соединений. Управляющее соединение **libvirt** откроет порт TCP для входящих запросов (при этом аутентификация будет осуществляться на основе сертификатов x509, а передаваемые данные будут зашифрованы), а для каждой виртуальной машины будет настроена VNC-консоль, использующая аутентификацию x509.

При этом пользователю не нужна учетная запись оболочки в удаленной системе. Но потребуется создать дополнительные правила межсетевого экрана для доступа к службе управления и консоли VNC. Для ограничения доступа пользователей можно использовать специальные списки отзыва сертификатов.

Последовательность действий при настройке доступа TLS/SSL для virt-manager

Здесь подразумевается, что вы начинаете работу с нуля и не обладаете опытом работы с сертификатами TLS/SSL. При наличии сервера управления сертификатами первые шаги можно пропустить.

Настройка сервера с помощью **libvirt**

Информацию о создании сертификатов можно найти на сайте **libvirt** по адресу *http://libvirt.org/remote.html*.

VNC-сервер Xen

На VNC-сервере Xen можно включить TLS, изменив файл конфигурации **/etc/xen/xend-config.sxp**. Снимите комментарий с параметра **(vnc-tls 1)**.

Каталог **/etc/xen/vnc** должен содержать следующие файлы:

- **ca-cert.pem** — сертификат CA;
- **server-cert.pem** — сертификат сервера, подписанный центром сертификации CA;
- **server-key.pem** — частный ключ сервера.

Это обеспечит шифрование канала данных. В некоторых случаях рекомендуется при аутентификации требовать от клиентов предоставить сертификат x509. Для этого снимите комментарий с параметра **(vnc-x509-verify 1)**.

Настройка клиентов **virt-manager** и **virsh**

В настоящее время процесс настройки клиентов может варьироваться. Для активации API **libvirt** через TLS необходимо поместить сертификаты клиента и CA в **/etc/pki**. На сайте *http://libvirt.org/remote.html* можно найти подробную информацию.

В окне интерфейса **virt-manager** выберите **SSL/TLS** в качестве транспортного механизма, используемого при подключении к узлу.

Формат ссылки URI для **virsh**:

- **qemu://hostname.guestname/system** для KVM;
- **xen://hostname.guestname/** для Xen.

Чтобы включить SSL и TLS для VNC, необходимо поместить сертификатор CA и сертификаты клиента в **$HOME/.pki**. Эти файлы включают:

- CA или **ca-cert.pem**: Сертификат CA.
- **libvirt-vnc** или **clientcert.pem**: Сертификат клиента, подписанный центром CA.
- **libvirt-vnc** или **clientkey.pem**: Частный ключ клиента.

13.3. Режимы передачи данных

libvirt поддерживает следующие режимы передачи для удаленного управления:

TLS

Протокол TLS (Transport Layer Security) 1.0 (SSL 3.1) — криптографический протокол, предоставляющий возможности аутентификации и безопасной передачи данных по TCP/IP. Обычно используется для прослушивания общего порта, по умолчанию — 16514. Для этого потребуется создать сертификаты для клиента и сервера.

Сокеты UNIX

Сокеты доменов UNIX доступны только на локальной машине. Они не зашифрованы и для аутентификации используют разрешения UNIX или SELinux. Стандартные имена сокетов — **/var/run/libvirt/libvirt-sock** и **/var/run/libvirt/libvirt-sock-ro** (только для чтения).

SSH

Для передачи данных с использованием SSH (Secure Shell) необходимо установить Netcat (пакет *nc*), в удаленной системе должен выполняться демон **`libvirtd`**, а порт 22 должен быть открыт для SSH-доступа. Инструменты, подобные **`ssh-agent`**, позволят избежать необходимости повторного ввода пароля.

ext

Параметр ext используется внешними программами, которые подключаются к удаленной машине без помощи libvirt, что обычно включает программы сторонних производителей без официальной поддержки безопасности.

tcp

Незашифрованный сокет TCP/IP по умолчанию использует порт 16509 и не рекомендуется для использования в производственной среде, поэтому он обычно отключен. Администраторы могут прибегнуть к его помощи в целях тестирования или при работе в доверенной сети.

Если способ передачи не задан, по умолчанию будет использоваться TLS.

Удаленные URI

`virsh и libvirt используют единообразный идентификатор ресурса URI (Uniform Resource Identifier) для подключения к удаленному узлу. Аргумент --connect команды virsh использует URI для выполнения команд на удаленном узле.`

Формат URI в libvirt (необязательные параметры заключены в квадратные скобки):

```
драйвер[+протокол]://[пользователь@][узел][:порт]/[путь][?
дополнительные_параметры]
```

Отличие локального URI состоит в том, что для него не будет указан способ передачи и имя узла.

Примеры параметров удаленного управления

- Подключение пользователя `ccurran` к удаленному гипервизору Xen на узле `towada` с использованием SSH:

  ```
  xen+ssh://ccurran@towada/
  ```

- Подключение к удаленному гипервизору Xen на узле `towada` с использованием TLS:

  ```
  xen://towada/
  ```

- Подключение к удаленному гипервизору Xen на узле `towada` с использованием TLS. Выражение *`no_verify=1`* отключает проверку сертификата сервера утилитой libvirt.

  ```
  xen://towada/?no_verify=1
  ```

- Подключение к удаленному гипервизору KVM на узле `towada` с использованием SSH:

```
qemu+ssh://towada/system
```

Примеры тестирования

- Подключение к локальному гипервизору KVM с использованием нестандартного сокета UNIX с указанием полного пути:

```
qemu+unix:///system?socket=/opt/libvirt/run/libvirt/libvirt-sock
```

- Подключение к демону libvirt на сервере с IP-адресом 10.1.1.10 (порт 5000) с использованием незашифрованного соединения TCP/IP и стандартных настроек драйвера test.

```
test+tcp://10.1.1.10:5000/default
```

Дополнительные параметры URI

Таблица 13.1, «Дополнительные параметры URI» содержит список дополнительных параметров, которые могут быть добавлены к URI. Обратите внимание, что перед каждым параметром надо добавить знак вопроса («?»); специальные символы будут преобразованы в формат URI. Любые другие параметры будут проигнорированы.

Параметр	Режим передачи	Description	Пример
name	все	Имя можно получить из URI путем удаления протокола, имени узла, пользователя, номера порта и всех дополнительных параметров. В некоторых случаях, однако, имя рекомендуется задать напрямую. Оно будет передано удаленной функции virConnectOpen.	name=qemu:///system
command	ssh, ext	Внешняя команда. Обязательна для ext. Для SSH по умолчанию используется «ssh». Поиск команды будет осуществляться в соответствии со значением PATH.	command=/opt/openssh/bin/ssh
socket	unix, ssh	Путь к сокету домена UNIX.	socket=/opt/libvirt/run/libvirt/libvirt-sock

Параметр	Режим передачи	Description	Пример
		Переопределяет путь, используемый по умолчанию. Для SSH будет передаваться удаленной команде netcat (см. ниже).	
netcat	ssh	Команда netcat на удаленной машине. По умолчанию используется nc. Формат команды при использовании SSH: «command -p порт [-l пользователь] узел netcat -U socket», где «порт», «пользователь», «узел» являются составляющими удаленного URI, а «command», «netcat» и «socket» — дополнительные параметры.	netcat=/opt/netcat/bin/nc
no_verify	tls	Ненулевое значение отключает проверку сертификата сервера клиентом. Но чтобы отключить проверку сертификата клиента или IP-адреса сервером потребуется изменить конфигурацию libvirtd.	no_verify=1
no_tty	ssh	Ненулевое значение отключает запрос пароля, если SSH не может пройти авторизацию автоматически на удаленной машине (для ssh-agent и пр.). Используется при отсутствии доступа к терминалу, например, в графических программах использующих libvirt.	no_tty=1

Таблица 13.1. Дополнительные параметры URI

Часть IV. Подробнее о виртуализации

Команды виртуализации, системные утилиты, приложения и дополнительные системы

Последующие главы содержат подробное описание команд виртуализации, системных утилит и приложений в составе Fedora для опытных пользователей, интересующихся расширенными возможностями.

Утилиты виртуализации

Далее приведен список инструментов для управления виртуализацией, отладки, а также сетевых утилит, которые используются в системах с Xen.

Утилиты системного администрирования

- **vmstat**
- **iostat**
- **lsof**

```
# lsof -i :5900
xen-vncfb 10635  root  5u  IPv4 218738  TCP
 grumble.boston.redhat.com:5900 (LISTEN)
```

- **qemu-img**

Расширенные утилиты отладки

- **systemTap**
- **crash**
- **xen-gdbserver**
- **sysrq**
- **sysrq t**
- **sysrq w**
- **sysrq c**

Сетевое окружение

brtcl

-
```
# brctl show
bridge name  bridge id            STP enabled    interfaces
xenbr0       8000.feffffffffff    no             vif13.0
                                                 pdummy0
                                                 vif0.0
```

-
```
# brctl showmacs xenbr0
port no  mac addr                is local?       aging timer
  1      fe:ff:ff:ff:ff:ff       yes             0.00
```

-
```
# brctl showstp xenbr0
xenbr0
bridge id              8000.feffffffffff
designated root        8000.feffffffffff
```

```
root port              0                    path cost
 0
max age                20.00                bridge max age
 20.00
hello time             2.00                 bridge hello time
 2.00
forward delay          0.00                 bridge forward delay
 0.00
aging time            300.01
hello timer            1.43                 tcn timer
 0.00
topology change timer  0.00                 gc timer
 0.02
flags

vif13.0 (3)
port id                8003                 state
 forwarding
designated root        8000.fefffffffffff   path cost
 100
designated bridge      8000.fefffffffffff   message age timer
 0.00
designated port        8003                 forward delay timer
 0.00
designated cost        0                    hold timer
 0.43
flags

pdummy0 (2)
port id                8002                 state
 forwarding
designated root        8000.fefffffffffff   path cost
 100
designated bridge      8000.fefffffffffff   message age timer
 0.00
designated port        8002                 forward delay timer
 0.00
designated cost        0                    hold timer
 0.43
flags

vif0.0 (1)
port id                8001                 state
 forwarding
designated root        8000.fefffffffffff   path cost
 100
designated bridge      8000.fefffffffffff   message age timer
 0.00
designated port        8001                 forward delay timer
 0.00
```

```
designated cost        0                    hold timer
 0.43
flags
```

- **ifconfig**
- **tcpdump**

Утилиты KVM

- **ps**
- **pstree**
- **top**
- **kvmtrace**
- **kvm_stat**

Утилиты Xen

- **xentop**
- **xm dmesg**
- **xm log**

Управление виртуальными машинами с помощью virsh

Текстовая утилита **virsh** предназначена для управления гостевыми системами и гипервизором.

virsh использует libvirt API и служит альтернативой **xm** и графическому менеджеру виртуальных машин (**virt-manager**). Непривилегированные пользователи могут выполнять доступ в только в режиме чтения. С помощью **virsh** можно исполнять сценарии для виртуальных машин.

Обзор команд virsh

Приведенные ниже таблицы содержат перечень основных параметров командной строки virsh.

Команда	Description
help	Краткая справка.
list	Просмотр всех виртуальных машин.
dumpxml	Вывести файл конфигурации XML для заданной виртуальной машины.
create	Создать виртуальную машину из файла конфигурации XML и ее запуск.
start	Запустить неактивную виртуальную машину.
destroy	Принудительно остановить работу виртуальной машины.
define	Определяет файл конфигурации XML для заданной виртуальной машины.
domid	Просмотр идентификатора виртуальной машины.
domuuid	Просмотр UUID виртуальной машины.
dominfo	Просмотр сведений о виртуальной машине.
domname	Просмотр имени виртуальной машины.
domstate	Просмотр состояния виртуальной машины.
quit	Закрыть интерактивный терминал.
reboot	Перезагрузить виртуальную машину.
restore	Восстановить сохраненную в файле виртуальную машину.
resume	Возобновить работу приостановленной виртуальной машины.
save	Сохранить состояние виртуальной машины в файл.
shutdown	Корректно завершить работу виртуальной машины.
suspend	Приостановить работу виртуальной машины.

Команда	Description
undefine	Удалить все файлы виртуальной машины.
migrate	Перенести виртуальную машину на другой узел.

Таблица 15.1. Команды управления виртуальными машинами

Для управления ресурсами виртуальной машины и гипервизора используются следующие команды **virsh**:

Команда	Description
setmem	Определяет размер выделенной виртуальной машине памяти.
setmaxmem	Ограничивает максимально доступный гипервизору объем памяти.
setvcpus	Изменяет число предоставленных гостю виртуальных процессоров.
vcpuinfo	Просмотр информации о виртуальных процессорах.
vcpupin	Настройка соответствий виртуальных процессоров.
domblkstat	Просмотр статистики блочных устройств для работающей виртуальной машины.
domifstat	Просмотр статистики сетевых интерфейсов для работающей виртуальной машины.
attach-device	Подключить определенное в XML-файле устройство к гостю.
attach-disk	Подключить новое дисковое устройство к гостю
attach-interface	Подключить новый сетевой интерфейс к гостю
detach-device	Отключить устройство от гостя (принимает те же определения XML, что и **attach-device**).
detach-disk	Отключить дисковое устройство от гостя.
detach-interface	Отключить сетевой интерфейс от гостя.

Таблица 15.2. Параметры управления ресурсами

Другие команды **virsh**:

Команда	Description
version	Просмотр версии **virsh**.
nodeinfo	Просмотр информации о гипервизоре.

Таблица 15.3. Другие команды

Подключение к гипервизору

Подключение к сессии гипервизора с помощью **virsh** :

```
# virsh connect {узел ИЛИ URL}
```

где **<узел>** — имя машины гипервизора. Чтобы начать сессию в режиме чтения, добавьте параметр **-readonly**.

Создание XML-файла конфигурации виртуальной машины

Выведите файл конфигурации виртуальной машины:

```
# virsh dumpxml {ID_домена, имя_домена, UUID_домена}
```

Эта команда выведет информацию о домене в **stdout**. Сохраните вывод в файл:

```
virsh dumpxml ID_гостя > guest.xml
```

Теперь на основе файла **guest.xml** можно создать новую гостевую систему (см. *Редактирование файла конфигурации виртуальной машины*). Отредактируйте файл конфигурации, добавив дополнительные гостевые системы или изменив устройства. *Раздел 18.1, «Использование файлов конфигурации с помощью virsh»* содержит дополнительную информацию об изменении файлов, созданных с помощью **virsh dumpxml**.

Пример вывода **virsh dumpxml**:

```
# virsh dumpxml r5b2-mySQL01
<domain type='xen' id='13'>
    <name>r5b2-mySQL01</name>
    <uuid>4a4c59a7ee3fc78196e4288f2862f011</uuid>
    <bootloader>/usr/bin/pygrub</bootloader>
    <os>
        <type>linux</type>
        <kernel>/var/lib/libvirt/vmlinuz.2dgnU_</kernel>
        <initrd>/var/lib/libvirt/initrd.UQafMw</initrd>
        <cmdline>ro root=/dev/VolGroup00/LogVol00 rhgb quiet</cmdline>
    </os>
    <memory>512000</memory>
    <vcpu>1</vcpu>
    <on_poweroff>destroy</on_poweroff>
    <on_reboot>restart</on_reboot>
    <on_crash>restart</on_crash>
    <devices>
        <interface type='bridge'>
            <source bridge='xenbr0'/>
            <mac address='00:16:3e:49:1d:11'/>
            <script path='vif-bridge'/>
        </interface>
        <graphics type='vnc' port='5900'/>
        <console tty='/dev/pts/4'/>
    </devices>
</domain>
```

Создание виртуальной машины на основе файла конфигурации

Виртуальную машину можно создать на основе файла конфигурации XML. Так, можно скопировать уже существующий файл из других систем или использовать опцию **dumpxml** (см. *Создание XML-файла конфигурации виртуальной машины*). Команда создания виртуальной машины из XML-файла с помощью **virsh** выглядит так:

```
# virsh create configuration_file.xml
```

Редактирование файла конфигурации виртуальной машины

Настройки виртуальной машины можно изменить не только, когда она отключена, но и во время работы. Это можно сделать с помощью команды **virsh edit**. Пример изменения настроек виртуальной машины с именем *testing*:

```
# virsh edit testing
```

Откроется окно текстового редактора, заданного переменной оболочки **$EDITOR** (по умолчанию используется **vi**).

Приостановка виртуальной машины

Команда приостановки виртуальной машины с помощью **virsh**:

```
# virsh suspend {ID_домена, имя_домена, UUID_домена}
```

В этом состоянии виртуальная машина все еще продолжает использовать системную память, но освобождает ресурсы процессора. Операции ввода и вывода приостанавливаются. Работа виртуальной машины может быть возобновлена с помощью команды **resume** (см. *Возобновление работы виртуальной машины*).

Возобновление работы виртуальной машины

Возобновить работу приостановленной виртуальной машины можно с помощью параметра **resume** команды **virsh**:

```
# virsh resume {ID_домена, имя_домена, UUID_домена}
```

Работа машины будет возобновлена немедленно. Параметры будут сохраняться между циклами **suspend** и **resume**.

Сохранение виртуальной машины

Команда сохранения текущего состояния виртуальной машины:

```
# virsh save {ID_домена, имя_домена, UUID_домена} файл
```

В результате выполнения этой команды виртуальная машина будет остановлена, а ее данные будут сохранены в заданный файл (что может занять некоторое время в зависимости от доступного гостю объема памяти). Состояние может быть позднее восстановлено с помощью команды **restore** (см. *Восстановление виртуальной машины*).

Восстановление виртуальной машины

Чтобы восстановить виртуальную машину, заранее сохраненную с помощью команды **virsh save** (см. *Сохранение виртуальной машины*), выполните:

```
# virsh restore файл
```

Сохраненная машина будет восстановлена из файла и перезапущена, что может занять некоторое время. Имя и идентификатор UUID виртуальной машины останутся неизменными, но будет предоставлен новый идентификатор домена.

Завершение работы виртуальной машины

Команда завершения работы:

```
# virsh shutdown {ID_домена, имя_домена, UUID_домена}
```

Поведение выключаемого гостя можно контролировать с помощью параметра **on_shutdown** в его файле конфигурации.

Перезагрузка виртуальной машины

Команда перезагрузки:

```
#virsh reboot {ID_домена, имя_домена, UUID_домена}
```

Поведение перезагружаемого гостя можно контролировать с помощью параметра **on_reboot** в его файле конфигурации.

Принудительная остановка виртуальной машины

Команда принудительной остановки:

```
# virsh destroy {ID_домена, имя_домена, UUID_домена}
```

Эта команда выполнит немедленное отключение и остановит все сессии гостевых доменов. Стоит помнить, что такое внезапное завершение может привести к повреждению файловых систем виртуальной машины. Команду **destroy** рекомендуется использовать только в случае, если виртуальная машина не отвечает на запросы. При работе с паравиртуализированными гостями используйте опцию **shutdown** (*Завершение работы виртуальной машины*).

Определение идентификатора домена

Команда определения идентификатора домена виртуальной машины:

```
# virsh domid {имя_домена, UUID_домена}
```

Определение имени домена

Команда определения имени домена виртуальной машины:

```
# virsh domname {ID_домена, UUID_домена}
```

Определение UUID

Команда определения универсального идентификатора UUID виртуальной машины:

```
# virsh domuuid {ID_домена, имя_домена}
```

Пример вывода **virsh domuuid**:

```
# virsh domuuid r5b2-mySQL01
4a4c59a7-ee3f-c781-96e4-288f2862f011
```

Получение информации о виртуальной машине

Команда для получения информации:

```
# virsh dominfo {ID_домена, имя_домена, UUID_домена}
```

Пример вывода **virsh dominfo**:

```
# virsh dominfo r5b2-mySQL01
id:             13
name:           r5b2-mysql01
uuid:           4a4c59a7-ee3f-c781-96e4-288f2862f011
os type:              linux
state:          blocked
cpu(s):         1
cpu time:             11.0s
max memory:     512000 kb
used memory:    512000 kb
```

Получение информации об узле

Команда получения информации об узле:

```
# virsh nodeinfo
```

Пример вывода **virsh nodeinfo**:

```
# virsh nodeinfo
CPU model                    x86_64
CPU (s)                      8
CPU frequency                2895 Mhz
CPU socket(s)                2
Core(s) per socket           2
Threads per core:            2
Numa cell(s)                 1
```

```
Memory size:                  1046528 kb
```

Вывод содержит информацию об узле и машинах, поддерживающих виртуализацию.

Просмотр списка виртуальных машин

Команда для просмотра списка виртуальных машин и их состояния:

```
# virsh list
```

Можно добавить аргументы:

`--inactive` покажет список неактивных доменов (неактивным считается тот домен, который был определен, но в настоящий момент не является активным).

`--all` покажет все виртуальные машины независимо от их состояния. Пример:

```
# virsh list --all
 Id Name                 State
----------------------------------
  0 Domain-0             running
  1 Domain202            paused
  2 Domain010            inactive
  3 Domain9600           crashed
```

Столбец «Status» может содержать следующие значения:

- `running` — работающие виртуальные машины, то есть те машины, которые используют ресурсы процессора в момент выполнения команды.
- `blocked` — заблокированные, неработающие машины. Такой статус может быть вызван ожиданием ввода/вывода или пребыванием машины в спящем режиме.
- `paused` — приостановленные домены. В это состояние они переходят, если администратор нажал кнопку паузы в окне менеджера виртуальных машин или выполнил команду **`xm pause`** или **`virsh suspend`**. В приостановленном состоянии гость продолжает потреблять ресурсы, но не может занимать больше процессорных ресурсов.
- `shutdown` — виртуальные машины, завершающие свою работу. При получении виртуальной машиной сигнала завершения работы, она начнет завершать все процессы. Стоит отметить, что некоторые операционные системы не отвечают на такие сигналы.
- `dying` — сбойные домены и домены, которые не смогли корректно завершить свою работу.
- `crashed` — сбойные домены, работа которых была прервана. В этом состоянии домены находятся, если не была настроена их перезагрузка в случае сбоя.

Получение информации о виртуальных процессорах

Команда получения информации о виртуальных процессорах:

```
# virsh vcpuinfo {ID_домена, имя_домена, UUID_домена}
```

Пример вывода:

```
# virsh vcpuinfo r5b2-mySQL01
VCPU:           0
CPU:            0
State:          blocked
CPU time:       0.0s
CPU Affinity:   yy
```

Настройка соответствий виртуальных процессоров

Команда сопоставления виртуальных процессоров физическим:

```
# virsh vcpupin {ID_домена, имя_домена, UUID_домена} vcpu, список_cpu
```

Здесь **vcpu** — номер виртуального процессора, а **список_cpu** — сопоставляемые ему физические процессоры.

Изменение числа виртуальных процессоров

Команда изменения числа процессоров для домена:

```
# virsh setvcpus {ID_домена, имя_домена, UUID_домена} число
```

Обратите внимание, что заданное число не может превышать значение, определенное при создании гостя.

