Федеральное государственное унитарное предприятие
Российский федеральный ядерный центр
Всероссийский научно-исследовательский институт экспериментальной физики

УТВЕРЖДЁН

07623615.00439-10 92 01-ЛУ

КОМПЛЕКС ПРОГРАММ В ЗАЩИЩЁННОМ ИСПОЛНЕНИИ
«СИСТЕМА ПОЛНОГО ЖИЗНЕННОГО ЦИКЛА ИЗДЕЛИЙ
«ЦИФРОВОЕ ПРЕДПРИЯТИЕ»

Программный модуль «Технология «тонкого клиента» «Синергия-ТК»

Руководство администратора

07623615.00439-10 92 01

Листов 167

2022

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
35-18632				

АННОТАЦИЯ

Настоящий документ содержит сведения о назначении программного продукта «Программный модуль «Технология «тонкого клиента» «Синергия-ТК», его структуре и порядке функционирования.

Документ представляет собой руководство для администраторов — пользователей программного продукта, наделённых особыми полномочиями и привилегиями. Руководство содержит инструкции по настройке компонентов продукта и «гостевых» виртуальных машин.

СОДЕРЖАНИЕ

1. НАЗНАЧЕНИЕ ПРОГРАММЫ

Программный модуль «Технология «тонкого клиента» «Синергия-ТК»» представляет собой менеджер виртуальных машин (гипервизор) первого типа, выполняется непосредственно на аппаратуре средств вычислительной техники (СВТ) и не требует какого-либо дополнительного программного обеспечения для своего запуска и функционирования, за исключением входящей в состав аппаратуры СВТ прошивки базовой системы ввода-вывода.

Программный модуль «Технология «тонкого клиента» «Синергия-ТК» предназначен для использования в автоматизированных системах, в том числе в защищённом исполнении класса «1Б» включительно, построенных с применением технологий виртуализации и паравиртуализации.

Основными функциями программного модуля «Технология «тонкого клиента» «Синергия-ТК» являются управление и разделение ресурсов физической системы для обеспечения возможности запуска нескольких виртуальных машин в совокупности с операционными системами (далее - ОС) обеспечивающими взаимодействие с внешними системами и информационной инфраструктурой. Программный модуль «Технология «тонкого клиента» «Синергия-ТК» может использоваться как основа для создания информационных систем более высокого уровня.

Программный модуль «Технология «тонкого клиента» «Синергия-ТК» представляет собой систему низкоуровневого управления и обеспечения выполнения виртуальных машин. Для создания конечных решений на его основе необходимы соответствующие задаче операционные системы и прикладное ПО. Программный модуль «Технология «тонкого клиента» «Синергия-ТК» рассчитан на профессиональных специалистов в сфере информационных технологий и инженеров для использования в качестве основы при создании информационных и автоматизированных систем.

Программный модуль «Технология «тонкого клиента» «Синергия-ТК» состоит из следующих функциональных модулей:

- функциональный модуль управления виртуальными машинами;

- функциональный модуль «Комплекс средств управления»;
- клиентская часть;
- функциональный модуль «Комплекс средств защиты».

Функциональный модуль управления виртуальными машинами является гипервизором первого типа, выполняющимся непосредственно на аппаратуре СВТ и не требующим дополнительной среды для своего функционирования.

Функциональный модуль управления виртуальными машинами обеспечивает низкоуровневое управление, изоляцию и контролируемое выполнение виртуальных машин.

Функциональный модуль управления виртуальными машинами предназначен для разделения ресурсов физической системы на изолированные виртуальные машины, предоставляющие возможность контролируемого запуска сторонних операционных систем, управления ими, обеспечения их взаимодействия с внешними системами и информационной инфраструктурой.

Функциональный модуль управления виртуальными машинами применяется для создания виртуальных сред функционирования, выполнения ОС в виртуальных средах в режиме изоляции и обеспечения дискреционного, ролевого и мандатного контроля доступа к данным и потокам информации.

Функциональный модуль «Комплекс средств управления» выполняется в составе программного модуля «Технология «тонкого клиента» «Синергия-ТК» под управлением функционального модуля управления виртуальными машинами и не требует дополнительного программного или аппаратного обеспечения для своего функционирования. Взаимодействие пользователя с функциональным модулем «Комплекс средств управления» осуществляется с помощью WEB-браузера на автоматизированном рабочем месте (далее - АРМ) администратора.

Функциональный модуль «Комплекс средств управления» представляет собой систему централизованного управления виртуальной инфраструктурой, реализуемой с использованием программного модуля «Технология «тонкого клиента» «Синергия-ТК».

Функциональный модуль «Комплекс средств управления» предназначен для предоставления доступа администратору информационной инфраструктуры к управлению функциональным модулем управления виртуальными машинами и функциональным модулем «Комплекс средств защиты» посредством графического интерфейса пользователя, реализованного как WEB-приложение.

Функциональный модуль «Комплекс средств управления» применяется для снижения порога сложности процесса администрирования, уменьшения влияния человеческого фактора и автоматизации рутинных действий по управлению одним или более СВТ, работающих под управлением функционального модуля управления виртуальными машинами, в составе информационной системы, созданной на базе программного модуля «Технология «тонкого клиента» «Синергия-ТК».

Клиентская часть программного модуля «Технология «тонкого клиента» «Синергия-ТК» — это прикладная программа, запускаемая на терминалах (физических автоматизированных рабочих местах).

Функциональный модуль «Комплекс средств защиты» является неотъемлемой частью программного модуля «Технология «тонкого клиента» «Синергия-ТК», интегрируемой в функциональность модулей управления виртуальными машинами и «Комплекс средств управления».

Функциональный модуль «Комплекс средств защиты» предназначен для реализации механизмов защиты информации, а также ряда дополнительных функций, связанных с централизованной идентификацией, аутентификацией и управлением политиками безопасности.

2. ТРЕБОВАНИЯ ПО ЭКСПЛУАТАЦИИ

2.1. Требования к аппаратному обеспечению

Для успешного применения серверной части программного модуля «Технология «тонкого клиента» «Синергия-ТК» необходимо использовать оборудование на основе процессоров с архитектурой x86_64 классом не ниже Core i3/i5/i7 производства Intel/AMD с технологией VT.

Для работы виртуальных машин в режиме полной виртуализации, обеспечивающего максимальный уровень изоляции и производительности, процессор должен поддерживать расширенный набор инструкций для аппаратного ускорения механизмов виртуализации: Intel VT/d или AMD IOMMU.

Для надёжной и безопасной работы необходимо установить все обновления BIOS, микропрограмм, и т.д., рекомендуемые производителем оборудования, на котором производится работа с программным модулем «Синергия-ТК».

2.2. Требования к программному обеспечению

Установка программного модуля «Технология «тонкого клиента» «Синергия-ТК» производится в среде ОС Astra Linux Special Edition «Смоленск» версии 1.7 или выше.

Для работы ОС и прикладного ПО в среде виртуализации (вАРМ), предоставляемой серверной частью программного модуля «Технология «тонкого клиента» «Синергия-ТК», необходимо:

- обеспечить работу менеджера виртуальных машин и управляющего домена программного модуля программного модуля «Технология «тонкого клиента» «Синергия-ТК»;
- в виртуальных машинах использовать ОС, поддерживающие эмулируемое оборудование — как минимум, чипсеты Intel i440 и Q35;
- для обеспечения максимальной производительности целевой ОС вАРМ использовать паравиртуальные драйверы ввода-вывода, совместимые с гипервызовами XEN (Включены в большинство ядер UNIX/*BSD/Linux, для MS Windows необходимо устанавливать отдельно).

2.3. Требования к персоналу

Программный модуль «Технология «тонкого клиента»«Синергия-ТК» является сложным техническим решением, для полноценного использования всех возможностей которого необходимо хорошее знание Администратором принципов работы вычислительной техники и сетевого оборудования, наличие навыков администрирования ОС семейства UNIX/Linux в информационной среде предприятия, знание стандартов RFC и понимание принципов работы сетевых протоколов, используемых в корпоративных информационных системах.