Изменение выделенного объема памяти

Команда изменения выделенного виртуальной машине объема памяти:

```
# virsh setmem {ID_домена, имя_домена} число
```

Объем памяти, определяемый заданным числом, должен быть указан в килобайтах. Обратите внимание, что объем не может превышать значение, определенное при создании виртуальной машины, но в то же время не должен быть меньше 64 мегабайт. Изменение максимального объема памяти может оказать влияние на функциональность гостя только в том случае, если указанный размер меньше исходного. В таком случае использование памяти будет ограничено.

Получение информации о блочных устройствах

Команда для получения информации о блочных устройствах работающей виртуальной машины:

```
# virsh domblkstat виртуальная_машина блочное_устройство
```

Получение информации о сетевых устройствах

Команда для получения информации о сетевых интерфейсах работающей виртуальной машины:

```
# virsh domifstat виртуальная_машина интерфейс
```

Миграция виртуальных машин

virsh позволяет переносить виртуальные машины с одного узла на другой. Для выполнения живой миграции просто нужно указать параметр **--live**. Команда переноса выглядит так:

```
# virsh migrate --live GuestName DestinationURL
```

Параметр *--live* не является обязательным.

The *GuestName* parameter represents the name of the guest which you want to migrate.

The *DestinationURL* parameter is the URL or hostname of the destination system. The destination system must run the same version of Fedora, be using the same hypervisor and have **libvirt** running.

Once the command is entered you will be prompted for the root password of the destination system.

Управление виртуальными сетями

В этой секции будет рассмотрены управляющие команды **virsh**. Например, команда просмотра списка виртуальных сетей выглядит так:

```
# virsh net-list
```

Пример вывода этой команды:

```
# virsh net-list
Name                 State      Autostart
-----------------------------------------
default              active     yes
vnet1                     active     yes
vnet2                     active     yes
```

Просмотр информации для заданной виртуальной сети:

```
# virsh net-dumpxml имя_сети
```

Пример вывода этой команды (в формате XML):

```
# virsh net-dumpxml vnet1
<network>
  <name>vnet1</name>
  <uuid>98361b46-1581-acb7-1643-85a412626e70</uuid>
  <forward dev='eth0'/>
  <bridge name='vnet0' stp='on' forwardDelay='0' />
  <ip address='192.168.100.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.100.128' end='192.168.100.254' />
```

```
    </dhcp>
  </ip>
</network>
```

Другие команды управления виртуальными сетями:

- **virsh net-autostart *имя_сети*** — автоматический запуск заданной сети.
- **virsh net-create *файл_XML*** — создание и запуск новой сети на основе существующего XML-файла.
- **virsh net-define *файл_XML*** — создание нового сетевого устройства на основе существующего XML-файла. Устройство не будет запущено.
- **virsh net-destroy *имя_сети*** — удаление заданной сети.
- **virsh net-name *UUID_сети*** — преобразование заданного идентификатора в имя сети.
- **virsh net-uuid *имя_сети*** — преобразование заданного имени в идентификатор UUID.
- **virsh net-start *имя_неактивной_сети*** — запуск неактивной сети.
- **virsh net-undefine *имя_неактивной_сети*** — удаление определения неактивной сети.

Управление виртуальными машинами с помощью менеджера виртуальных машин (virt-manager)

Данная глава содержит описание элементов интерфейса менеджера виртуальных машин: окон, диалогов, управляющих компонентов.

`virt-manager` предоставляет графический интерфейс для доступа к гипервизорам и виртуальным машинам в локальной и удаленных системах. С помощью **`virt-manager`** можно создать и полностью виртуализированные, и паравиртуализированные виртуальные машины. Кроме того, **`virt-manager`** выполняет управляющие функции:

- выделение памяти;
- выделение виртуальных процессоров;
- мониторинг производительности;
- сохранение и восстановление, приостановка и возобновление работы, запуск и завершение работы виртуальных машин;
- доступ к текстовой и графической консоли;
- автономная и живая миграция.

16.1. Окно соединений

Это окно появится первым и предложит выбрать сессию гипервизора. Непривилегированные пользователи смогут запустить сессию в режиме чтения, а пользователь root может получить полный доступ. Обычно в качестве гипервизора можно выбрать локальный узел Xen или QEMU (при наличии KVM).

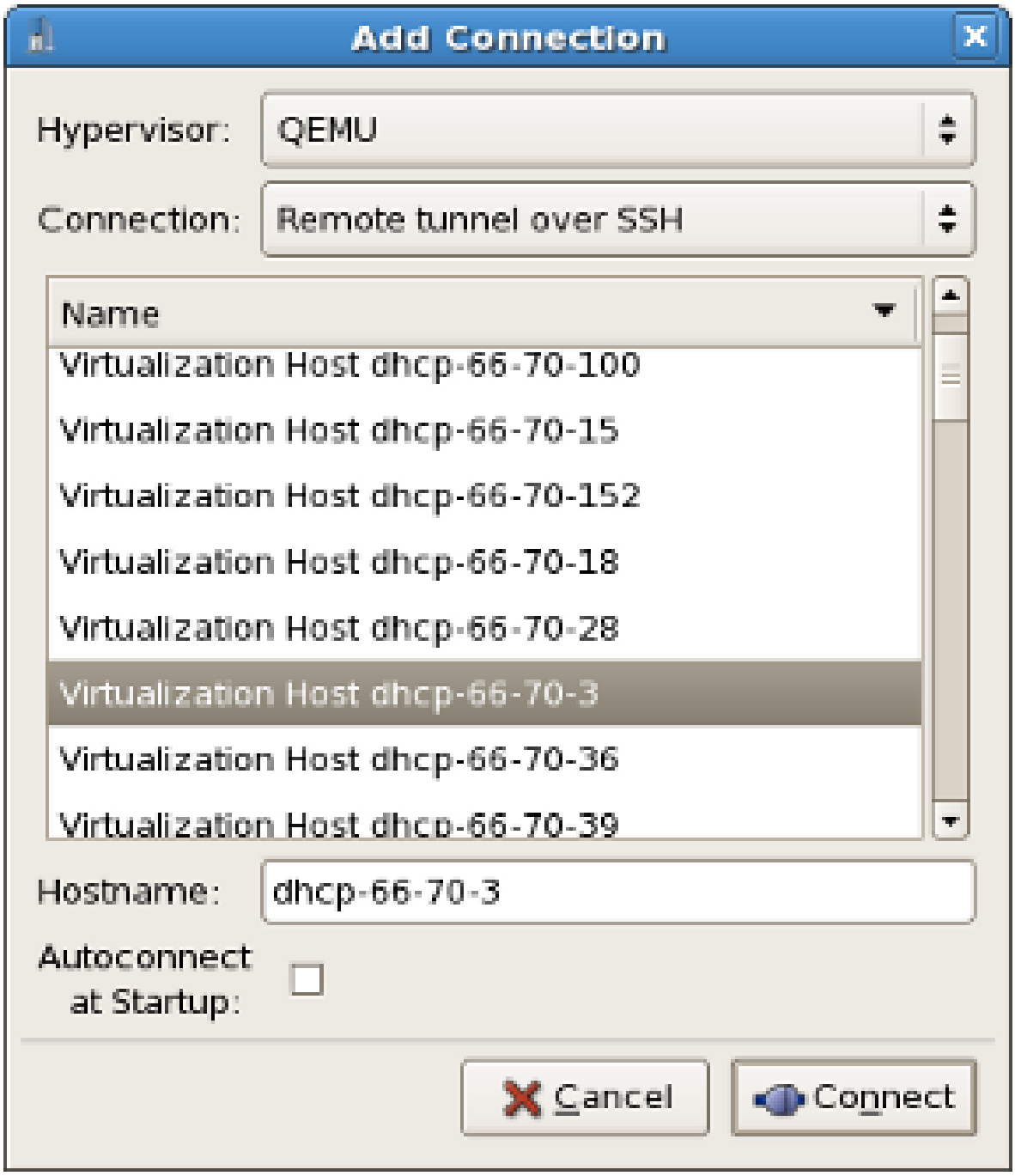


Рисунок 16.1. Окно соединений менеджера виртуальных машин

16.2. Главное окно менеджера виртуальных машин

В главном окне менеджера показаны все выполняющиеся виртуальные машины и выделенные им ресурсы (домен 0 включительно). Показанные поля можно отфильтровать. Двойной щелчок на имени виртуальной машины откроет ее консоль. Выбор виртуальной машины и двойной щелчок на кнопке **Подробности** (Details) откроет окно сведений об этой машине. Новая машина может быть создана в меню **Файл** (File).

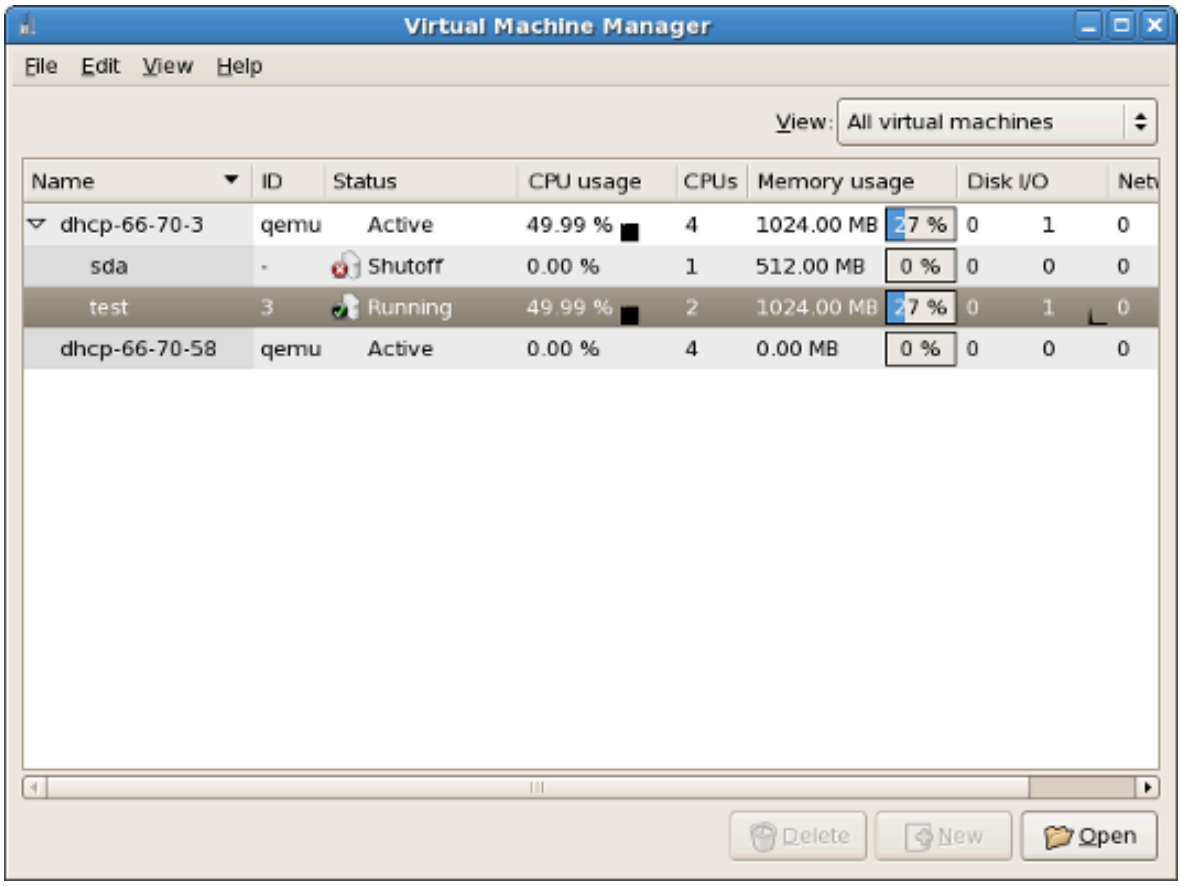


Рисунок 16.2. Главное окно менеджера виртуальных машин

16.3. Окно сведений менеджера виртуальных машин

В этом окне показаны диаграммы и статистика утилизации ресурсов в реальном времени. Поле UUID содержит значение уникального идентификатора виртуальной машины.

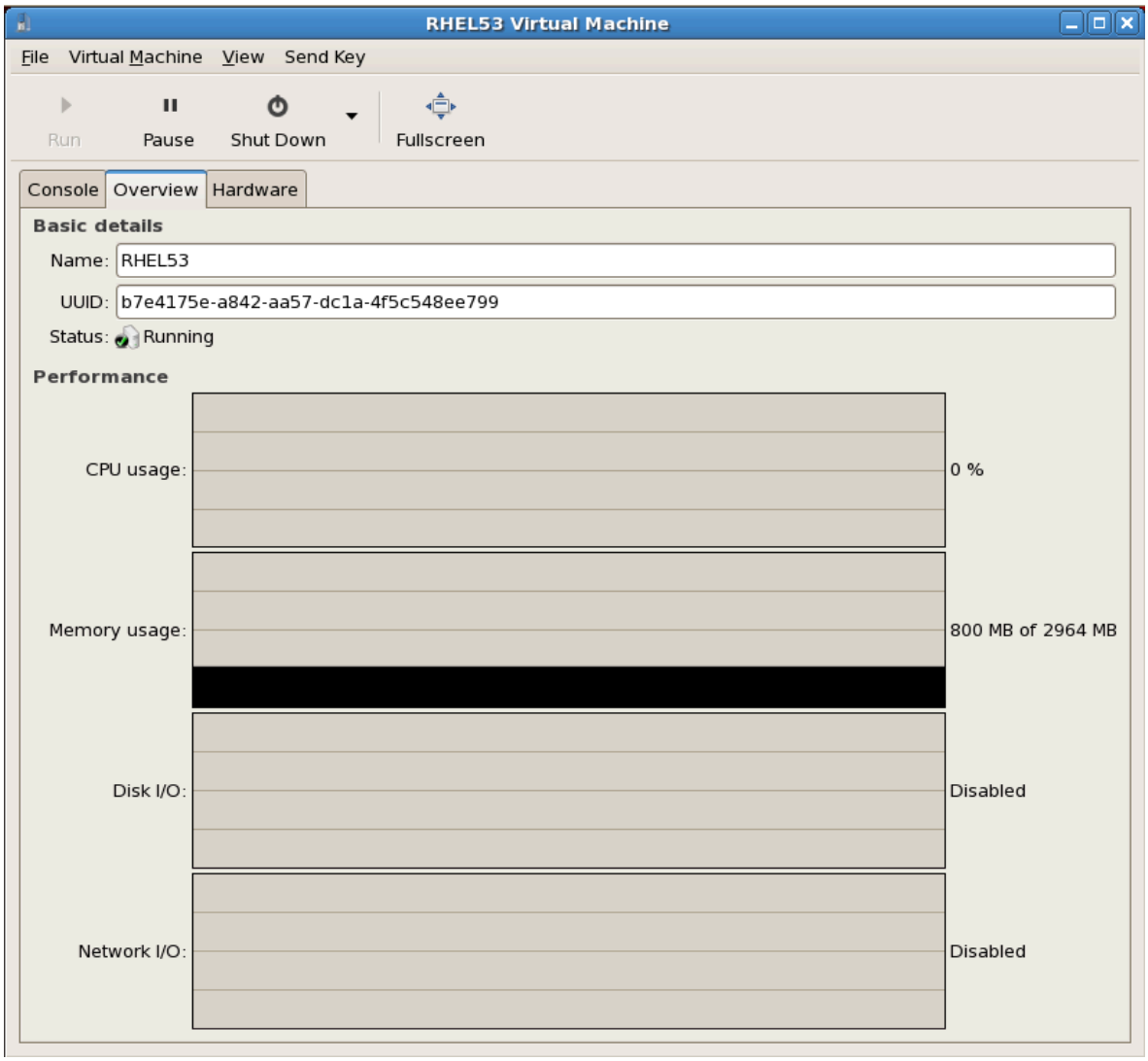


Рисунок 16.3. Окно сведений менеджера виртуальных машин.

16.4. Графическая консоль виртуальной машины

Это окно содержит графическую консоль виртуальной машины. Паравиртуализированные и полностью виртуализированные машины будут использовать различные методы экспортирования локального виртуального буфера кадров, но в то же время оба способа используют **VNC** для обеспечения доступа с консоли. Если настройки виртуальной машины требуют аутентификации, сначала будет запрошен ввод пароля.

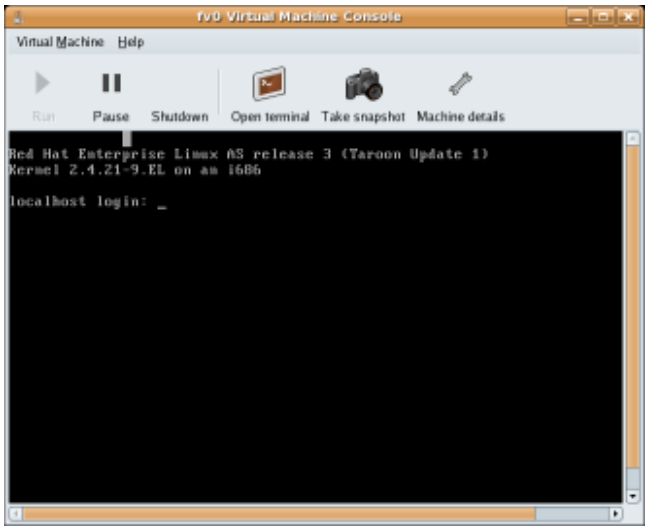


Рисунок 16.4. Окно графической консоли

Замечание относительно безопасности и VNC

Многие эксперты сомневаются в защите VNC, хотя некоторые изменения были внесены с целью усиления защиты VNC при использовании в окружении виртуализации Fedora. Так, гостевые машины прослушивают только петлевой адрес (`127.0.0.1`) локального узла dom0, что гарантирует, что только пользователи с правами доступа к оболочке могут обращаться к virt-manager и виртуальной машине через VNC.

Глава 13, Удаленное управление виртуализированными гостевыми системами содержит инструкции по удаленному администрированию. Для обеспечения необходимого уровня защиты можно использовать TLS.

Окружение локального рабочего стола способно перехватывать комбинации клавиш (например, Ctrl+Alt+F11) для предотвращения их отправки гостевой машине. Чтобы отправить такие последовательности, используйте свойство «западания» клавиш **`virt-manager`**. Нажмите клавишу модификатора (Ctrl или Alt) 3 раза для ее перехода в нажатое состояние. Клавиша будет считаться нажатой до тех пор, пока не будет нажата любая клавиша, отличная от модификатора. Таким образом, чтобы передать гостевой системе комбинацию Ctrl-Alt-F11, необходимо последовательно нажать Ctrl Ctrl Ctrl Alt+F11.

16.5. Starting virt-manager

Чтобы начать сессию менеджера виртуальных машин, в меню приложений выберите **`Система`**, затем **`Virtual Machine Manager`** (**`virt-manager`**).

Появится главное окно менеджера виртуальных машин.

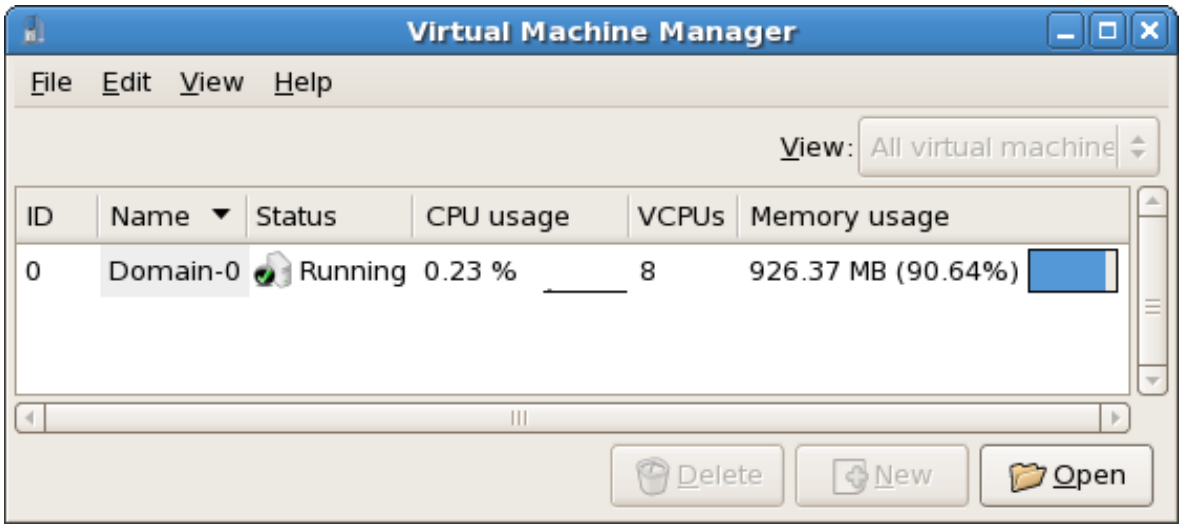


Рисунок 16.5. Запуск **virt-manager**

Или же **virt-manager** можно запустить удаленно с использованием SSH. Пример:

```
ssh -X адрес_узла[remotehost]# virt-manager
```

Раздел 13.1, «Удаленное управление с помощью SSH» содержит информацию об управлении виртуальными машинами и узлами с помощью **ssh**.

16.6. Восстановление сохраненной машины

Все установленные в вашей системе виртуальные машины показаны в главном окне менеджера. Исходная система обозначена как Domain0. Если список пуст, это значит, что в настоящий момент нет работающих машин.

Последовательность действий при восстановлении ранее сохраненной сессии:

1. В меню **Файл** (File) выберите **Восстановить виртуальную машину** (Restore saved machine).

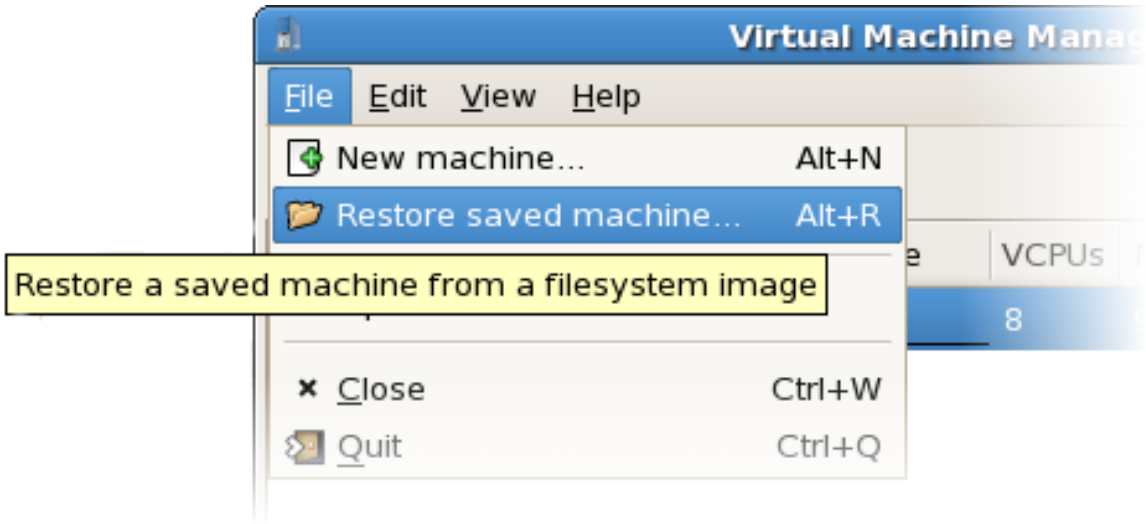


Рисунок 16.6. Восстановление виртуальной машины

2. Появится окно восстановления.