3. СБОРКА ПРОГРАММЫ

Исходный код программы представляет собой две группы проектов: проекты для сборки дистрибутива менеджера виртуальных машин, проекты для сборки дистрибутива клиентской части и ПО Терминала.

Структура исходного кода приведена в таблице 1.

Таблица 1 - Структура исходного кода программы

Каталог исходного кода	Описание
host/	**Исходный код ФМ УВМ**
host/build-tools/	Средства сборки исходного кода
host/spice/	Реализация Spice-сервера
host/xen/	Гипервизор Xen
host/ovs/	Виртуальный коммутатор OpenvSwitch
host/host_rootfs_overlay/	Средства автоматизации и конфигурация хоста
host/websockify/	Web-socket сервер Websockify
ksu/	**Исходный код ФМ КСУ**
ksu/build-tools/	Средства сборки исходного кода
ksu/dependenceis/	Зависимости сборки КСУ
ksu/hv-manager/	Приложение КСУ
ksu/ksu_rootfs_overlay/	Средства автоматизации и конфигурация КСУ
terminal/	**Исходный код ПО Терминала**
terminal/build-tools/	Средства сборки исходного кода
terminal/dependencies/	Зависимости ПО Терминала
terminal/tk-terminal/	Исходный код ПО Терминала

Для сборки программы необходимо определить отдельный каталог для сборки дистрибутива. В дальнейшем он будет использоваться в качестве параметра при вызове скриптов сборки.

Сборка производится в среде ОС Astra Linux Special Edition 1.7 «Смоленск». Для настройки среды сборки необходимо перед началом сборки изделия подключить к машине диск с дистрибутивом ОС, диски разработчика №1 и №2.

У пользователя, осуществляющего сборку, должны быть права осуществлять операции от имени суперпользователя (sudo-пользователь).

Требования к среде сборки приведены в таблице 2.

Таблица 2 - Структура исходного кода программы

Компонент	Требования
Процессор	С архитектурой x86_64 классом не ниже Core i3/i5/i7 производства Intel/AMD с технологией VT
ОЗУ	Не менее 6ГБ
HDD	Не менее 15ГБ
ОС	Astra Linux Special Edition 1.7 «Смоленск»

3.1. Сборка менеджера виртуальных машин

Для сборки менеджера виртуальных машин необходимо в каталоге с распакованным исходным кодом перейти в подкаталог build-tools. Подготовка к сборке и сборка ФМ УВМ осуществляется с помощью находящихся в данном каталоге скриптов автоматизации сборки.

Для подготовки среды сборки необходимо выполнить скрипт:

sudo ./set-env.sh

При выполнении скрипта на машину сборки будут установлены необходимые пакеты из состава ОС, а также из каталога packages/. Скрипт выполняется от имени суперпользователя.

После успешной установки всех необходимых пакетов нужно собрать и установить на машине сборки реализацию Spice-сервера:

sudo ./build-spice.sh

Для установки собранных компонентов Spice-сервера необходимы права суперпользователя, поэтому скрипт запускается в режиме sudo.

После того как необходимые компоненты установлены, можно запускать скрипт сборки менеджера виртуальных машин:

./build.sh <каталог дистрибутива>/host

При запуске скрипта указывается подкаталог host каталога для сборки дистрибутива. В случае если каталог существует, он будет удален и снова создан автоматически.

После завершения сборки дистрибутив менеджера виртуальных машин будет доступен в указанном каталоге.

3.2. Сборка комплекса средств управления

Для сборки комплекса средств управления необходимо в каталоге с распакованным исходным кодом перейти в подкаталог build-tools. Подготовка к сборке и сборка комплекса средств управления осуществляется с помощью находящихся в данном каталоге скриптов автоматизации сборки.

Для установки инструментов сборки и переменных окружения необходимо выолнить скрипт:

```
source set-environment.sh
```

Скрипт устанавливает средства управления пакетами npm в составе программного пакета NodeJS в каталог /opt/tools/. Для выполнения скрипта у пользователя должны быть права rwxd на каталог /opt.