3. Перейдите к каталогу, содержащему файл сессии, и выберите файл.

4. Нажмите **Открыть** (Open).

Виртуальная система появится в главном окне менеджера.

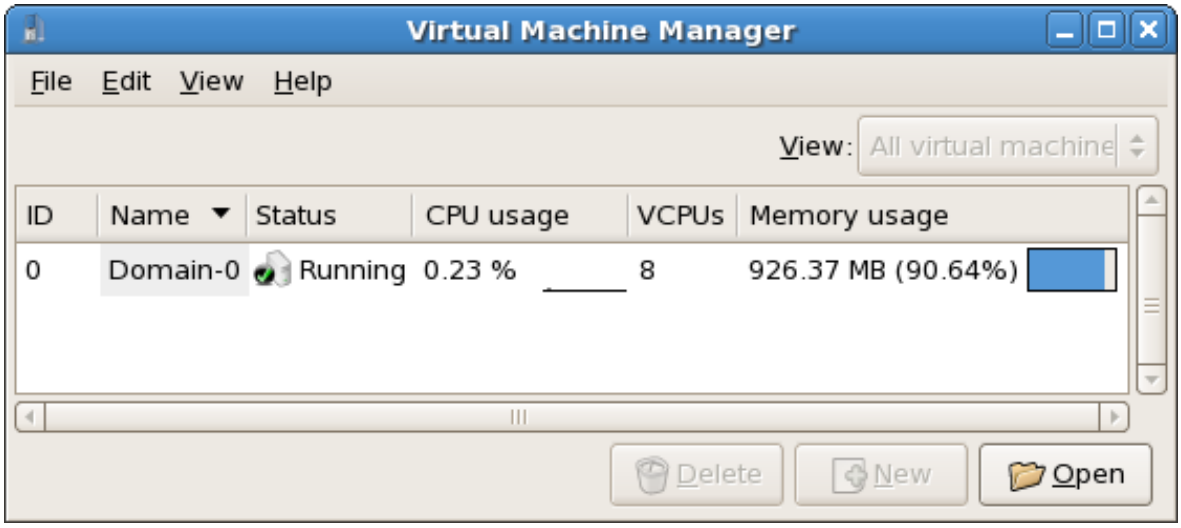


Рисунок 16.7. Восстановленная сессия виртуальной машины

16.7. Просмотр информации о гостевой системе

С помощью менеджера виртуальных машин можно получить доступ к подробной информации о всех виртуальных машинах.

Порядок действий:

1. В главном окне выберите виртуальную машину.

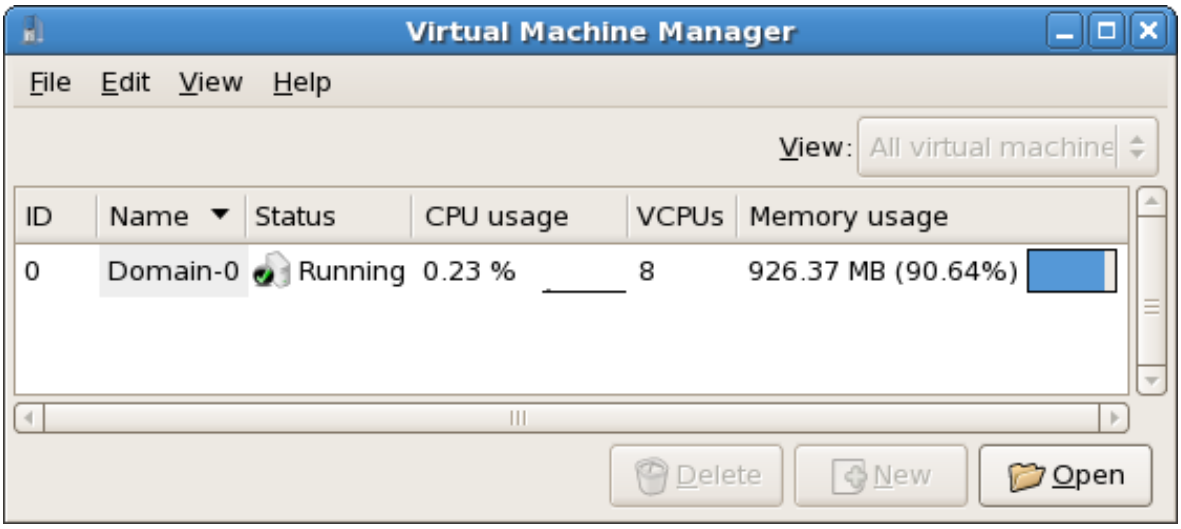


Рисунок 16.8. Выбор виртуальной машины

2. В меню **Правка** (Edit) выберите **Подробнее о виртуальной машине** (Virtual Machine Details) или нажмите кнопку **Подробности** (Details) в нижней части главного окна менеджера.

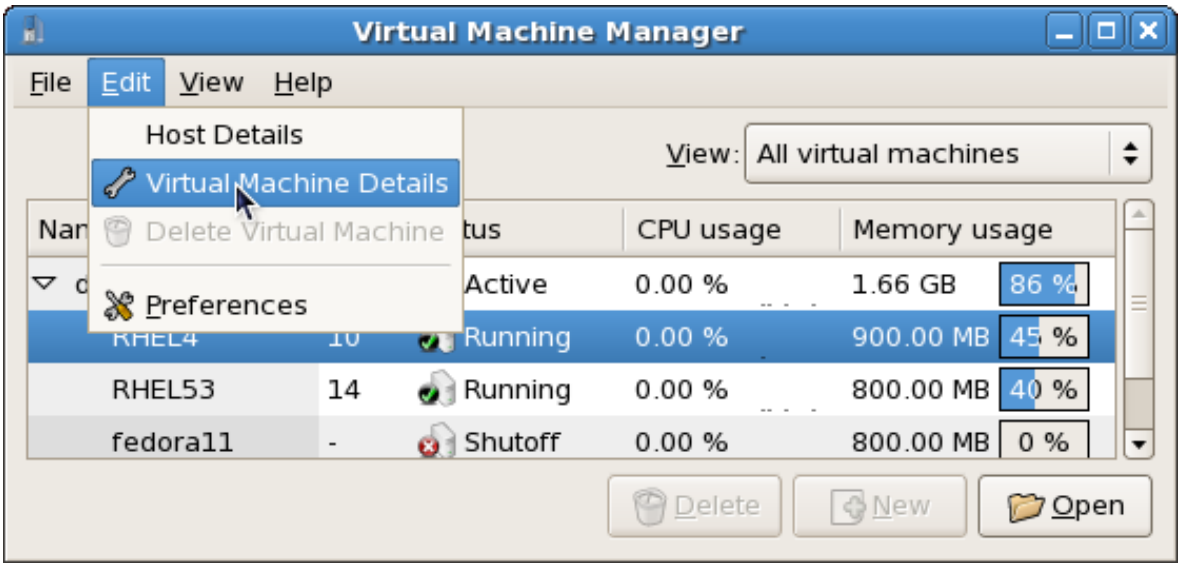


Рисунок 16.9. Меню для получения сведений виртуальной машины

Появится окно просмотра сведений виртуальной машины, где будут доступна информация об использовании ресурсов процессора и памяти.

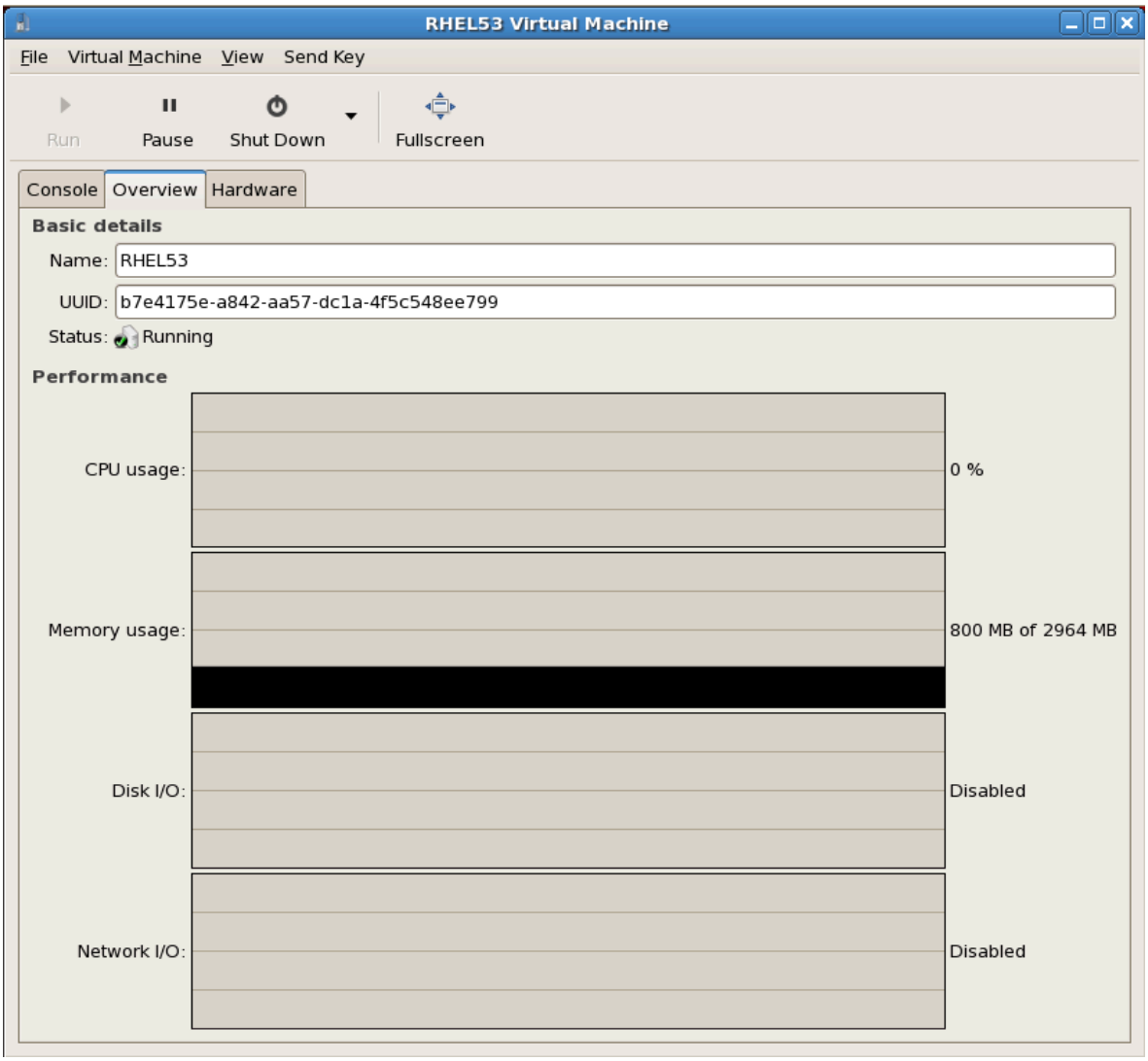


Рисунок 16.10. Обзор информации о гостевой системе

3. Выберите вкладку **Оборудование** (Hardware).

   Появится окно сведений об оборудовании.

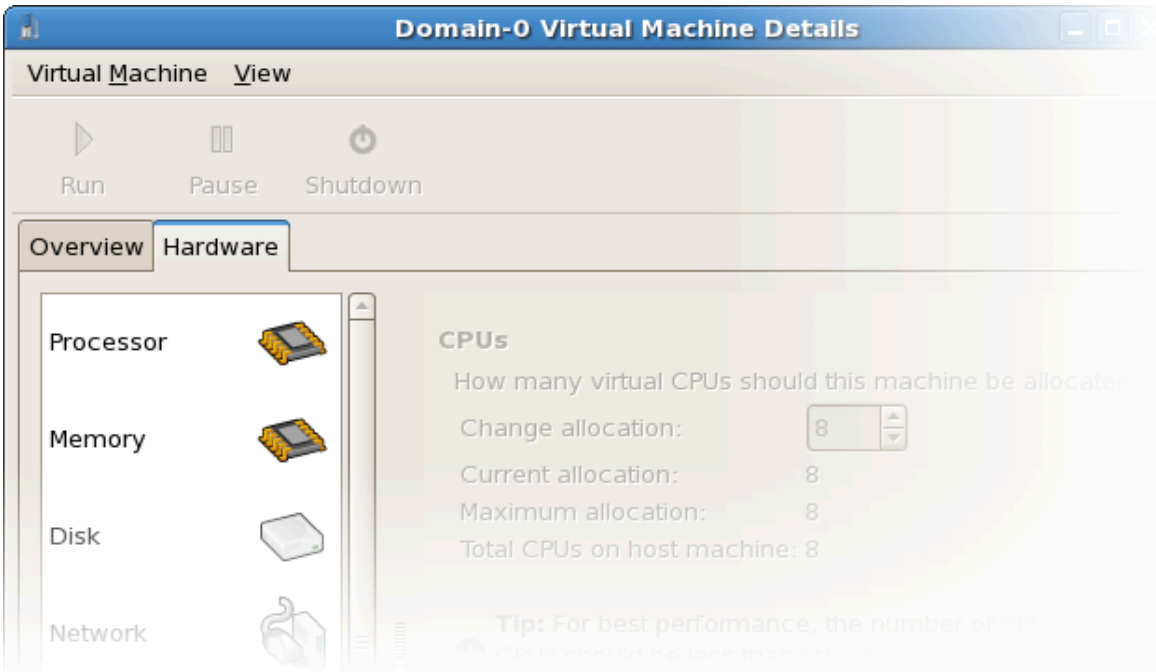


Рисунок 16.11. Обзор информации об оборудовании

4. Для просмотра или изменения числа виртуальных процессоров выберите **Процессор** (Processor).

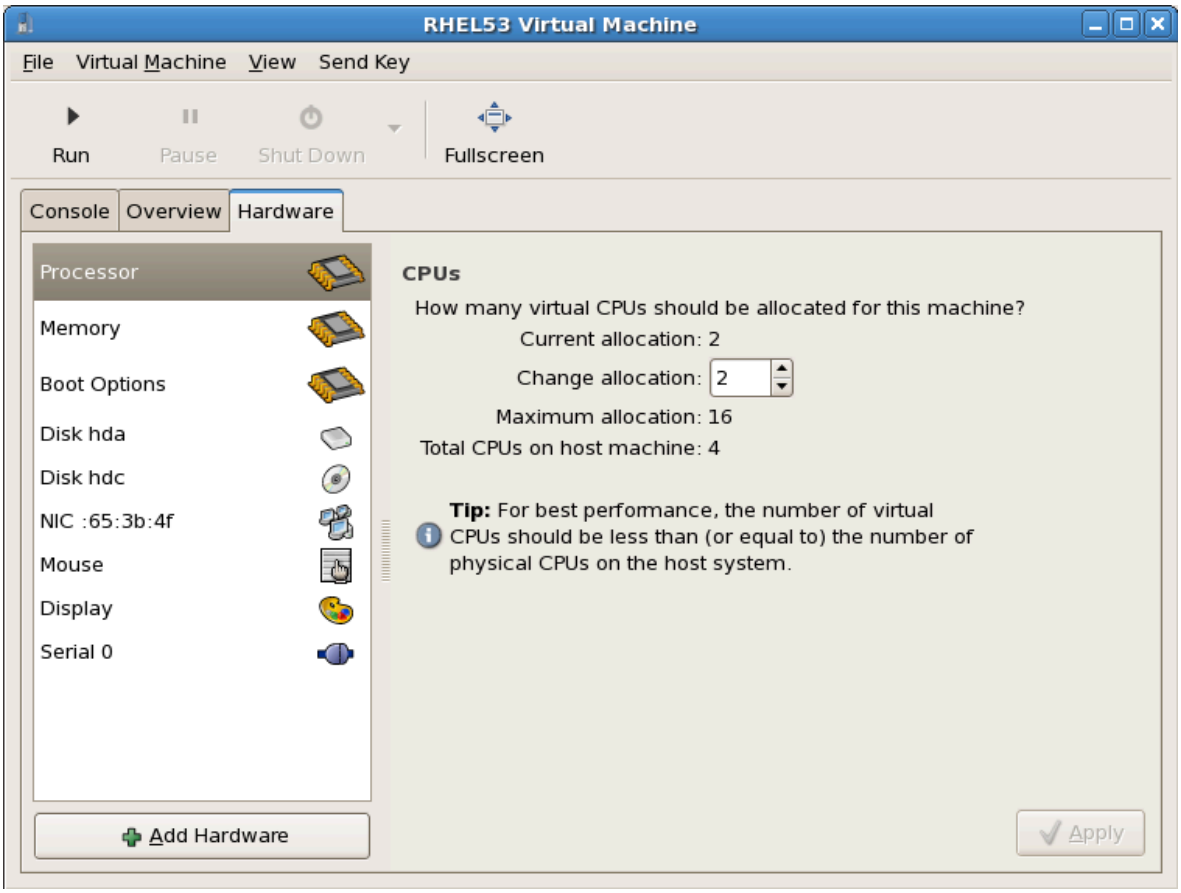


Рисунок 16.12. Панель распределения процессоров

5. Для просмотра или изменения распределения ресурсов памяти выберите **Память** (Memory).

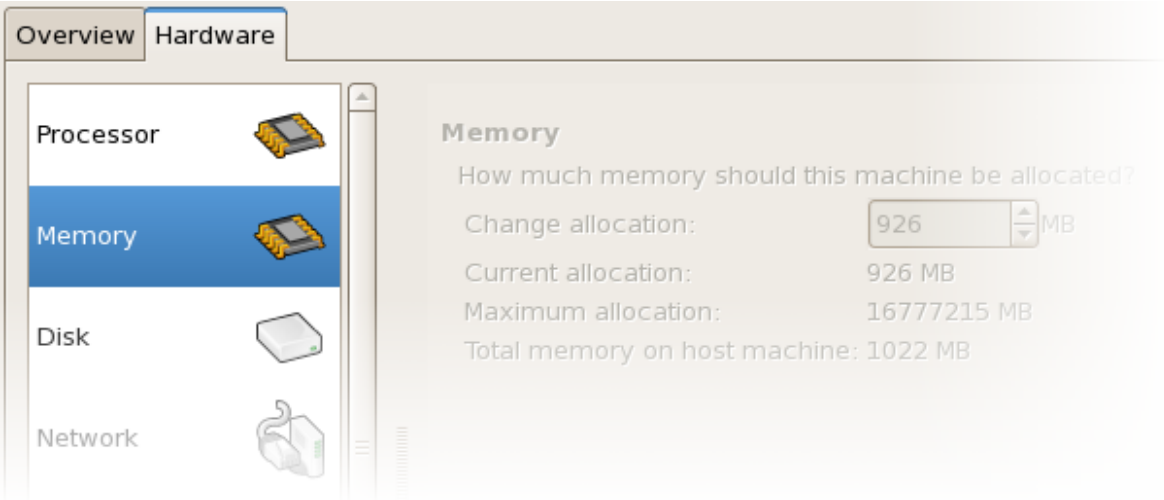


Рисунок 16.13. Панель распределения ресурсов памяти

6. Для просмотра или изменения дисковой конфигурации выберите **Диск** (Disk).

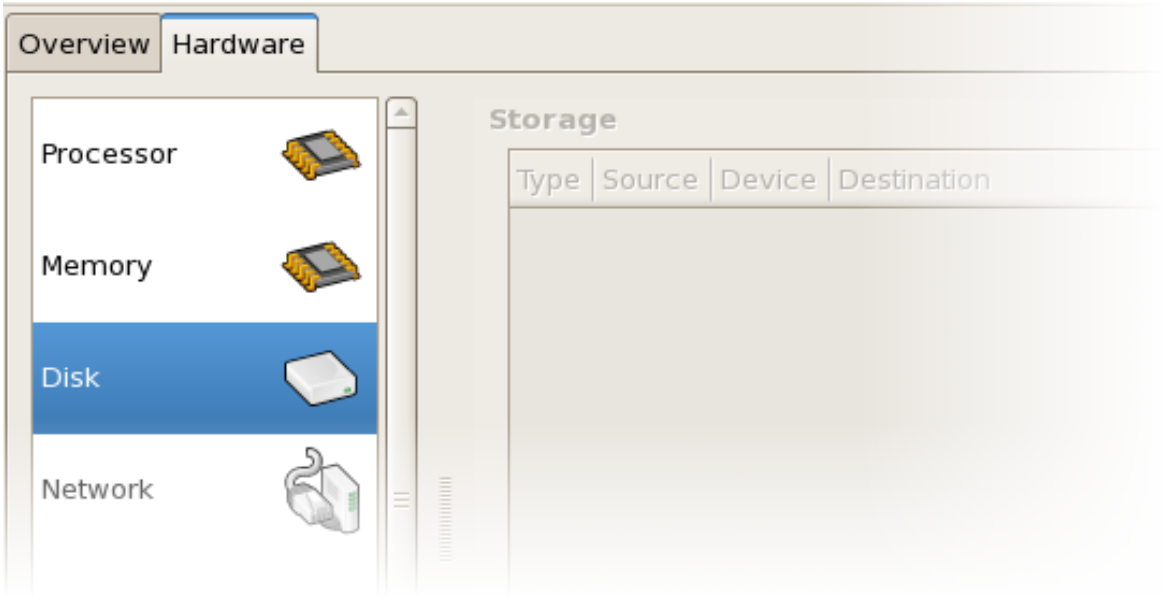


Рисунок 16.14. Панель дисковой конфигурации

7. Для просмотра или изменения сетевой конфигурации выберите **Сеть** (Network).

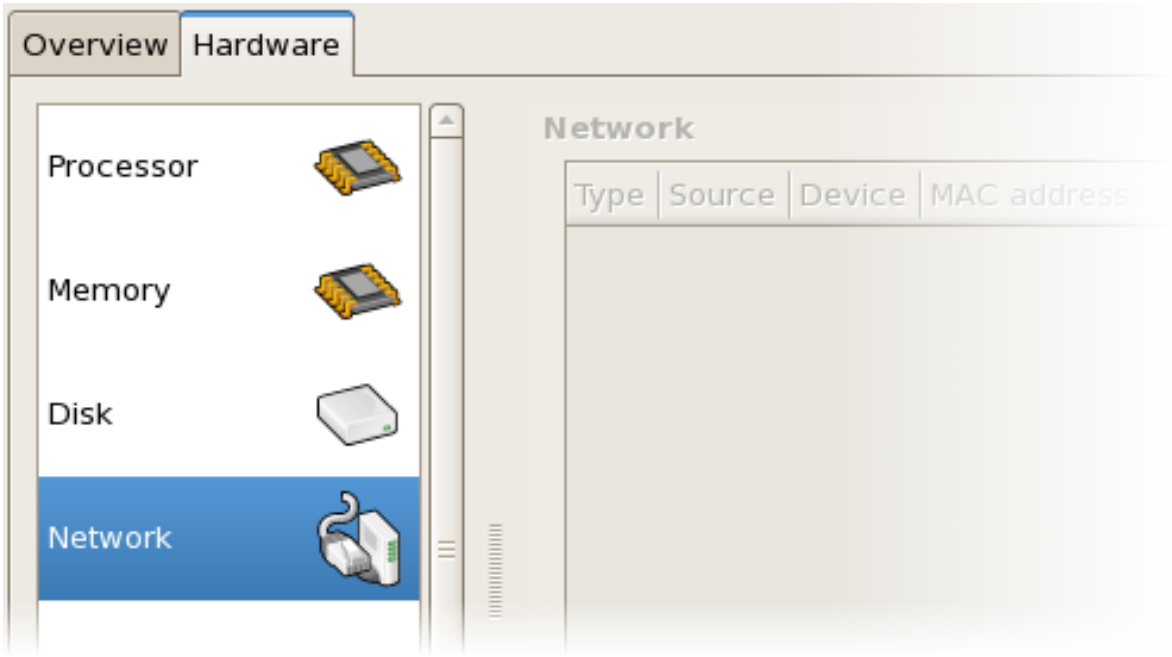


Рисунок 16.15. Панель сетевой конфигурации

16.8. Мониторинг состояния

С помощью менеджера можно изменить настройки контроля статуса.

Порядок действий при настройке мониторинга состояния и активации консолей:

1. В меню **Правка** (Edit) выберите **Параметры** (Preferences).

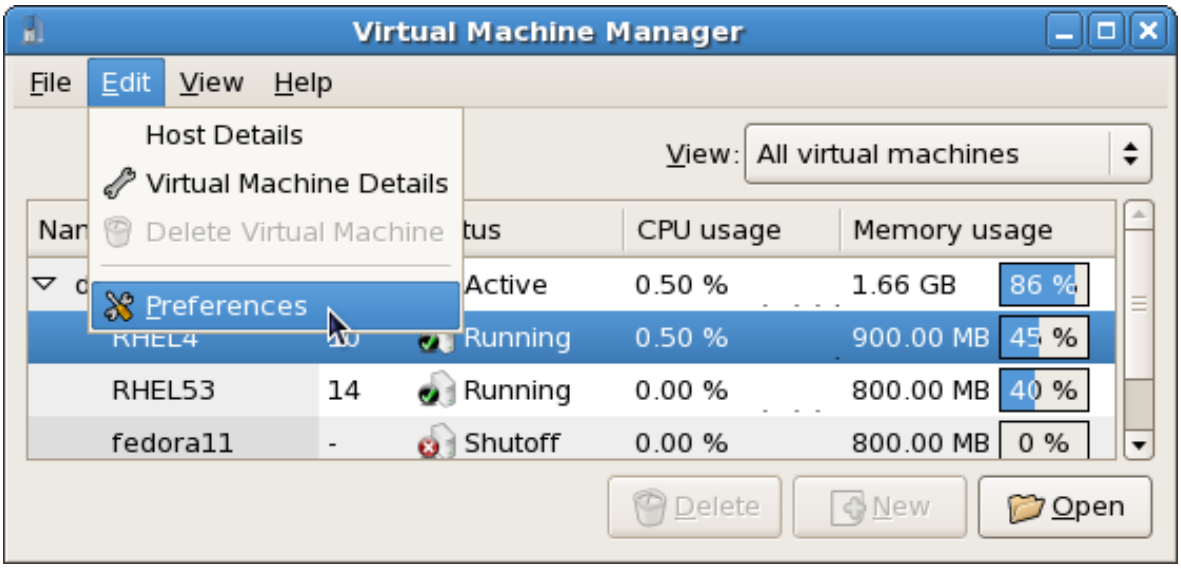


Рисунок 16.16. Изменение параметров гостевой машины

Появится окно параметров.

2. Укажите время обновления состояния виртуальной машины в секундах.

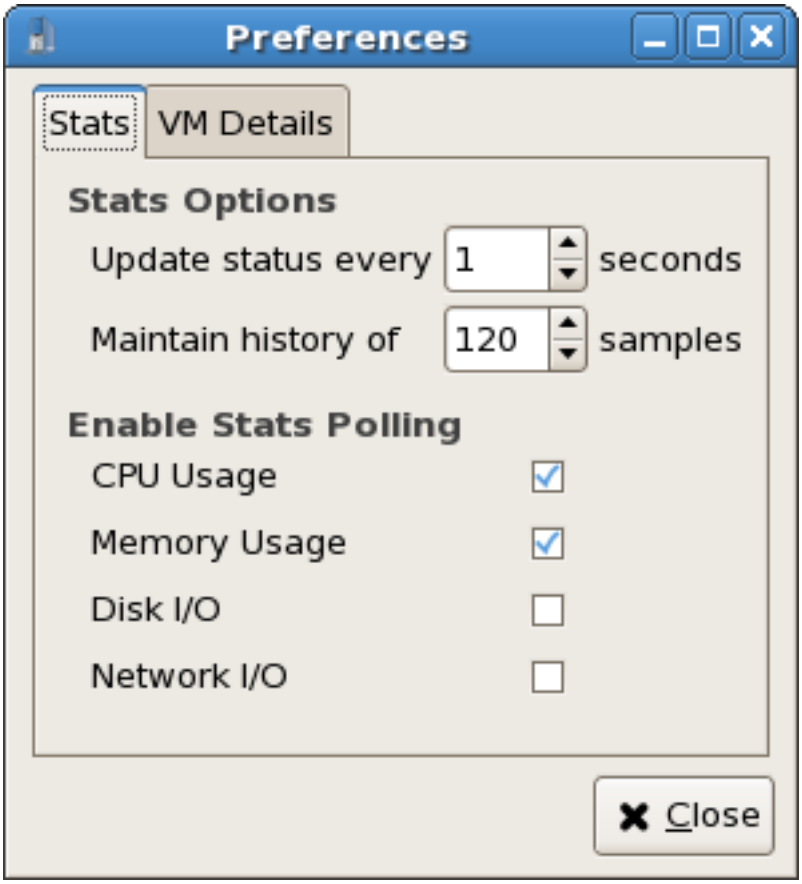


Рисунок 16.17. Настройка мониторинга состояния

3. В области консолей выберите, как открывать консоль, и укажите устройство ввода.

16.9. Просмотр идентификаторов виртуальных машин

Порядок действий при просмотре идентификаторов для всех виртуальных машин в системе:

1. В меню **Вид** (View) установите флажок **ID домена** (Domain ID).

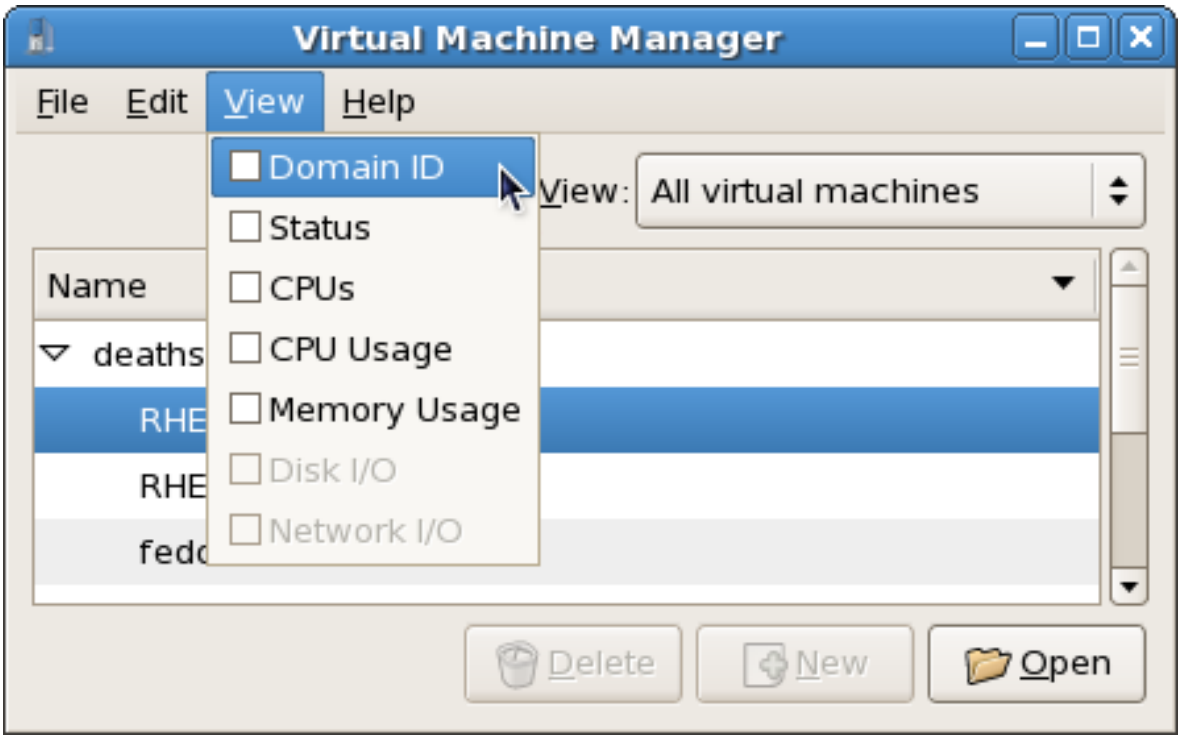


Рисунок 16.18. Выбор просмотра идентификаторов

2. Менеджер покажет идентификаторы всех доменов в системе.

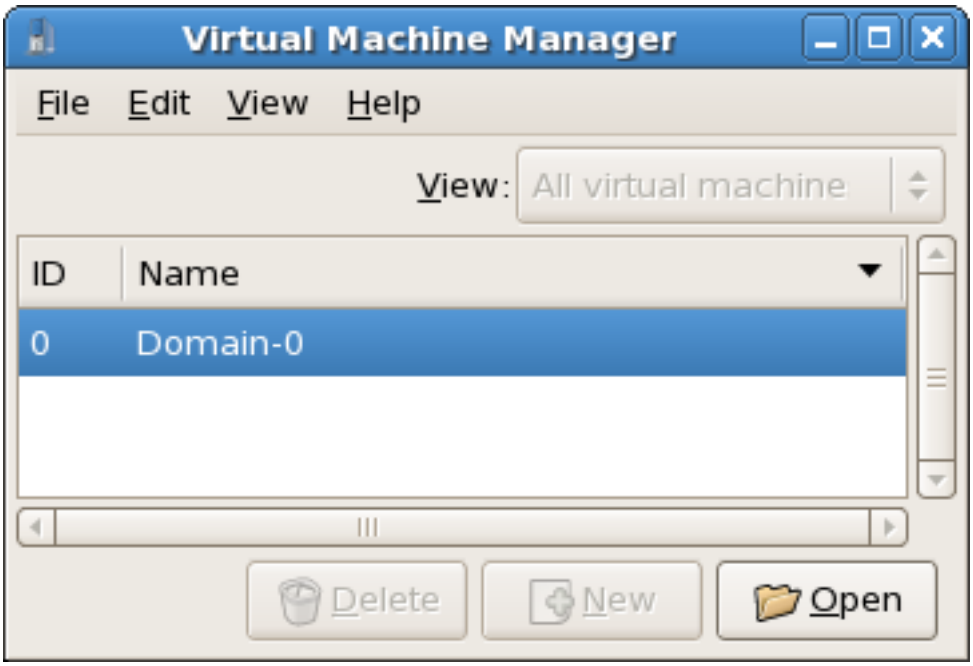


Рисунок 16.19. Просмотр идентификаторов доменов

16.10. Просмотр состояния гостевой системы

Порядок действий при просмотре состояния всех виртуальных машин в системе:

1. В меню **Вид** (View) установите флажок **Состояние** (Status).

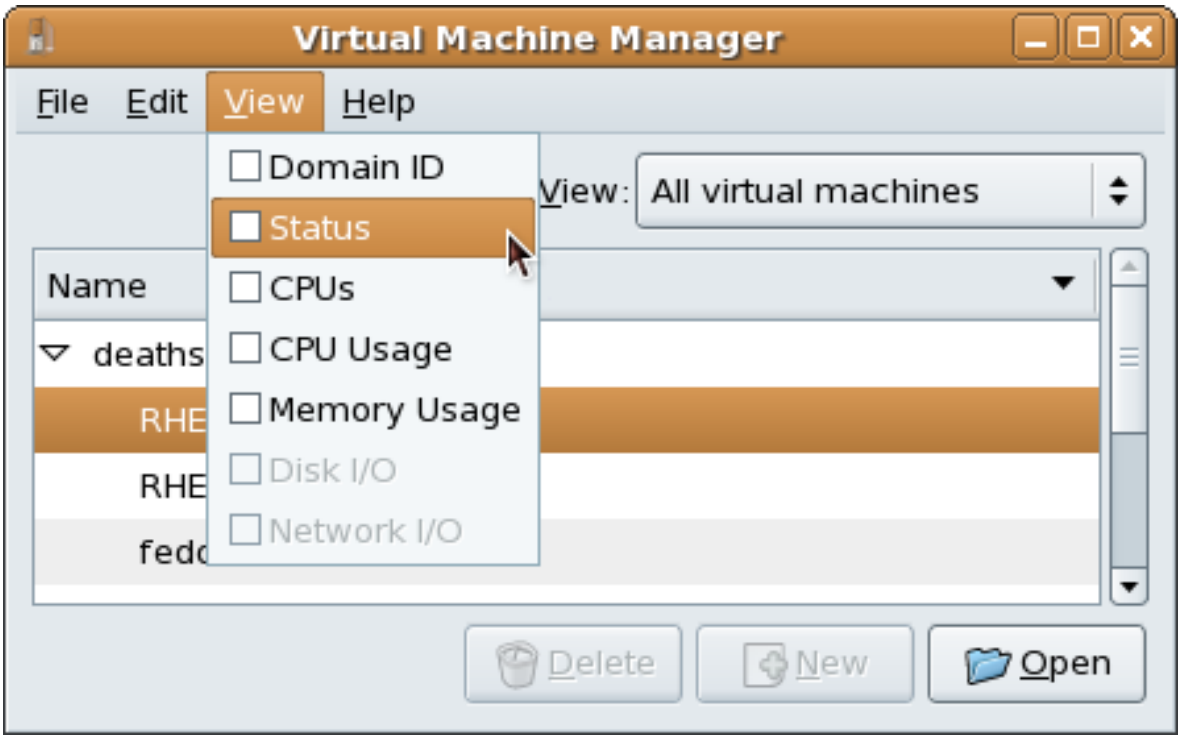


Рисунок 16.20. Выбор просмотра состояния виртуальной машины

2. Менеджер покажет состояние всех виртуальных машин в системе.

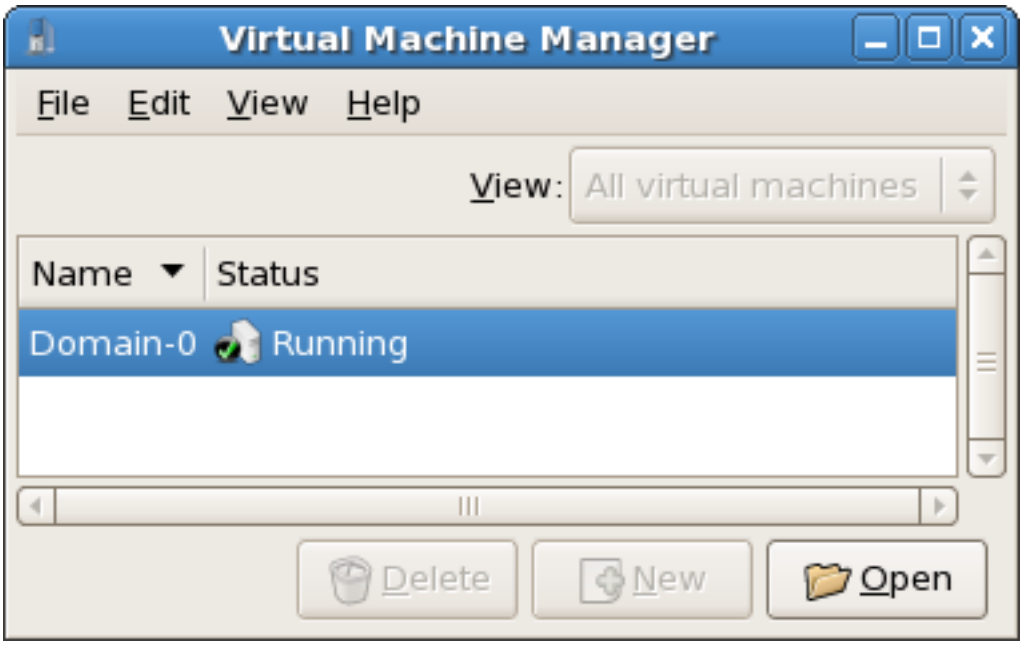


Рисунок 16.21. Просмотр состояния виртуальной машины

16.11. Просмотр виртуальных процессоров

Порядок действий при просмотре виртуальных процессоров для всех виртуальных машин в системе:

1. В меню **Вид** (View) установите флажок **Виртуальные процессоры** (Virtual CPUs).

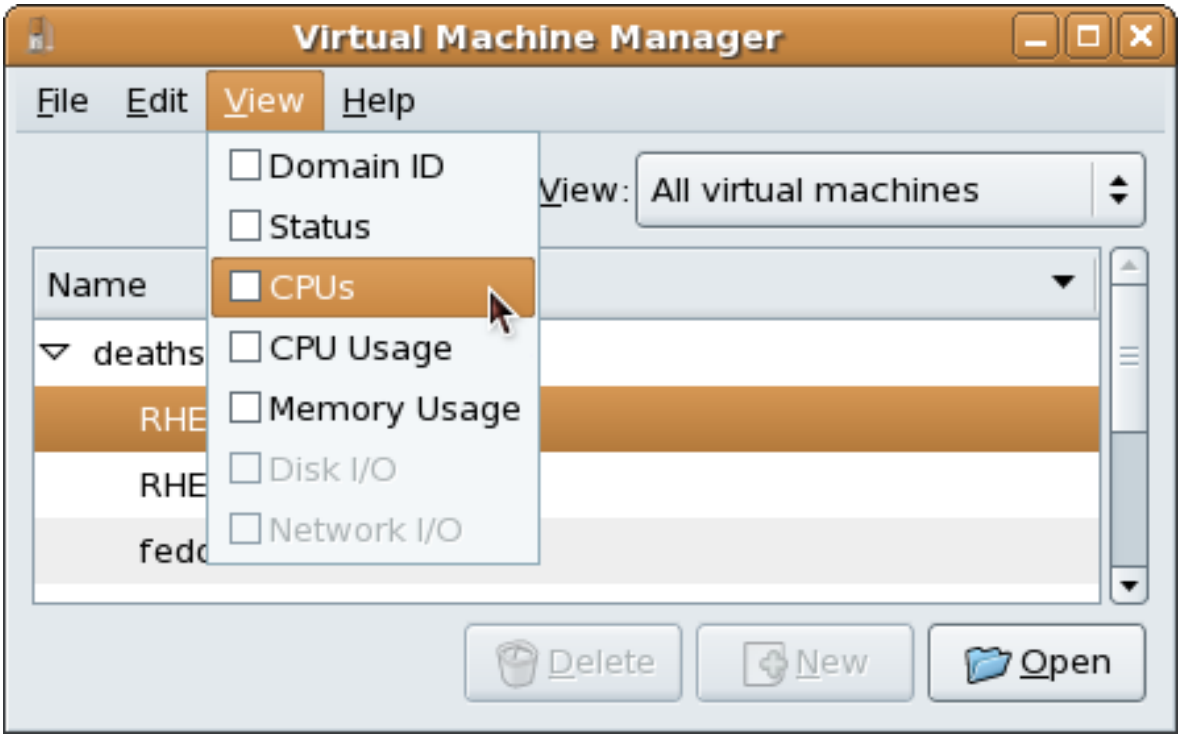


Рисунок 16.22. Выбор просмотра виртуальных процессоров

2. Менеджер покажет список виртуальных процессоров для всех виртуальных машин.

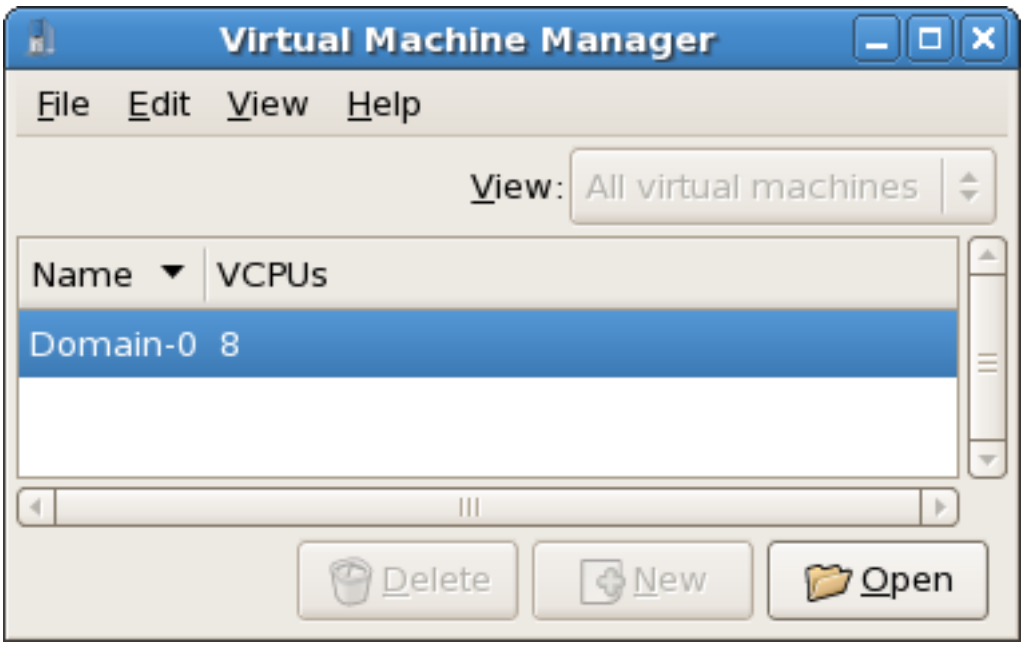


Рисунок 16.23. Просмотр виртуальных процессоров

16.12. Просмотр информации о занятости процессора

Порядок действий при просмотре информации о занятости процессоров:

1. В меню **Вид** (View) установите флажок **Использование процессора** (CPU Usage).