Далее необходимо установить пакеты необходимые для сборки комплекса средств управления:

```
sudo ./install-packages.sh
```

Скрипт устанавливает пакеты ОС и пакеты python3 из состава ОС. Для выполнения скрипта требуются права суперпользователя, поэтому его необходимо запускать в режиме sudo.

После установки пакетов необходимо произвести сборку зависимостей приложения КСУ:

```
./build-dependencies.sh <каталог дистрибутива>/ksu/dependencies
```

Для сборки проекта требуется установить собранные зависимости на машину сборки. Установка требует прав суперпользователя и производится в режиме sudo:

```
sudo ./install-dependencies-src.sh
```

После того как необходимые пакеты установлены, можно производить сборку:

```
./rebuild-ksu.sh <каталог дистрибутива>/ksu
```

После завершения сборки дистрибутив комплекса средств управления будет доступен в указанном каталоге.

3.3. Сборка ПО Терминала

Перед выполнением сборки ПО Терминала необходимо выполнить установку или убедиться в наличии следующих библиотек из состава Astra Linux:

- libgtk-3.so.0;
- libgdk-3.so.0;
- libpangocairo-1.0.so.0;
- libpango-1.0.so.0;
- libatk-1.0.so.0;
- libcairo-gobject.so.2;
- libcairo.so.2;
- libgdk_pixbuf-2.0.so.0;
- libgio-2.0.so.0;
- libgobject-2.0.so.0;
- libglib-2.0.so.0;
- libstdc++.so.6;
- libm.so.6;
- libgcc_s.so.1;
- libpthread.so.0;
- libc.so.6;
- libatk-bridge-2.0.so.0;
- libxkcommon.so.0;
- libwayland-cursor.so.0;
- libwayland-egl.so.1;
- libatspi.so.0.

Для сборки ПО Терминала необходимо в каталоге с распакованным исходным кодом перейти в подкаталог build-tools. Подготовка к сборке и сборка ПО Терминала осуществляется с помощью находящихся в данном каталоге скриптов автоматизации сборки.

Для сборки библиотеки Poco необходимо выполнить скрипт:

./build-poco.sh <каталог дистрибутива>/terminal/

После сборки необходимо установить библиотеку Poco на машину сборки:

sudo ./install-poco.sh <каталог дистрибутива>/terminal/

Скрипт устанавливает библиотеки и заголовочные файлы в каталог /usr/local/.

После сборки зависимостей необходимо произвести сборку ПО Терминала:

./build-terminal.sh <каталог дистрибутива>/terminal/

После завершения сборки результирующие артефакты будут находиться в указанном при сборке каталоге дистрибутива.

4. УСТАНОВКА ПРОГРАММЫ

Дистрибутив программного модуля «Технология «тонкого клиента» «Синергия-ТК» включает три основных компонента:

- дистрибутив менеджера виртуальных машин;
- дистрибутив комплекса средств управления;
- ПО Терминала.

Программный модуль «Технология «тонкого клиента» «Синергия-ТК» поставляется на DVD-диске либо на специально подготовленном USB Flash накопителе.

Дистрибутив программы представляет собой систему каталогов с компонентами функциональных модулей, которая представлена в таблице 3.

Таблица 3 - Структура дистрибутива программы

Каталог исходного кода	Описание
host/	**Исходный код менеджера виртуальных машин**
host/xen/	Гипервизор Xen
host/ovs/	Виртуальный коммутатор OpenvSwitch
host/host_rootfs_overlay.tar.gz	Средства автоматизации и конфигурация хоста
host/websockify/	Web-socket сервер Websockify
ksu/	**Исходный код комплекса средств управления**
ksu/dependenceis/	Зависимости сборки комплекса средств управления
ksu/hv-manager.tar.gz	Приложение комплекса средств управления
terminal/	**Исходный код ПО Терминала**
terminal/poco.tar.gz	Библиотеки Poco
terminal/tk_connector	Приложение ПО Терминала

Установка производится на физическую или виртуальную машину под управлением ОС Astra Linux Special Edition 1.7 «Смоленск».