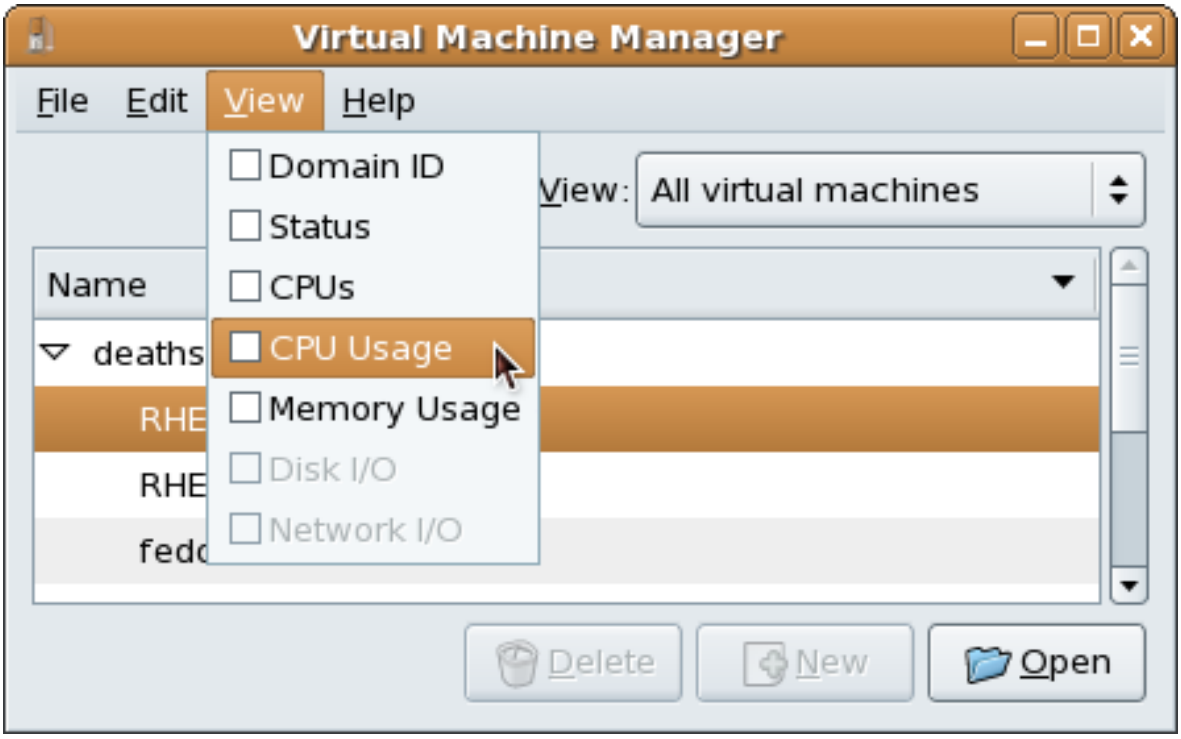


Рисунок 16.24. Выбор просмотра сведений о занятости процессора

2. Менеджер покажет информацию о занятости ресурсов процессоров (в процентах) для всех виртуальных машин в системе.

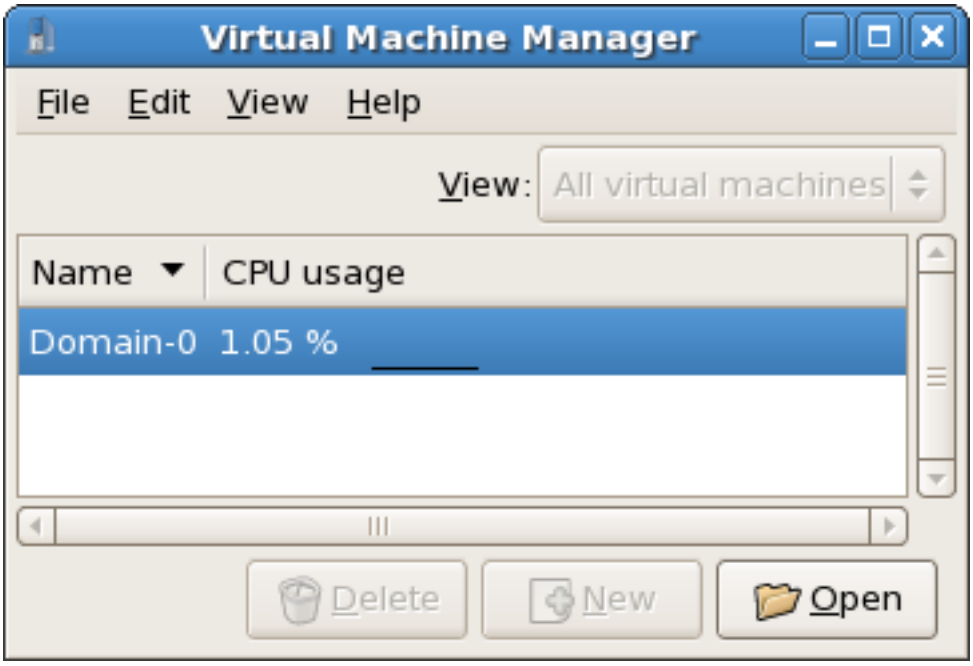


Рисунок 16.25. Просмотр информации о занятости процессора

16.13. Просмотр информации о занятости памяти

Порядок действий при просмотре информации о занятости ресурсов памяти:

1. В меню **Вид** (View) установите флажок **Использование памяти** (Memory Usage).

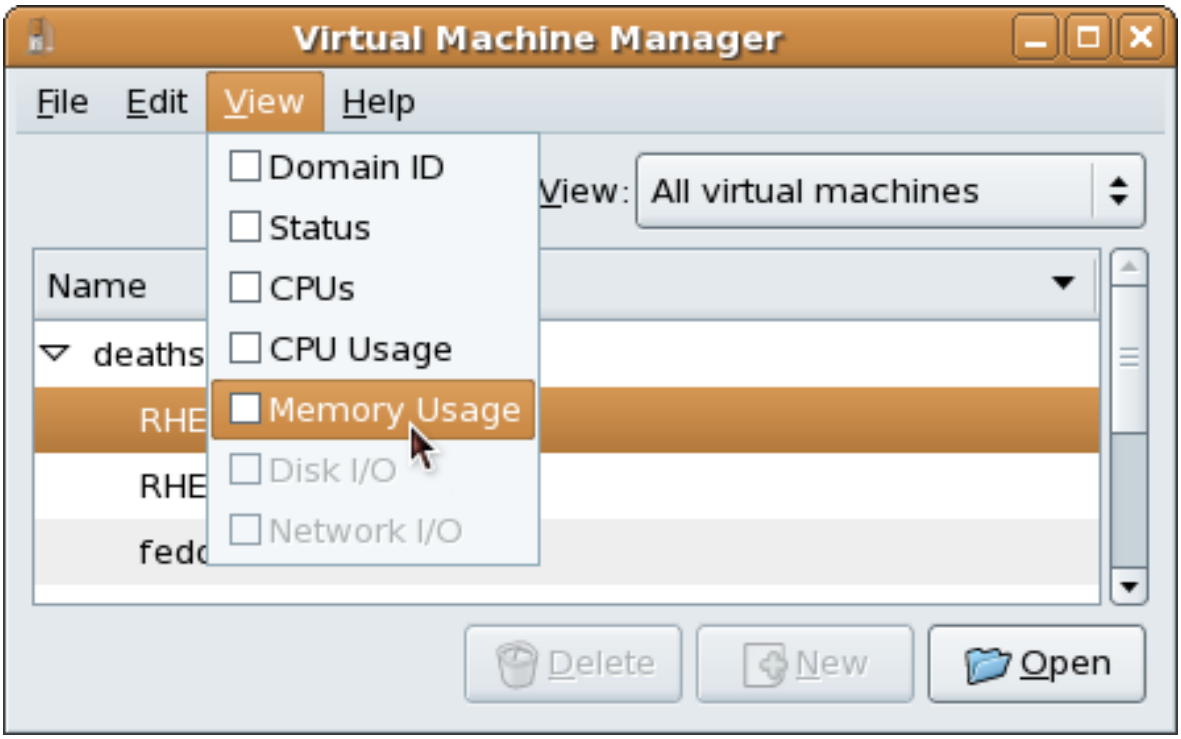


Рисунок 16.26. Выбор просмотра сведений о занятости памяти

2. Менеджер покажет сведения о занятости ресурсов памяти (в процентах) для всех виртуальных машин в системе.

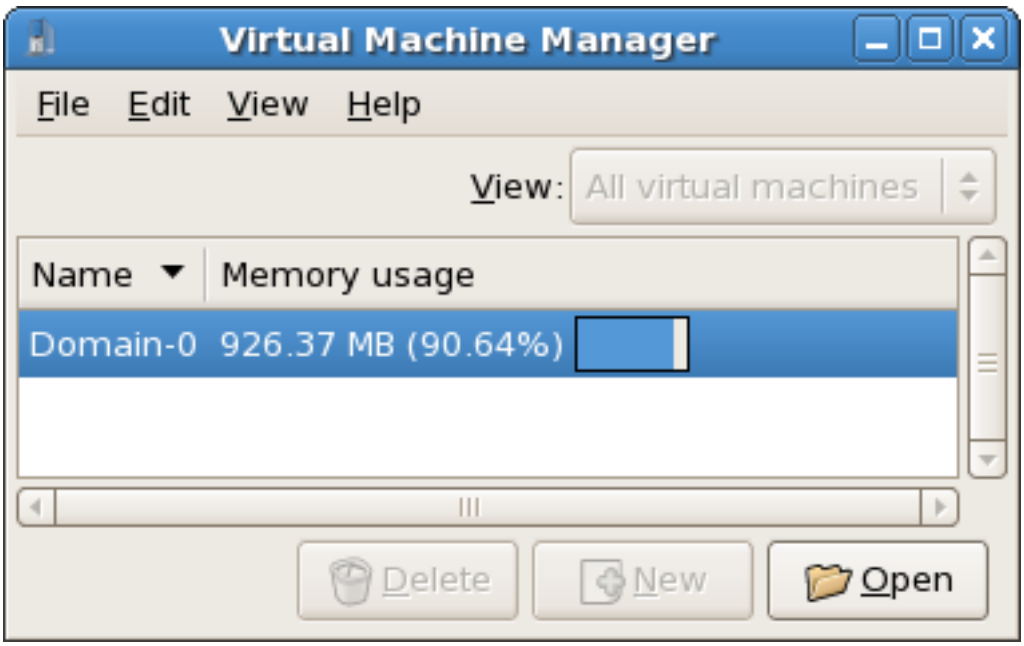


Рисунок 16.27. Просмотр информации о занятости памяти

16.14. Управление виртуальной сетью

Порядок действий при настройке виртуальной сети в системе:

1. В меню **Правка** (Edit) выберите **Параметры хоста** (Host Details).

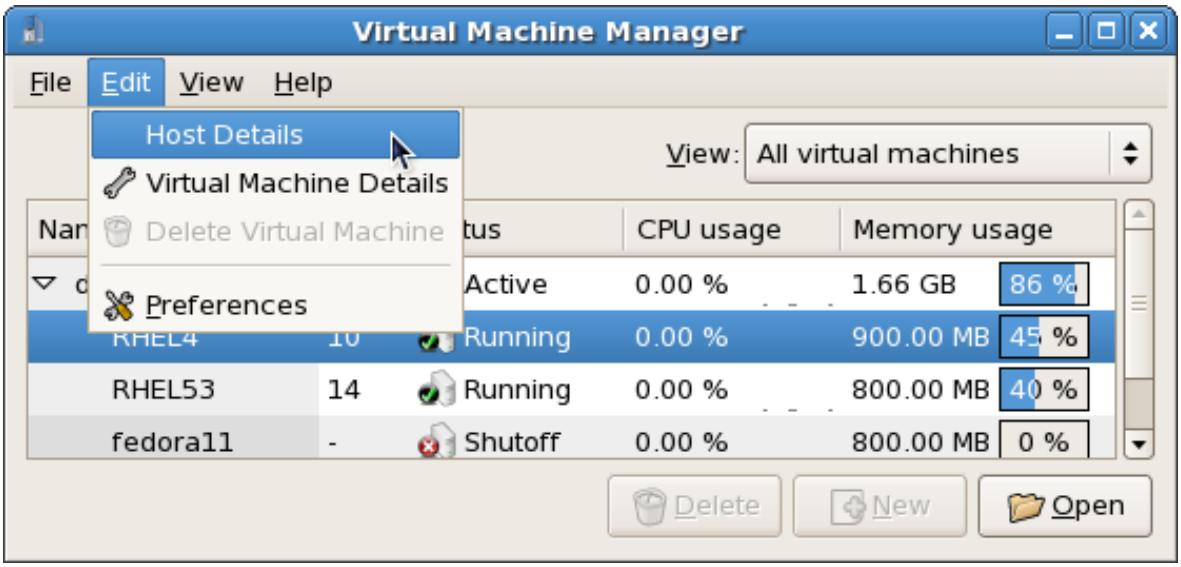


Рисунок 16.28. Выбор просмотра параметров узла

2. В открывшемся окне перейдите на вкладку **Виртуальные сети** (Virtual Networks).

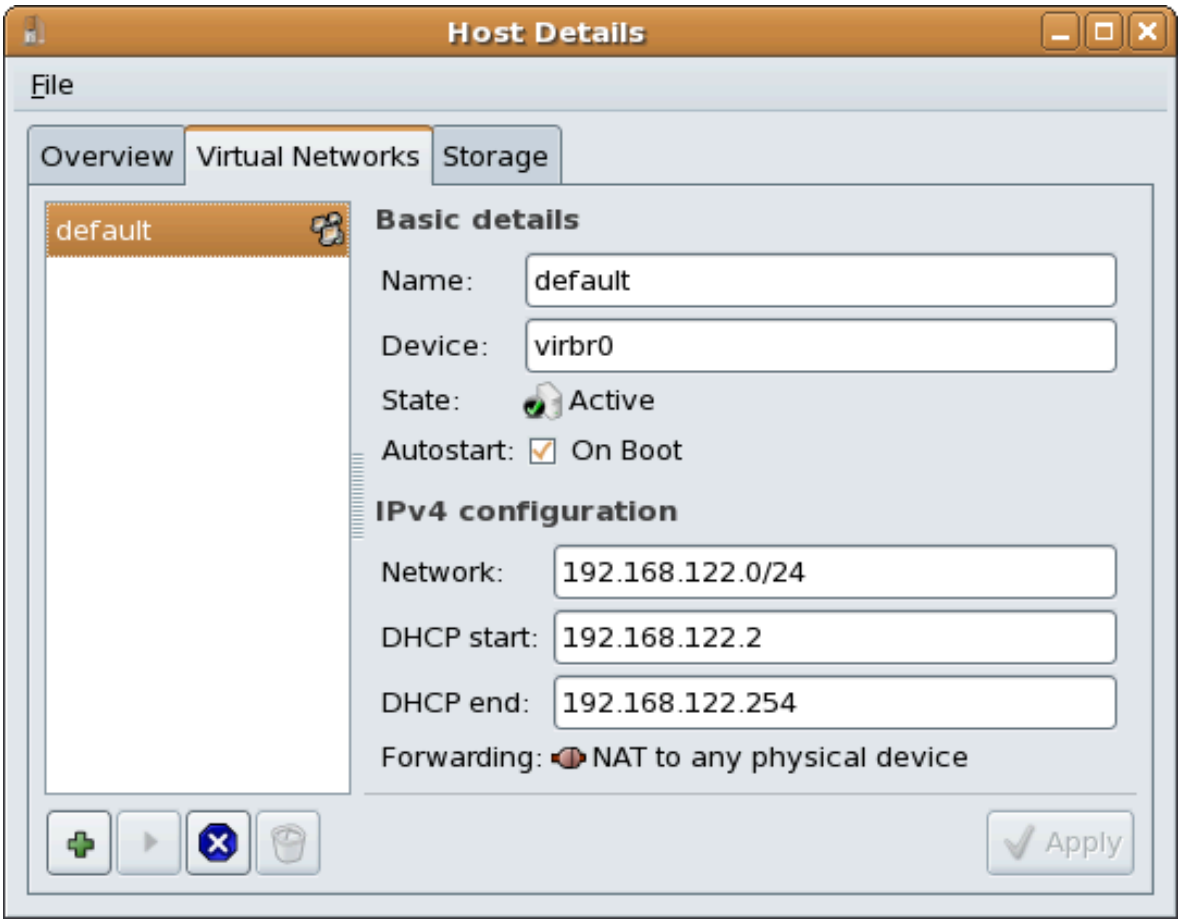


Рисунок 16.29. Окно параметров виртуальной сети

3. Доступные виртуальные сети будут перечислены в левой части окна. Выберите сеть для доступа к ее настройкам.

16.15. Создание виртуальной сети

Порядок действий при создании виртуальной сети:

1. Откройте меню параметров узла (см. *Раздел 16.14, «Управление виртуальной сетью»*) и нажмите кнопку **Добавить** (Add).

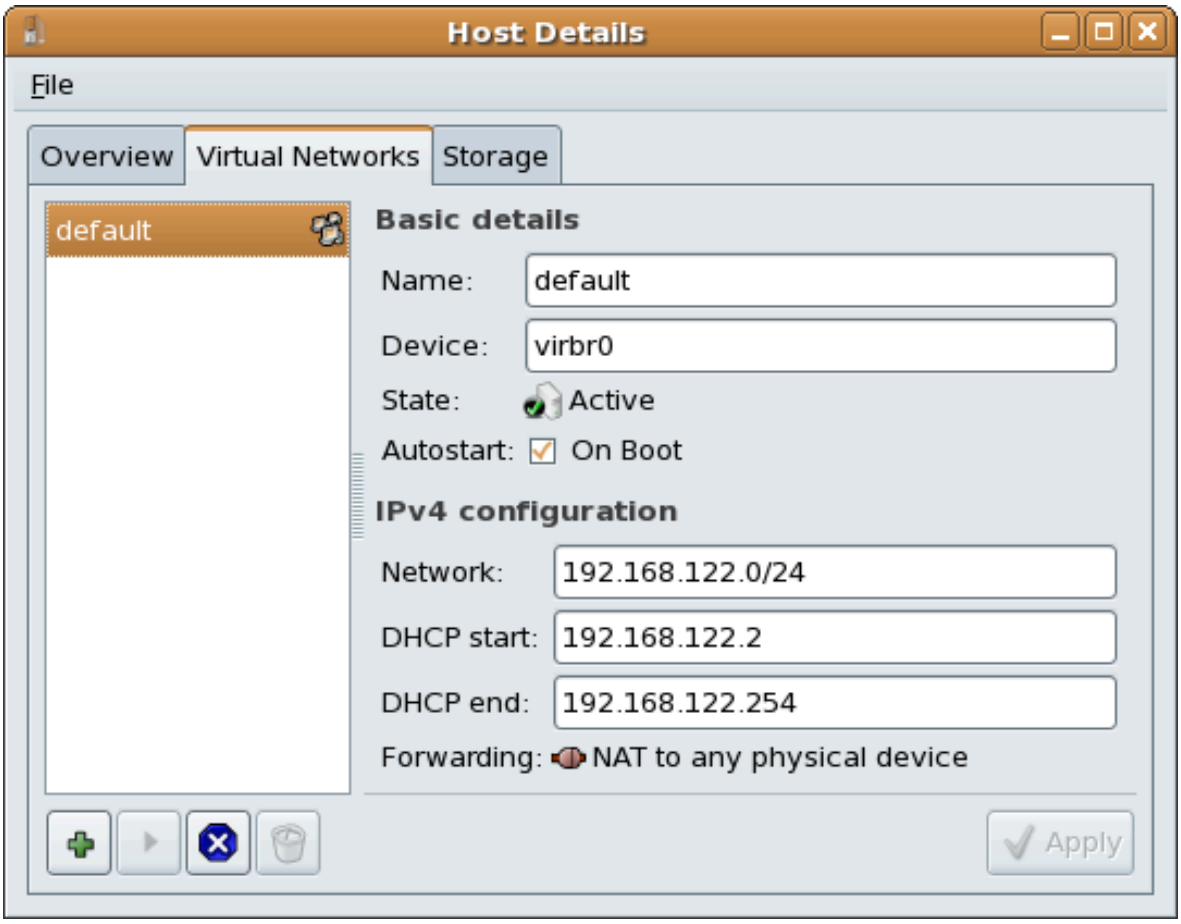


Рисунок 16.30. Окно параметров виртуальной сети

В открывшемся окне нажмите кнопку продолжения.

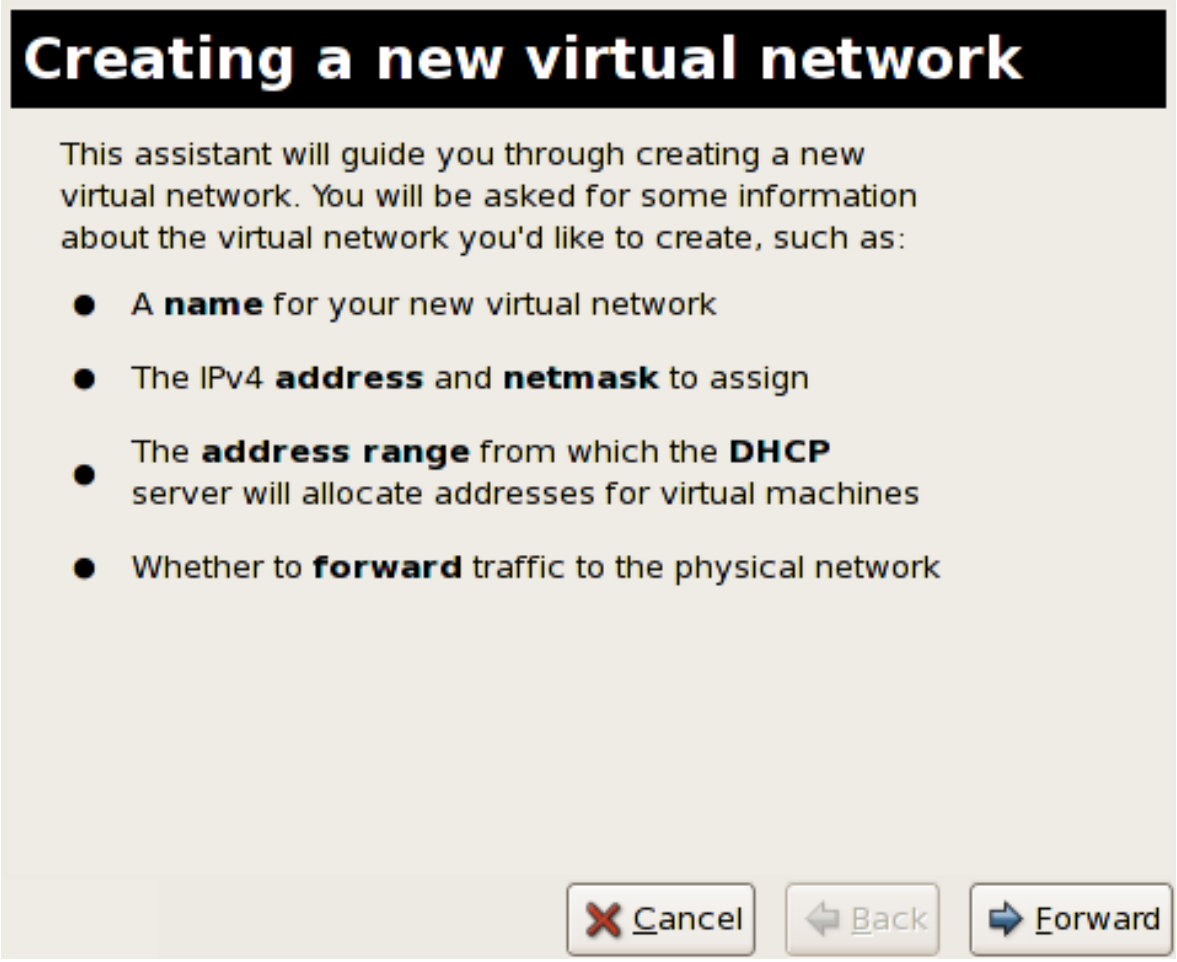


Рисунок 16.31. Создание виртуальной сети

2. Введите имя для новой сети нажмите **Далее** (Forward).

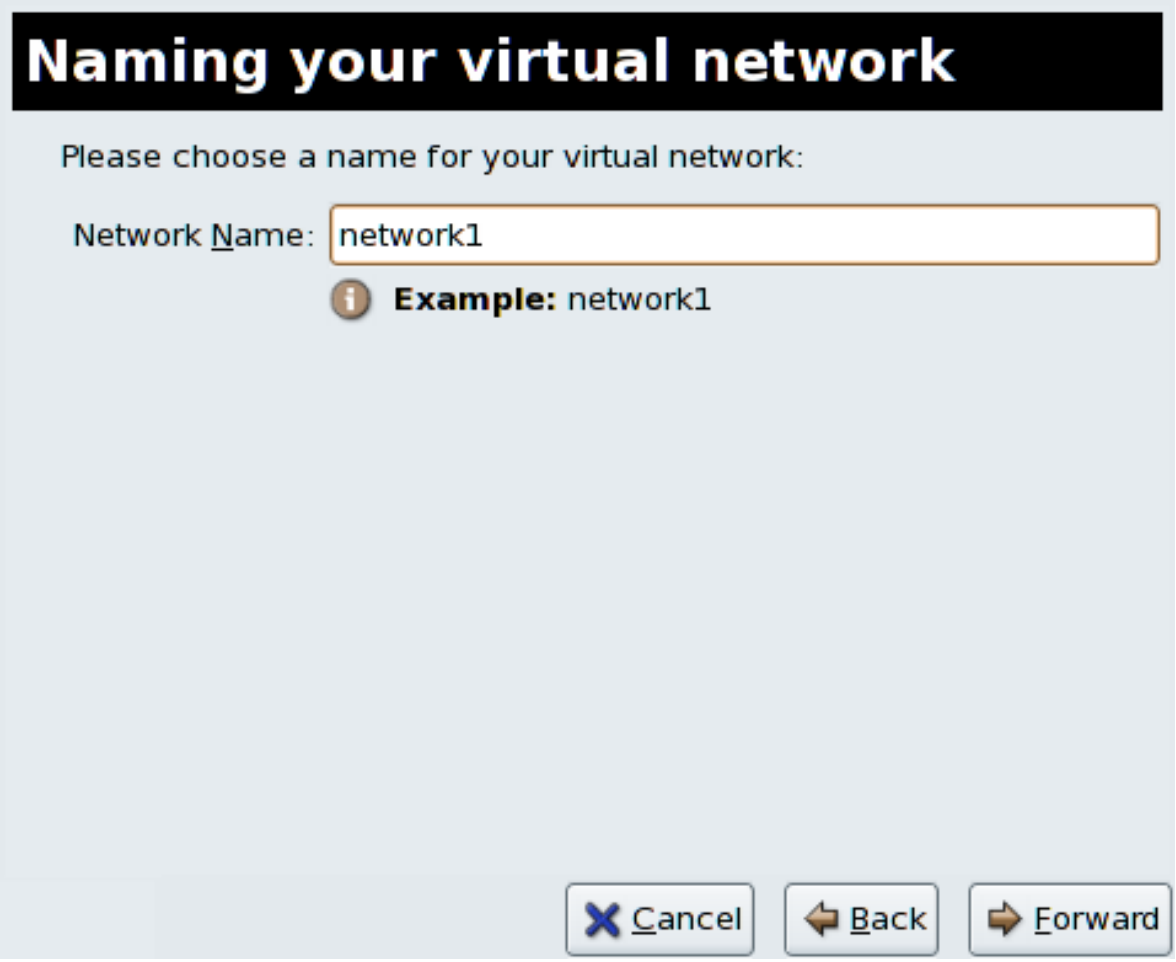


Рисунок 16.32. Присвоение имени сети

3. Введите пространство адресов IPv4 для виртуальной сети и нажмите **Далее** (Forward).

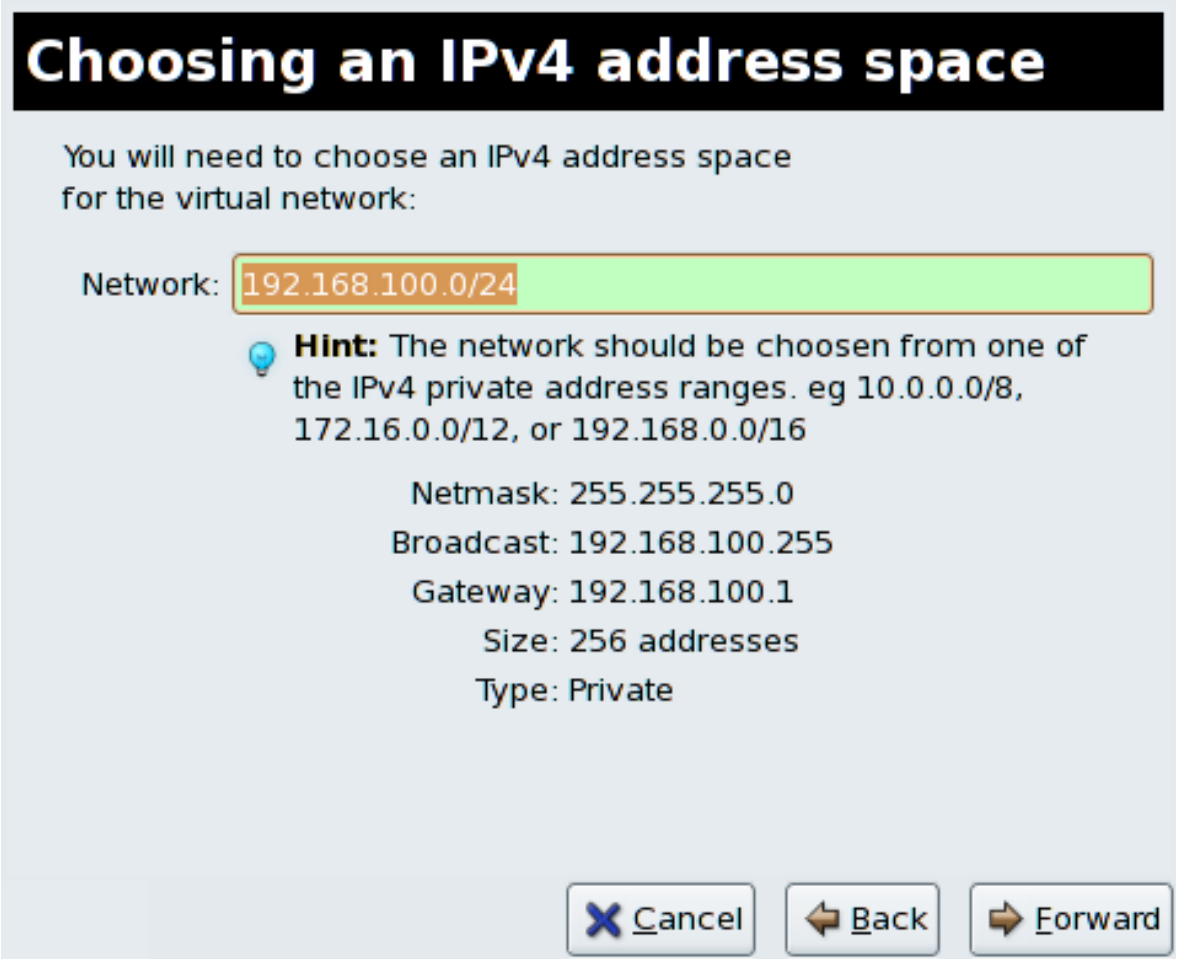


Рисунок 16.33. Выбор пространства адресов IPv4

4. Укажите диапазон DHCP для вашей виртуальной сети, задав начальный и конечный адрес. Нажмите кнопку продолжения.

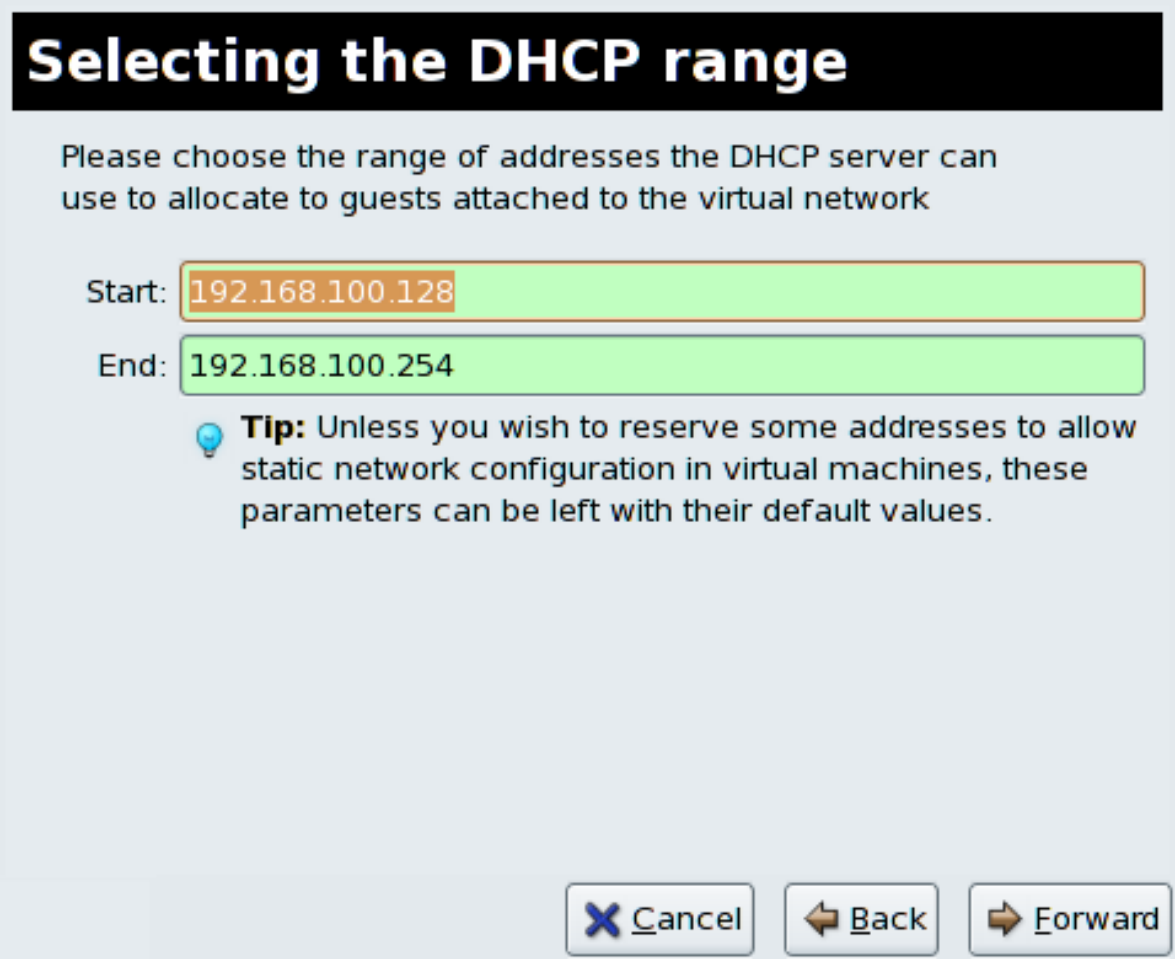


Рисунок 16.34. Выбор диапазона DHCP

5. Выберите способ подключения виртуальной сети к физической.

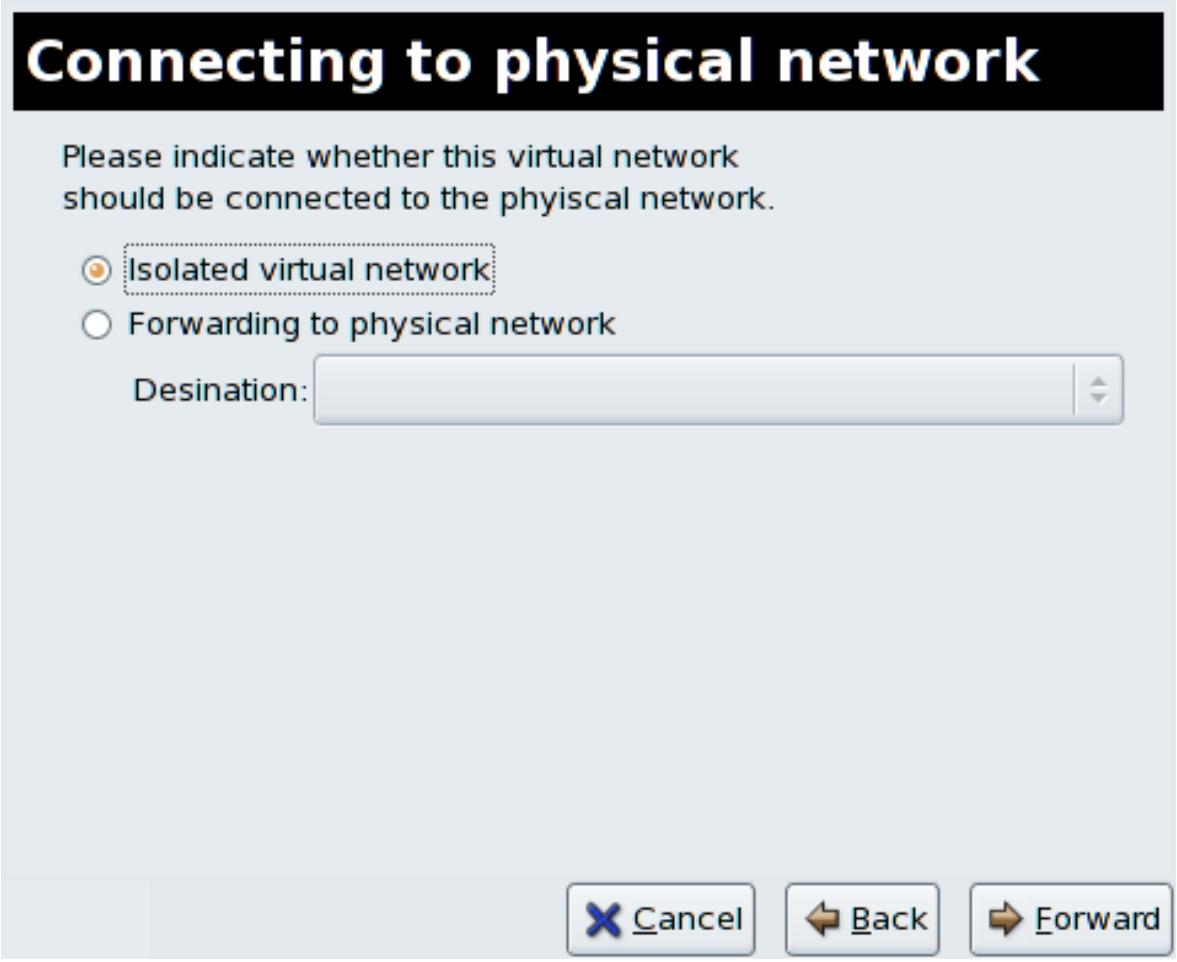


Рисунок 16.35. Подключение к физической сети

Если вы выбрали **Перенаправление в физическую сеть** (Forwarding to physical network), в поле **Назначение** (Destination) выберите либо **NAT на любое физическое устройство** (NAT to any physical device), либо **NAT на физическое устройство eth0** (NAT to physical device eth0).

Click **Forward** to continue.

6. Все готово для создания сети. Проверьте конфигурацию сети и нажмите **Готово** (Finish).

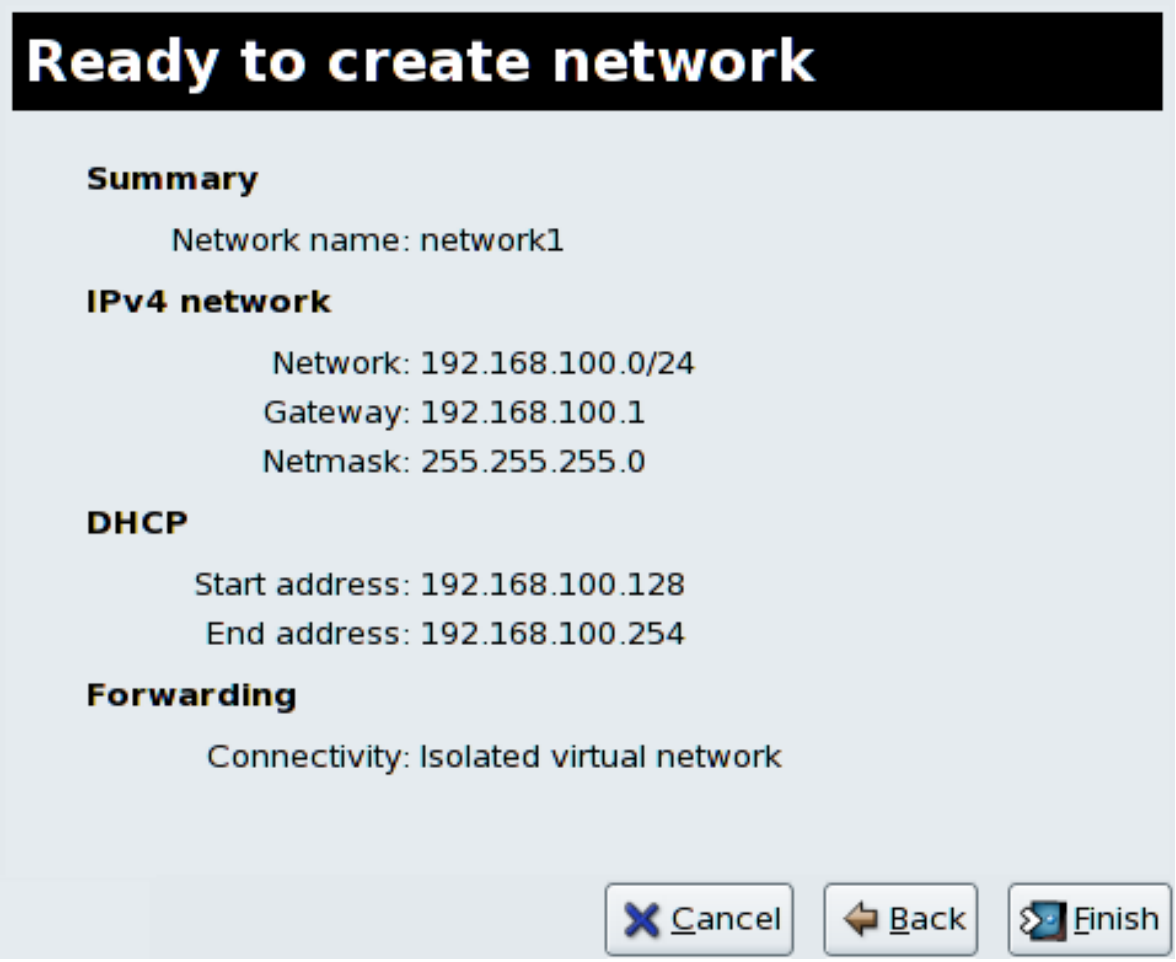


Рисунок 16.36. Все готово для создания сети

7. Сведения о новой виртуальной сети можно получить на вкладке **Виртуальные сети** (Virtual Network) меню **Параметры хоста** (Host Details).

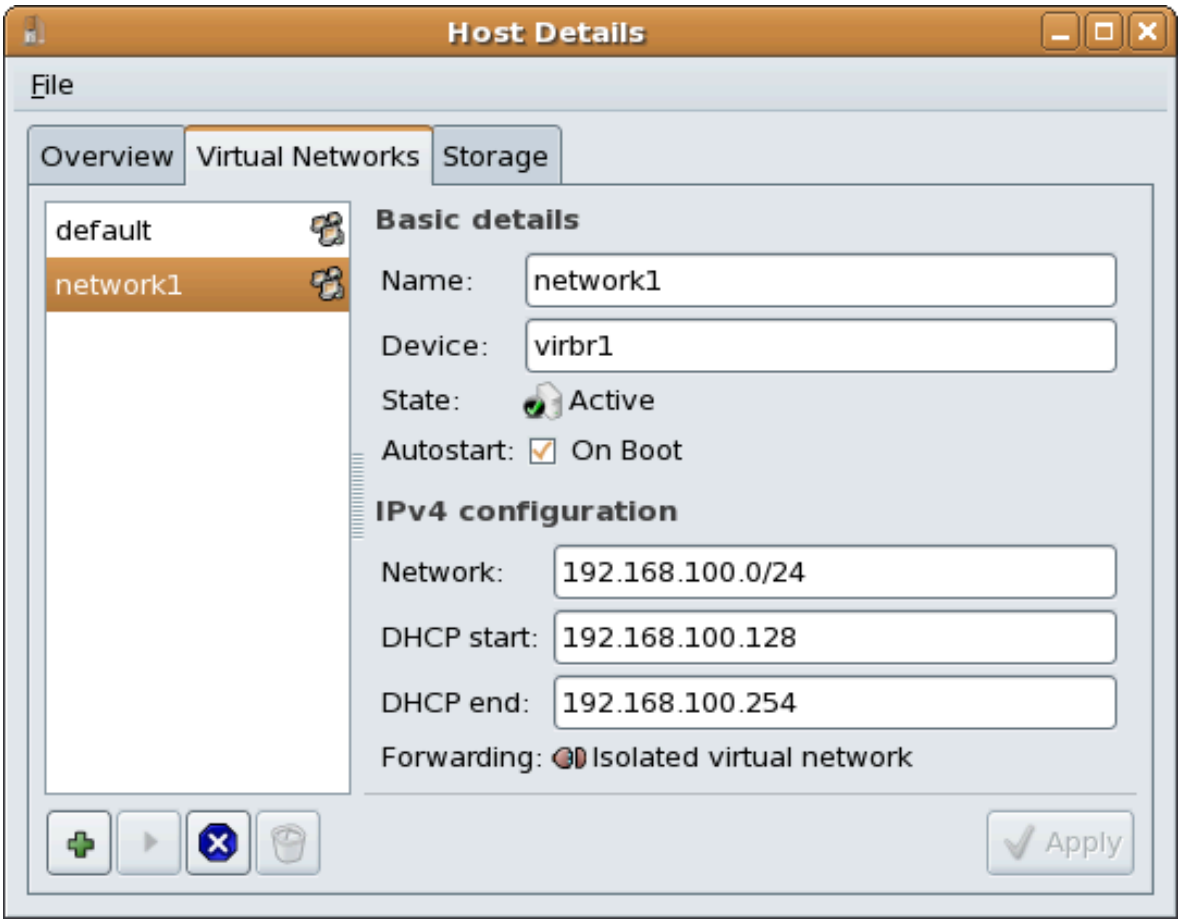


Рисунок 16.37. Новая виртуальная сеть теперь доступна

Часть V. Tips and Tricks

Советы и хитрости по улучшению производительности

В последующих главах будут приведены советы для повышения производительности, надежности и масштабируемости окружений виртуализации.

Советы и хитрости

В этой главе описаны различные приемы для повышения производительности, надежности и масштабируемости окружений виртуализации.

17.1. Автоматический запуск виртуальных машин

Здесь рассмотрена настройка автоматического запуска виртуальных машин во время загрузки размещающей системы.

Приведенный пример использует **`virsh`** для настройки автоматического запуска гостя с именем *`TestServer`* во время загрузки размещающей системы.

```
# virsh autostart TestServer
Domain TestServer marked as autostarted
```

Автоматический запуск виртуальной машины настроен.

Чтобы отключить автоматическую загрузку, выполните следующую команду с параметром `--disable`:

```
# virsh autostart --disable TestServer
Domain TestServer unmarked as autostarted
```

Автоматический запуск виртуальной машины теперь отключен.

17.2. Переключение между гипервизорами KVM и Xen

Здесь рассматривается переключение между гипервизорами KVM и Xen

Fedora разрешает выполнение только одного гипервизора в заданный момент времени.

Миграция виртуализированных гостевых систем между гипервизорами

В настоящее время нет специальных программ для переноса гостевых систем с Xen на KVM и наоборот. Гостевые системы должны выполняться только на гипервизоре, тип которого не отличается от типа гипервизора, на котором гость был создан.

17.2.1. Xen на KVM

Далее будет рассмотрен процесс изменения гипервизора Xen на KVM. Подразумевается, что пакет *kernel-xen* уже установлен и работает.

1. **Установите пакет KVM**
   Установите пакет *kvm*, если он еще не установлен.

   ```
   # yum install kvm
   ```

2. **Проверьте версию используемого ядра**
   В системе может быть установлен пакет *kernel-xen*, поэтому проверьте версию работающего ядра с помощью команды **uname**:

```
$ uname -r
2.6.23.14-107.fc8xen
```

   То есть выполняется ядро **2.6.23.14-107.fc8xen**. Если в результате выполнения команды вы получили версию ядра, которое используется по умолчанию, а именно **2.6.23.14-107.fc8**, можно сразу перейти к пункту 3.

- **Изменение ядра Xen на стандартное ядро**
  Выбор ядра для загрузки осуществляется в файле **grub.conf**. Чтобы изменить используемое по умолчанию ядро, внесите изменения в **/boot/grub/grub.conf**.

```
default=1
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Fedora (2.6.23.14-107.fc8)
        root (hd0,0)
        kernel /vmlinuz-2.6.23.14-107.fc8 ro root=/dev/VolGroup00/
LogVol00 rhgb quiet
        initrd /initrd-2.6.23.14-107.fc8.img
title Fedora (2.6.23.14-107.fc8xen)
        root (hd0,0)
        kernel /xen.gz-2.6.23.14-107.fc8
        module /vmlinuz-2.6.23.14-107.fc8xen ro root=/dev/
VolGroup00/LogVol00 rhgb quiet
        module /initrd-2.6.23.14-107.fc8xen.img
```

  Обратите внимание на выражение **default=1**, которое определяет порядок ядра в списке. Так, в этом случае GRUB использует вторую запись, то есть ядро Xen. Измените значение на *0*, чтобы использовалось первое ядро в списке:

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Fedora (2.6.23.14-107.fc8)
        root (hd0,0)
        kernel /vmlinuz-2.6.23.14-107.fc8 ro root=/dev/VolGroup00/
LogVol00 rhgb quiet
        initrd /initrd-2.6.23.14-107.fc8.img
title Fedora (2.6.23.14-107.fc8xen)
        root (hd0,0)
        kernel /xen.gz-2.6.23.14-107.fc8
        module /vmlinuz-2.6.23.14-107.fc8xen ro root=/dev/
VolGroup00/LogVol00 rhgb quiet
        module /initrd-2.6.23.14-107.fc8xen.img
```

3. **Перезагрузите систему с новым ядром**
   Перезагрузите систему. Модуль KVM загрузится автоматически вместе с ядром. Убедитесь, что он выполняется:

   ```
   $ lsmod | grep kvm
   kvm_intel              85992  1
   kvm                   222368  2 ksm,kvm_intel
   ```

   Если список содержит модуль **kvm**, а также **kvm_intel** или **kvm_amd**, то все работает нормально.

17.2.2. KVM на Xen

Далее будет рассмотрен процесс изменения гипервизора KVM на Xen. Подразумевается, что пакет *kvm* уже установлен и работает.

1. **Установите пакеты Xen**
   Установите пакеты *kernel-xen* и *xen*, если они еще не установлены.

   ```
   # yum install kernel-xen xen
   ```

   Пакет *kernel-xen* вполне может быть уже установлен, но отключен.

2. **Проверьте версию используемого ядра**
   Проверьте версию работающего ядра с помощью команды **uname**:

   ```
   $ uname -r
   2.6.23.14-107.fc8
   ```

   То есть выполняется ядро **2.6.23.14-107.fc8**, которое используется по умолчанию. Если в результате выполнения команды вы получили версию ядра, в конце имени которого есть **xen** (например, **2.6.23.14-107.fc8xen**), можно сразу перейти к пункту 3.

   - **Изменение стандартного ядра на ядро Xen**
     Выбор ядра для загрузки осуществляется в файле **grub.conf**. Чтобы изменить используемое по умолчанию ядро, внесите изменения в **/boot/grub/grub.conf**.

     ```
     default=0
     timeout=5
     splashimage=(hd0,0)/grub/splash.xpm.gz
     hiddenmenu
     title Fedora (2.6.23.14-107.fc8)
             root (hd0,0)
             kernel /vmlinuz-2.6.23.14-107.fc8 ro root=/dev/VolGroup00/
     LogVol00 rhgb quiet
             initrd /initrd-2.6.23.14-107.fc8.img
     title Fedora (2.6.23.14-107.fc8xen)
             root (hd0,0)
             kernel /xen.gz-2.6.23.14-107.fc8
     ```

```
        module /vmlinuz-2.6.23.14-107.fc8xen ro root=/dev/
VolGroup00/LogVol00 rhgb quiet
        module /initrd-2.6.23.14-107.fc8xen.img
```

Обратите внимание на выражение **default=0**, которое определяет порядок ядра в списке. Так, в этом случае GRUB использует первую запись, то есть стандартное ядро. Измените значение на *1*, чтобы использовалось ядро Xen:

```
default=1
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Fedora (2.6.23.14-107.fc8)
        root (hd0,0)
        kernel /vmlinuz-2.6.23.14-107.fc8 ro root=/dev/VolGroup00/
LogVol00 rhgb quiet
        initrd /initrd-2.6.23.14-107.fc82.6.23.14-107.fc8.img
title Fedora (2.6.23.14-107.fc8xen)
        root (hd0,0)
        kernel /xen.gz-2.6.23.14-107.fc8
        module /vmlinuz-2.6.23.14-107.fc8xen ro root=/dev/
VolGroup00/LogVol00 rhgb quiet
        module /initrd-2.6.23.14-107.fc8xen.img
```

3. **Перезагрузите систему с новым ядром**
   Перезагрузите систему с ядром Xen. Проверьте версию с помощью команды **uname**:

   ```
   $ uname -r
   2.6.23.14-107.fc8xen
   ```

   Если в результате выполнения команды вы получили версию ядра, в конце имени которого есть `xen`, значит, выполняется ядро Xen.

17.3. qemu-img

Текстовая утилита **qemu-img** применяется для форматирования различных файловых систем, используемых Xen и KVM. Именно с ее помощью следует выполнять форматирование виртуализированных гостевых систем, дополнительных устройств хранения и сетевых хранилищ. Ниже будут рассмотрены параметры и формат **qemu-img**.

Форматирование и создание новых устройств и образов

Команда создания нового образа диска:

```
# qemu-img create [-6] [-e] [-b базовый_образ] [-f формат] имя_файла
 [размер]
```

Если задан «базовый_образ», то полученный образ будет содержать только отличия. В этом случае размер можно не указывать. Базовый образ останется неизменным до тех пор, пока вы его не измените с помощью команды «commit».

Преобразование формата существующего образа

Для преобразования формата используется опция **convert** утилиты qemu-img.

Формат команды:

```
# qemu-img convert [-c] [-e] [-f формат] файл1 [-O полученный_формат] файл2
```

Эта команда преобразует образ диска с именем «файл1» в образ «файл2» в формате «полученный_формат». Полученный образ может быть дополнительно зашифрован (опция «-e») или сжат (опция «-c»).

Следует отметить, что только формат «qcow» поддерживает сжатие и шифрование. Дополнительно, в случае повторной перезаписи сжатого сектора записываемые данные уже не будут сжаты.

Шифрование выполняется в формате AES с использованием 128-разрядных ключей. Для усиления защиты рекомендуется увеличить длину пароля (до 16-ти символов).

Одним из достоинств преобразования образов является возможность получения небольшого образа при использовании формата, допускающего рост (например, **qcow** или **cow**). При этом пустые сектора будут удалены из полученного образа.

Получение информации об образе

Опция **info** утилиты qemu-img позволяет получить сведения о дисковом образе. Формат команды:

```
# qemu-img info [-f формат] имя_файла
```

В результате будут показаны сведения о запрошенном образе, в том числе зарезервированный объем на диске, а также информация о снимках виртуальных машин (если они включены в состав образа).

Поддерживаемые форматы

Формат образа обычно определяется автоматически. Поддерживаются следующие форматы:

raw
: Этот формат используется по умолчанию, его достоинствами являются простота и возможность экспортирования в другие эмуляторы. Если ваша файловая система поддерживает фрагментацию (ext2 или ext3 в Linux, NTFS в Windows), только непосредственно записанные секторы будут занимать место на диске. Действительный объем пространства, занимаемый образом, можно определить с помощью команд **qemu-img info** или **ls -ls** (в Linux).

qcow2
: Формат QEMU. Это наиболее гибкий формат. Его рекомендуется использовать для небольших образов (в частности, если файловая система не поддерживает фрагментацию), дополнительного шифрования AES, сжатия zlib и поддержки множества снимков VM.

qcow
: Старый формат QEMU. Используется только в целях обеспечения совместимости со старыми версиями.

cow
: Формат COW (Copy On Write). Используется только в целях обеспечения совместимости со старыми версиями. Не работает в Windows.

vmdk
: Формат образов, совместимый с VMware 3 и 4.

cloop
: Формат CLOOP (Compressed Loop). Его единственное применение состоит в обеспечении повторного использования сжатых напрямую образов CD-ROM, например, Knoppix CD-ROM.

17.4. Перераспределение ресурсов с помощью KVM

Гипервизор KVM поддерживает перераспределение ресурсов памяти и процессоров. Под этим термином понимается возможность выделения превышающий доступный в системе объем оперативной памяти и число ядер процессора. Так, например, в случае с процессорами, слабо загруженные виртуализированные серверы и рабочие станции могут выполняться на меньшем числе серверов, что позволит снизить финансовые и энергозатраты.

Поддержка Xen

Гипервизор Xen не поддерживает перераспределение процессорных ресурсов. Попытки перераспределения могут нарушить стабильность работы системы и привести к сбою размещающей системы и виртуальных машин.

Перераспределение памяти

Большинство операционных систем и приложений не использует 100% доступной оперативной памяти. Из этого можно извлечь пользу с помощью KVM, разрешив виртуальным машинам использовать больше памяти, чем им физически доступно.

KVM организует работу виртуальных машин в виде процессов Linux. Виртуальным машинам не выделяется физическая память, а поскольку они функционируют как процессы, каждому процессу может быть выделена память по запросу. Так, если гостевая операционная система запрашивает больше памяти, процессу будет выделена память. Виртуальная машина использует лишь незначительно больше физической памяти, чем его виртуализированная операционная система.

Когда физическая память практически занята или процесс неактивен на протяжении какого-то времени, Linux поместит память процесса в область подкачки (swap), которая представляет собой раздел на жестком диске или отдельный диск, используемый для расширения виртуальной памяти. Стоит заметить, что область подкачки работает гораздо медленнее по сравнению с оперативной памятью.

Так как виртуальные машины KVM на самом деле являются процессами Linux, используемая виртуализированными гостями память может быть помещена в область подкачки, если гость неактивен. Общий объем памяти, который можно теоретически перераспределить, не ограничивается общим объемом доступной области подкачки и физической памяти. Это может привести к проблемам, если гостевые системы используют всю выделенную им память. При нехватке пространства подкачки для виртуальных машин их процессы будут вызывать процесс **pdflush**, который освобождает память за счет принудительного прерывания других процессов.

Это достаточно опасно, так как **pdflush** может прервать работу виртуальных машин или других системных процессов, что, в свою очередь, может привести к ошибкам файловой системы и даже повредить виртуализированные гостевые системы так, что их вообще невозможно будет загрузить.

Warning

Если области подкачки не хватает, работа гостевых операционных систем будет принудительно завершена. Поэтому всегда нужно помнить о том, что не стоит перераспределять больше памяти, чем доступно в области подкачки.

Раздел swap используется для размещения не используемой активно памяти на жестком диске с целью повышения производительности. Размер раздела подкачки рассчитывается на основе объема оперативной памяти и коэффициента перераспределения (обычно 0.5, то есть 50%). Рекомендуется его увеличить, если вы намереваетесь использовать возможности перераспределения памяти с KVM. Таким образом, формула расчета будет выглядеть так:

```
(0.5 * ОЗУ) + (коэффициент перераспределения * ОЗУ) = рекомендуемый размер
 области подкачки
```

Дальнейшую информацию можно найти в *Базе знаний*[1].

Другой способ расчета коэффициента состоит в умножении числа виртуальных машин на 10. Но это будет работать только при определенной нагрузке (если виртуализация не использует 100% ресурсов). Значение коэффициента в большинстве случаев подбирается индивидуально в процессе тестирования.

Перераспределение ресурсов виртуализированных процессоров

Гипервизор KVM поддерживает возможности перераспределения виртуализированных процессоров при условии, что это позволяет нагрузка на виртуализированные гостевые системы. Соблюдайте осторожность, так как приближенные к 100% нагрузки могут привести к потере запросов и недопустимо долгому времени ответа.

Процессоры проще всего перераспределять, если каждой виртуальной машине назначен только один виртуальный процессор. В этом случае планировщик Linux будет особенно эффективен. KVM сможет без проблем перераспределять до пяти процессоров виртуальных машин с неполной загрузкой.

Нельзя перераспределять ресурсы симметричных многопроцессорных гостей на ядрах, число которых превышает физическое число процессорных ядер. Так, например, гостевая система с четырьмя VCPU не должна выполняться на узле с процессором с двойным ядром, так как в этом случает значительно пострадает производительность.

Допускается назначение числа виртуальных процессоров гостевой системы, которое не превышает число физических ядер. Например, выполнение виртуализированных гостевых систем с четырьмя VCPU на узле с четырьмя ядрами. В этой конфигурации гостевые системы с неполной нагрузкой будут эффективно функционировать.

[1] http://kbase.redhat.com/faq/docs/DOC-15252

Сначала проверьте

Не пытайтесь перераспределять процессорные ресурсы в производственной среде без предварительного тестирования, так как может нарушиться работа использующих 100% памяти приложений и ресурсов. Всегда тщательно тестируйте, прежде чем приступить к развертыванию.

17.5. Редактирование /etc/grub.conf

В этой секции объясняется, как правильно настроить использование ядра виртуализации в файле **/etc/grub.conf**. Ядро `xen` необходимо для успешной работы гипервизора Xen. Скопируйте всю существующую запись ядра `xen`, чтобы избежать паники системы во время загрузки (т.к. **initrd** будет иметь нулевую длину). При необходимости в строку `xen` в соответствующей записи GRUB добавьте параметры для гипервизора.

Ниже приведен пример записи из файла **grub.conf** системы, где выполняется пакет *kernel-xen*. Обратите внимание на блок текста, начиная со строки **title** и заканчивая следующей пустой строкой.

```
#boot=/dev/sda
default=0
timeout=15
#splashimage=(hd0,0)/grub/splash.xpm.gz hiddenmenu
serial --unit=0 --speed=115200 --word=8 --parity=no --stop=1
terminal --timeout=10 serial console

title Fedora (2.6.23.14-107.fc8xen)
        root (hd0,0)
        kernel /xen.gz-2.6.23.14-107.fc8 com1=115200,8n1
        module /vmlinuz-2.6.23.14-107.fc8xen ro root=/dev/VolGroup00/
LogVol00
        module /initrd-2.6.23.14-107.fc8xen.img
```

Важно

Ваш файл **grub.conf** может отличаться от приведенного примера, если его уже изменяли ранее.

Так, указав `dom0_mem=256M` в строке `xen` файла конфигурации **grub.conf**, вы выделите размещающей системе 256 мегабайт памяти во время загрузки. Файл будет выглядеть так:

```
#boot=/dev/sda
default=0
timeout=15
#splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
serial --unit=0 --speed=115200 --word=8 --parity=no --stop=1
terminal --timeout=10 serial console
```

```
title Fedora (2.6.23.14-107.fc8xen)
        root (hd0,0)
        kernel /xen.gz-2.6.23.14-107.fc8 com1=115200,8n1 dom0_mem=256MB
        module /vmlinuz-2.6.23.14-107.fc8xen ro
        root=/dev/VolGroup00/LogVol00
        module /initrd-2.6.23.14-107.fc8xen.img
```

17.6. Проверка расширений виртуализации

В этой секции будет рассказано, как определить, обладает ли ваша система аппаратными расширениями виртуализации. Расширения виртуализации (Intel VT или AMD-V) потребуются для полной виртуализации.

Можно ли использовать виртуализацию без расширений?

Если нет аппаратных расширений виртуализации, можно организовать паравиртуализацию с помощью пакета *kernel-xen*.

Следующая команда проверит наличие расширений виртуализации:

```
$ grep -E 'svm|vmx' /proc/cpuinfo
```

Если вывод содержит запись `vmx`, это значит, что процессор Intell включает расширения Intel VT. Пример:

```
flags   : fpu tsc msr pae mce cx8 apic mtrr mca cmov pat pse36 clflush
        dts acpi mmx fxsr sse sse2 ss ht  tm syscall lm constant_tsc pni
 monitor ds_cpl
        vmx est tm2 cx16 xtpr lahf_lm
```

Следующий вывод содержит запись `svm`, которая обозначает наличие процессора AMD с расширениями AMD-V:

```
flags   :  fpu tsc msr pae mce cx8 apic mtrr mca cmov pat pse36 clflush
        mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt lm 3dnowext 3dnow
 pni cx16
        lahf_lm cmp_legacy svm cr8legacy ts fid vid ttp tm stc
```

Секция "`flags:`" может повторяться для каждого гиперпотока, ядра и процессора.

Расширения виртуализации можно отключить на уровне BIOS. *Процедура 19.1, «Активация расширений виртуализации в BIOS»* содержит информацию об их активации.

17.7. Определение типа гостевой системы

Приведенный ниже сценарий поможет определить тип системы, в которой он выполняется(паравиртуализированная, полностью виртуализированная или гипервизор).

```
#!/bin/bash
declare -i IS_HVM=0
```

```
declare -i IS_PARA=0
check_hvm()
{
        IS_X86HVM="$(strings /proc/acpi/dsdt | grep int-xen)"
          if [ x"${IS_X86HVM}" != x ]; then
           echo "Guest type is full-virt x86hvm"
           IS_HVM=1
        fi
}
check_para()
{
        if $(grep -q control_d /proc/xen/capabilities); then
          echo "Host is dom0"
          IS_PARA=1
        else
          echo "Guest is para-virt domU"
          IS_PARA=1
        fi
}
if [ -f /proc/acpi/dsdt ]; then
        check_hvm
fi

if [ ${IS_HVM} -eq 0 ]; then
        if [ -f /proc/xen/capabilities ] ; then
                check_para
        fi
     fi
if [ ${IS_HVM} -eq 0 -a ${IS_PARA} -eq 0 ]; then
        echo "Baremetal platform"
fi
```

Проверка размещающих систем

Для проверки размещающих систем используется команда **`virsh capabilites`**.

17.8. Создание уникального MAC-адреса

В некоторых случаях может потребоваться сгенерировать новый уникальный *MAC-адрес* для виртуальной машины. В настоящее время нет утилиты, которая может напрямую это сделать, поэтому ниже приведен специальный сценарий. Сохраните его в гостевой системе как **`macgen.py`**. Для генерации нового MAC-адреса из текущего каталога выполните команду **`./macgen.py`**. Пример:

```
$ ./macgen.py
00:16:3e:20:b0:11

#!/usr/bin/python
# macgen.py script to generate a MAC address for virtualized guests on Xen
```

```
#
import random
#
def randomMAC():
        mac = [ 0x00, 0x16, 0x3e,
                random.randint(0x00, 0x7f),
                random.randint(0x00, 0xff),
                random.randint(0x00, 0xff) ]
        return ':'.join(map(lambda x: "%02x" % x, mac))
#
print randomMAC()
```

Другой способ генерации нового MAC-адреса

Для генерации MAC-адреса и **UUID** также можно использовать встроенные модули **python-virtinst**:

```
# echo  'import virtinst.util ; print\
 virtinst.util.uuidToString(virtinst.util.randomUUID())' | python
# echo  'import virtinst.util ; print virtinst.util.randomMAC()' | python
```

Приведенный выше сценарий можно сохранить в отдельный файл:

```
#!/usr/bin/env python
#  -*- mode: python; -*-
print ""
print "New UUID:"
import virtinst.util ; print
 virtinst.util.uuidToString(virtinst.util.randomUUID())
print "New MAC:"
import virtinst.util ; print virtinst.util.randomMAC()
print ""
```

17.9. Безопасный ftpd

vsftpd обеспечивает доступ к установочному дереву или другим данным для паравиртуализированных гостевых систем. Если вы не установили vsftpd в процессе установки сервера, можно установить соответствующий RPM-пакет из каталога **Server** установочного носителя с помощью команды **rpm -ivh vsftpd*.rpm** (при этом RPM-пакет должен находиться в вашем текущем каталоге).

1. Настройка параметров vsftpd осуществляется в файле **/etc/passwd**. Для его редактирования используйте **vipw** и в качестве домашнего каталога укажите каталог, в котором будет размещено дерево установки для паравиртуализированных гостевых систем. Пример записи FTP-пользователя:

   ```
   ftp:x:14:50:FTP User:/xen/pub:/sbin/nologin
   ```

2. Утилита chkconfig позволяет включить автоматический запуск vsftpd в процессе системной загрузки.

3. Команда **chkconfig --list vsftpd** позволяет проверить, включен ли vsftpd.

```
$ chkconfig --list vsftpd
vsftpd          0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

4. Для автоматического запуска vsftpd на уровнях выполнения 3, 4 и 5 выполните команду **chkconfig --levels 345 vsftpd on**.

5. Проверьте еще раз, запускается ли vsftpd во время загрузки системы:

```
$ chkconfig --list vsftpd
vsftpd          0:off   1:off   2:off   3:on    4:on    5:on    6:off
```

6. Для запуска vsftpd выполните команду **service vsftpd start vsftpd**:

```
$service vsftpd start vsftpd
Starting vsftpd for vsftpd:                  [  OK  ]
```

17.10. Настройка постоянства LUN

В этой секции будет рассмотрено, как обеспечить постоянство *LUN* в гостевых системах и в размещающей системе с использованием многопутевых возможностей и без них.

Обеспечение постоянства LUN без Multipath

Если ваша система не использует многопутевые возможности, сохранение постоянства LUN можно реализовать с помощью **udev**. Но сначала убедитесь в правильности полученных UUID. Затем настройте сохранение постоянства LUN в файле **scsi_id**, который расположен в каталоге **/etc**. Открыв файл в окне текстового редактора, отметьте следующую строку как комментарий:

```
# options=-b
```

Замените на параметр

```
# options=-g
```

Это заставит udev наблюдать за полученными UUID от всех SCSI-устройств. Команда **scsi_id** поможет определить идентификатор UUID:

```
# scsi_id -g -s /block/sdc
*3600a0b8000132751000001542 7b625e*
```

Набор символов в выводе и есть идентификатор UUID. Он не изменяется при добавлении нового устройства в систему. Чтобы иметь возможность создания правил для устройств, получите UUID для каждого устройства. Создать правила можно в файле **20-names.rules** в каталоге **/etc/udev/rules.d** . Формат правил присвоения имени устройству:

```
# KERNEL="sd*",  BUS="scsi",  PROGRAM="sbin/scsi_id", RESULT="UUID",
 NAME="имя_устройства"
```

Замените существующие *UUID* и *имя_устройства* полученными значениями. Правило будет выглядеть примерно так:

```
KERNEL="sd*",  BUS="scsi",  PROGRAM="sbin/scsi_id",
 RESULT="3600a0b80001327510000015427b625e", NAME="мое_имя_устройства"
```

Так, устройства, удовлетворяющие шаблону **/dev/sd***, смогут проверить заданный UUID. Если совпадение найдено, то будет создан узел устройства */dev/имя_устройства*. Наконец, в файл **/etc/rc.local** добавьте строку:

```
/sbin/start_udev
```

Сохранение постоянства LUN с многопутевыми возможностями

Чтобы обеспечить сохранение постоянства LUN в многопутевом окружении, необходимо присвоить псевдонимы многопутевым устройствам. Их можно определить в файле **multipath.conf**, который расположен в каталоге **/etc/**:

```
multipath  {
             wwid       3600a0b80001327510000015427b625e
             alias      oramp1
}
multipath  {
             wwid       3600a0b80001327510000015427b6
             alias      oramp2
}
multipath  {
             wwid       3600a0b80001327510000015427b625e
             alias      oramp3
}
multipath  {
             wwid       3600a0b80001327510000015427b625e
             alias      oramp4
}
```

Здесь всего определено 4 LUN: **/dev/mpath/oramp1**, **/dev/mpath/oramp2**, **/dev/mpath/oramp3**, **dev/mpath/oramp4**. Все устройства будут расположены в каталоге **/dev/mpath**, при этом их имена не будут изменяться между перезагрузками.

17.11. Отключение SMART-мониторинга дисков для гостевых систем

SMART-мониторинг дисков можно отключить, поскольку мы работаем с виртуальными дисками, а физические накопители управляются размещающим узлом.

```
/sbin/service smartd stop
```

```
/sbin/chkconfig --del smartd
```

17.12. Дублирование гостевых файлов конфигурации

При настройке новых гостевых систем можно копировать уже существующие файлы конфигурации. При этом потребуется изменить имя. Новое имя будет отражено гипервизором и управляющими утилитами. Также с помощью **uuidgen** необходимо сгенерировать новый идентификатор UUID. Затем в записях **vif** укажите уникальный MAC-адрес для каждой гостевой системы (при копировании настроек существующего гостя для этой цели можно создать сценарий). При переносе существующего файла конфигурации на новый узел обязательно обновите запись **xenbr** так, чтобы она соответствовала локальным настройкам сетевого окружения. В записях Device укажите корректный образ в строке **'disk='**.

Не забудьте изменить системные настройки в гостевой системе, в частности, запись HOSTNAME в файле **/etc/sysconfig/network** должна содержать имя узла нового гостя.

Измените адрес **HWADDR** в файле **/etc/sysconfig/network-scripts/ifcfg-eth0**, чтобы он соответствовал выводу команды **ifconfig eth0**. Если же вы используете статические IP-адреса, измените запись **IPADDR**.

17.13. Дублирование существующей гостевой системы и файла конфигурации

В этой секции будет рассмотрено создание новой гостевой системы посредством копирования существующего файла конфигурации. Для успешного дублирования гостя потребуется изменить некоторые параметры.

`name`

Уникальное имя гостевой системы, которое используется гипервизором и управляющими утилитами для обращения к ней.

`uuid`

Уникальный идентификатор, который можно сгенерировать с помощью команды **uuidgen**. Пример вывода:

```
$ uuidgen
a984a14f-4191-4d14-868e-329906b211e5
```

`vif`

- *MAC-адрес* уникален для каждой виртуальной машины. Если вы копируете файл конфигурации существующего гостя, попробуйте использовать шаблон сценария (см. *Раздел 17.8, «Создание уникального MAC-адреса»*).
- Если вы решили переместить или скопировать файл конфигурации существующей гостевой системы на новый узел, не забудьте изменить запись `xenbr` так, чтобы она соответствовала локальным настройкам сетевого окружения. Для получения информации о мосте виртуализации выполните команду **brctl show**.
- Измените записи устройств в секции **disk=**, чтобы они указывали на нужный гостевой образ.

Теперь измените системные настройки в гостевой системе:

/etc/sysconfig/network

Измените запись `HOSTNAME`, указав в качестве значения имя узла гостевой системы.

/etc/sysconfig/network-scripts/ifcfg-eth0

- В качестве значения `HWADDR` укажите адрес, полученный в результате выполнения **`ifconfig eth0`**.
- Если используется статический IP-адрес, измените запись `IPADDR`.

Создание специализированных сценариев libvirt

В этой секции приведена информация для программистов и системных администраторов, планирующих создавать собственные сценарии с помощью **libvirt**.

Глава 17, Советы и хитрости содержит дополнительную информацию, с которой рекомендуется ознакомиться, если вы планируете создавать новые приложения, которые будут использовать **libvirt**.

18.1. Использование файлов конфигурации с помощью virsh

virsh может обрабатывать файлы конфигурации XML, что используется при создании специализированных сценариев. Например, в паравиртуализированную гостевую систему можно добавить заданные в XML-файле устройства. Так, для добавления в выполняющуюся гостевую систему ISO-файла как **hdc** создайте файл с таким содержимым:

```
# cat satelliteiso.xml
<disk type="file" device="disk">
        <driver name="file"/>
        <source file="/var/lib/libvirt/images/rhn-satellite-5.0.1-11-
redhat-linux-as-i386-4-embedded-oracle.iso"/>
        <target dev="hdc"/>
        <readonly/>
</disk>
```

Выполните команду **virsh attach-device** для добавления ISO как **hdc** в гостевую систему с именем "satellite" :

```
# virsh attach-device satellite satelliteiso.xml
```

Часть VI. Troubleshooting

Основы диагностики и решения проблем

В последующих главах будут рассмотрены способы решения проблем, с которыми вы можете столкнуться при работе в окружении виртуализации.

Важное замечание

Если ваша конкретная проблема не упомянута в этом документе, обратитесь к *Замечаниям к выпуску* для соответствующей версии и архитектуры. Замечания обычно содержат самую последнюю информацию; их можно найти на сайте Fedora по адресу *http://docs.fedoraproject.org*.

Troubleshooting

В этой главе будут рассмотрены основные проблемы виртуализации Fedora и способы их решения.

19.1. Ошибки петлевого устройства

Если используются файловые образы гостевых систем, может понадобиться увеличить число петлевых устройств. По умолчанию можно настроить до восьми активных устройств, а максимальное число можно изменить в файле **`/etc/modprobe.conf`**. Для этого добавьте строку:

```
options loop max_loop=64
```

В этом примере мы использовали значение 64, но можно указать любое число. Возможно, в системе придется организовать гостевые ОС на основе петлевых устройств, для чего в паравиртуализированных системах используются команды **`phy: block device`** и **`tap:aio`**, а в полностью виртуализированных — **`phy: device`** и **`file: file`**.

19.2. Как включить в BIOS аппаратные расширения виртуализации Intel VT и AMD-V?

Здесь будет рассказано, как правильно определить аппаратные расширения виртуализации и включить их в BIOS.

Расширения Intel VT могут быть отключены в BIOS. Некоторые производители ноутбуков отключают их по умолчанию.

Расширения виртуализации для процессоров AMD-V, установленных в сокете Rev 2, нельзя отключить в BIOS.

Производители ноутбуков иногда отключают расширения виртуализации в BIOS. , как можно активиров*Раздел 19.2, «Как включить в BIOS аппаратные расширения виртуализации Intel VT и AMD-V?»* содержит информацию о том,ать отключенные расширения.

Сначала проверьте, включены ли расширения в BIOS. Настройки BIOS для Intel® VT и AMD-V обычно расположены в меню **Chipset** или **Processor**, но иногда могут быть спрятаны в других меню, например, **Security Settings**.

Процедура 19.1. Активация расширений виртуализации в BIOS

1. Перезагрузите компьютер и войдите в системное меню BIOS (обычно с помощью комбинаций клавиш **`Alt`** + **`F4`** или **Delete**).

2. Восстановите стандартные значения (**Restore Defaults**) и выйдите из BIOS, сохранив изменения (**Save & Exit**).

3. Выключите компьютер и отключите источник питания.

4. Включите компьютер и войдите в BIOS. Перейдите к секции **Processor** и включите **Intel®Virtualization Technology** (в некоторых системах может называться **Virtualization Extensions**) или **AMD-V**. Сохраните изменения, нажав **Save & Exit**.

5. Выключите компьютер и отключите источник питания.

6. Выполните команду **`cat /proc/cpuinfo | grep vmx svm`**. Если вывод команды пуст, это может означать наличие ошибок в настройках BIOS или отсутствие в системе расширений. Непустой вывод команды будет свидетельстовать о том, что расширения виртуализации включены.

Приложение A. Дополнительные ресурсы

Перечисленные ниже ресурсы предоставляют дальнейшую информацию о виртуализации и Linux.

A.1. Интернет-ресурсы

- *http://www.cl.cam.ac.uk/research/srg/netos/xen/* — сайт проекта Xen™, на основе которого был создан пакет *kernel-xen*. Кроме прочей информации, там можно найти двоичные файлы, исходные коды, а также полезные сведения, обзоры архитектур, документацию и пр.
- Сайт сообщества Xen

  http://www.xen.org/
- *http://www.libvirt.org/* — официальный сайт API виртуализации **`libvirt`**.
- *http://virt-manager.et.redhat.com/* — сайт графического приложения менеджера виртуальных машин (virt-manager).
- Центр открытой виртуализации

  http://www.openvirtualization.com[1]
- Документация Fedora

  http://docs.fedoraproject.org
- Обзор технологий виртуализации

  http://virt.kernelnewbies.org[2]
- Группа развивающихся технологий Red Hat

  http://et.redhat.com[3]

A.2. Установленная документация

- **`/usr/share/doc/xen-<версия>/`** — содержит большой объем информации о гипервизоре Xen и соответствующих управляющих инструментах, примеры конфигурации, информацию об оборудовании и последнюю документацию Xen.
- **`man virsh`** и **`/usr/share/doc/libvirt-<версия>`**— содержат списки команд и параметров **`virsh`**, а также подробную информацию о API библиотеки виртуализации **`libvirt`**.
- **`/usr/share/doc/gnome-applet-vm-<версия>`** — содержит описание апплета GNOME, используемого для наблюдения за локальными виртуальными машинами и их управления. На английском.

- **`/usr/share/doc/libvirt-python-<версия>`** — в этом файле приведены соответствия Python для библиотеки **`libvirt`**. Пакет **`libvirt-python`** позволяет разработчикам Python создавать программы, взаимодействующие с библиотекой **`libvirt`**. На английском.
- **`/usr/share/doc/python-virtinst-<версия>`** — содержит описание команды **`virt-install`**, которая используется при установке Fedora и других дистрибутивов Linux на виртуальных машинах. На английском.
- **`/usr/share/doc/virt-manager-<версия>`** — документация по работе с менеджером виртуальных машин. На английском.

Приложение B. История изменений

Издание 12.1.3	Mon Oct 12 2009	Кристофер Каррен ccurran@redhat.com

На основе версии 5.4-61 руководства по виртуализации Red Hat Enterprise Linux 5

Приложение C. Издание

Это руководство создано в формате DocBook XML 4.3.

Авторы документа: Ян Марк Холзер, Кристофер Каррен.

Благодарности:

- Техническая редакция секции паравиртуализированных драйверов: Дон Дьютил
- Техническая редакция секции паравиртуализированных драйверов: Барри Донахью
- Техническая редакция секции описания менеджера виртуальных машин: Рик Ринг
- Техническая редакция секций описания использования XML-файлов конфигурации с virsh и виртуализированными съемными носителями: Майкл Кери
- Техническая редакция секции описания производительности и программной совместимост: Марко Григал
- Техническая редакция секции описания управления гостевыми системами с помощью virsh: Юджин Тео

Для публикации этой книги использовался инструмент Publican. Автор: Джеффри Ферн

Команда локализации Red Hat:

Восточно-азиатские языки

- Упрощенный китайский
  - Ли Ви Лиу
- Традиционный китайский
  - Честер Ченг
  - Терри Чанг
- Японский
  - Джунко Ито
- Корейский
  - Ун-Джу Ким

Языки латинской и славянской группы

- Французский
  - Сэм Фридман
- Немецкий
  - Хедда Питерс

- Итальянский
  - Франческо Валенте
- Португальский (Бразилия)
  - Глаусия Де Фрейтес
  - Летисия Де Лима
- Испанский
  - Анжела Гарсиа
  - Глэдис Герреро
- Русский
  - Юлия Пояркова

Глоссарий

Глоссарий содержит список терминов, которые вы встретите в этом руководстве.

Базовое оборудование, «bare metal»	Этот термин относится к физической архитектуре компьютера. Примером выполнения операционной системы на базовом оборудовании может служить *dom0* или установленная обычным способом ОС.
Домен 0 (dom0)	Также известна как *Размещающая система, узел*. **`dom0`** представляет собой экземпляр узла Linux, в котором выполняется *Гипервизор*. Dom0 отвечает за выделение оборудования и ресурсов самому себе и гостевым операционным системам.
Домены	*domU* и *Домены* выполняются на гипервизоре (см. *Гипервизор*). Домены в этом смысле подобны виртуальным машинам (см. *Виртуальные машины*), эти два термина взаимозаменяемы. Так, домен является виртуальной машиной.
domU	**`domU`** — гостевая операционная система, выполняющаяся в размещающей системе (см. *Домены*).
Полная виртуализация	Xen и KVM используют полную виртуализацию. Полная виртуализация создает уровень абстракции, независимый от физической системы (см. *Базовое оборудование, «bare metal»*) для реализации новой виртуальной системы, в которой будут выполняться гостевые системы (они не требуют модификации). Гостевые ОС и их приложения не будут «знать» о виртуализированном окружении, в котором они исполняются, и будут работать как обычно. Для организации же паравиртуализации потребуется модифицированная версия ОС Linux.
Полностью виртуализированная	См. *Полная виртуализация*.
Гостевая система	Также называются гостями, виртуальными машинами или доменами (см. *domU*).
HVM	См. *Полная виртуализация*
Гипервизор	Гипервизор представляет собой программный слой абстракции, отделяющий оборудование от операционной системы, что позволяет выполнять несколько ОС одновременно. Гипервизор исполняется в размещающей системе и отвечает за выполнение виртуализированных операционных систем.
Размещающая система, узел	Размещающая операционная система (также см. *Домен 0 (dom0)*). В размещающей системе выполняются программы виртуализации для полностью виртуализированных (см.

Полностью виртуализированная) и паравиртуализированных (см. *Паравиртуализированная*) гостевых систем.

I/O

Аббревиатура для ввода и вывода. Этот термин используется для описания программ, действий или устройств, передающих данные из системы периферийным устройствам и обратно. Каждое действие передачи будет представлять собой вывод для одного устройства и ввод для другого. Клавиатура и мышь являются устройствами ввода, в то время как принтеры являются устройствами вывода. А например, записывающий CD-ROM — устройство ввода и вывода одновременно.

KVM

KVM (Kernel-based Virtual Machine) — модуль полной виртуализации ядра (см. *Полная виртуализация*) для Linux на платформах AMD64 и Intel 64. KVM способен исполнять несколько виртуальных машин Windows или Linux без необходимости модификации операционной системы. KVM предоставляет собой гипервизор, использующий средства виртуализации libvirt (virt-manager и virsh).

KVM представляет собой набор модулей ядра Linux, которые управляют устройствами, памятью и управляющими API собственно модуля гипервизора. Виртуальные машины выполняются как процесы и потоки Linux, которыми эти модули управляют.

LUN

LUN (Logical Unit Number) — номер, назначенный логическому компоненту. Такой метод адресации используется протоколом SCSI.

Миграция

Процесс переноса виртуализированной гостевой системы с одного узла на другой. Миграция может быть выполнена в автономном режиме (работа гостевой системы останавливается перед переносом) или подключенном режиме (гостевая система продолжает работу во время переноса). Можно выполнять миграцию полностью виртуализированных, паравиртуализированных гостевых систем Xen и полностью виртуализированных гостей KVM.

Миграция является основополагающим аспектом виртуализации, так как на этом уровне программное обеспечение совершенно не зависит от оборудования. Основное назначение миграции:

- Load balancing - guests can be moved to hosts with lower usage when a host becomes overloaded.
- Hardware failover - when hardware devices on the host start to fail, guests can be safely relocated so the host can be powered down and repaired.
- Energy saving - guests can be redistributed to other hosts and host systems powered off to save energy and cut costs in low usage periods.

- Geographic migration - guests can be moved to another location for lower latency or in serious circumstances.

Для хранения гостевых образов используется общее хранилище. Без этого миграция будет невозможна.

An offline migration suspends the guest then moves an image of the guests memory to the destination host. The guest is resumed on the destination host and the memory the guest used on the source host is freed.

Длительность офлайн-миграции зависит от полосы пропускания и сетевой задержки. Так, перенос гостевой системы с 2 Гбайт памяти по 1 гигабит Ethernet займет несколько секунд.

Живая миграция характеризуется тем, что работа виртуальных машин не останавливается при переносе. Все изменяемые за это время страницы памяти отслеживаются и передаются целевому узлу после завершения передачи образа. Процесс продолжается до тех пор, пока не будут скопированы все страницы или пока не станет ясно, что за изменениями источника не успеть. Если страницы источника изменяются слишком быстро, то работа гостя на исходном узле будет приостановлена и будет выполнена передача регистров и буферов. Регистры будут загружены на новом узле и гость возобновит работу на целевом узле. Если же синхронизация невозможна, что вероятно в случае большой нагрузки, то виртуальная машина будет приостановлена для выполнения миграции в автономном режиме.

Длительность такой миграции зависит от полосы пропускания, сетевой задержки и активности гостевой системы. Нагрузка на процессор и большие объемы операций ввода и вывода также могут сказаться на длительности процесса.

MAC-адрес

MAC-адрес (Media Access Control) — аппаратный адрес контроллера сетевого интерфейса. Для виртуальных сетевых интерфейсов MAC-адреса должны быть сгенерированы, при этом адреса в локальном домене должны быть уникальны.

Паравиртуализация

Паравиртуализация использует специальное ядро, иногда называемое ядром Xen или *kernel-xen*, позволяющее использовать библиотеки и устройства размещающей системы. Паравиртуализированная установка будет иметь доступ ко всем устройствам в системе. Это можно ограничить с помощью SELinux. Паравиртуализация работает гораздо быстрее по сравнению полной виртуализацией и может эффективно использоваться для распределения нагрузки, поддержки работы и обеспечения безопасности.

Начиная с Fedora 9, нет необходимости в специальном ядре. После включения этой функциональности в основную версию

	Linux все последующие выпуски ядер Linux будут включать возможность виртуализации.
Паравиртуализированная	Также смотрите *Паравиртуализация*.
Паравиртуализированный драйвер	Драйвер устройства, работающий в полностью виртуализированных гостевых системах Linux. Такие драйвера значительно улучшают производительность сети и операций I/O для полностью виртуализированных гостей.
SELinux	SELinux (Security Enhanced Linux) использует специальные модули защиты Linux в ядре для создания набора минимальных привилегий, необходимых политике безопасности.
UUID	UUID (Universally Unique Identifier) стандартизирует нумерацию устройств, систем и программных компонентов в распределенных вычислительных окружениях. Типы UUID включают идентификаторы файловых систем `ext2` и `ext3`, идентификаторы устройств RAID, iSCSI и LUN, MAC-адреса и идентификаторы виртуальных машин.
Virtualization	Под виртуализацией понимается одновременное выполнение программного обеспечения, обычно операционных систем, в одной системе, но изолированно от других программ. В большинстве случаев гипервизор обеспечивает уровень абстракции, необходимый для работы нескольких операционных систем на одном компьютере. Способы виртуализации операционных систем включают: • Аппаратная виртуализация используется при полной виртуализации с Xen и KVM (см. *Полная виртуализация*). • Паравиртуализация используется Xen для организации работы гостевых систем Linux (см. *Паравиртуализация*). • Программная виртуализация (эмуляция) обеспечивает работу исходных операционных систем за счет двоичных преобразований и других способов эмуляции. Такой метод значительно медленнее по сравнению с аппаратной виртуализацией или паравиртуализацией.
Виртуальный процессор, VCPU	Число виртуальных процессоров в системе определяется числом ядер физического процессора. Эти процессоры могут быть предоставлены гостевыми виртуальным машинам.
Виртуальные машины	Виртуальная машина представляет собой программную реализацию физической машины или языка программирования (например, Java Runtime Environment или LISP). Виртуальные машины в контексте виртуализации — операционные системы, выполняемые на виртуализированном оборудовании.
Xen	Fedora поддерживает гипервизоры Xen и KVM (см. *KVM*). Эти гипервизоры отличаются по архитектуре и подходами в их разработке. Так, Xen работает на низком уровне, ниже операционной системы Linux (также известной как узел или

Домен 0 (dom0)). Домен 0 отвечает за управление системными ресурсами и API виртуализации.