4.1. Установка менеджера виртуальных машин

Для установки менеджера виртуальных машин на виртуальную машину необходимо убедиться, что в среде виртуализации установлены параметры виртуальной машины, допускающие вложенную виртуализацию:

- в случае использования виртуализации на базе VMWare в свойствах ВМ хоста необходимо установить признак для CPU «Expose hardware assisted virtualization to the guest OS»;
- в случае использования виртуализации на базе Xen в конфигурационном файле виртуальной машины необходимо установить признак «nestedhvm=1».

Для установки менеджера виртуальных машин необходимо выполнить следующий скрипт от имени суперпользователя из каталога дистрибутива менеджера виртуальных машин (подкаталог host):

sudo ./install.sh <каталог дистрибутива>/host

4.1.1. Настройка Apache HTTP Server

Для выполнения команд комплекса средств управления необходимо предоставить WEB-серверу менеджера виртуальных машин необходимые привилегии.

Для этого нужно добавить пользователя www-data в группу sudo:

echo “www-data ALL=(ALL) NOPASSWD:ALL” >> /etc/sudoers

Предоставить пользователю www-data необходимые привилегии:

usercaps -f www-data

Отключить режим «Astra Mode» путем изменения параметра в файле /etc/apache2/apache2.conf:

AstraMode on

AstraMode off

4.1.1.1. Настройка конфигурации HTTP-сервера для локальной аутентификации

Для управления серверным приложением менеджера виртуальных машин необходимо включить модули WEB-сервера следующим образом:

sudo a2enmod authnz_pam

```
sudo a2enmod cgi
```

Отключить стандартный сайт следующей командой:

```
sudo a2dissite 000-default.conf
```

Необходимо создать новый конфигурационный файл сайта и включить его:

```
sudo touch /etc/apache2/sites-available/host.conf
sudo a2ensite host.conf
```

Добавить в конфигурационный файл сайта /etc/apache2/sites-available/host.conf следующее содержимое:

```
<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
        ServerName host.domain.loc

        ScriptAlias /cgi /var/www/html
        <Directory /var/www/html>
           AddHandler cgi-script .cgi
           Options +ExecCGI +Indexes

           AuthType Basic
                AuthName "PAM authentication"
                AuthBasicProvider PAM
                AuthPAMService apache2
                Require valid-user
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

«host.domain.loc» необходимо заменить на имя сервера.

После завершения настройки необходимо перезапустить сервис HTTP-сервера следующей командой:

```
sudo systemctl restart apache2
```

4.1.1.2. Настройка конфигурации HTTP-сервера для аутентификации по протоколу Kerberos

Для выполнения настроек необходимо пройти аутентификацию учётной записью администратора домена:

```
sudo kinit admin
```

Создать принципала службы HTTP-сервера (на примере имени сервера HTTP «host.domain.loc»):

```
sudo ipa service-add HTTP/host.domain.loc@DOMAIN.LOC
```

После прохождения аутентификации необходимо получить таблицу ключей для HTTP-сервера (на примере имени контроллера домена «freeipa.domain.loc»):

```
sudo ipa-getkeytab -s freeipa.domain.loc -k /etc/apache2/http.keytab -p HTTP/host.domain.loc@DOMAIN.LOC
```

Для управления серверным приложением менеджера виртуальных машин необходимо включить модуль авторизации следующим образом:

```
sudo a2enmod auth_kerb
sudo a2enmod cgi
```

Отключить стандартный сайт следующей командой:

```
sudo a2dissite 000-default.conf
```

Необходимо создать новый конфигурационный файл сайта и включить его:

```
sudo touch /etc/apache2/sites-available/host.conf
sudo a2ensite host.conf
```

Установить файлу ключей необходимые атрибуты:

```
sudo chown www-data:www-data /etc/apache2/http.keytab
sudo chmod 600 /etc/apache2/http.keytab
```

Добавить в файл /etc/apache2/sites-avialable/host.conf следующее содержимое:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ServerName host.domain.loc

    ScriptAlias /cgi /var/www/html
    <Directory /var/www/html>
        AddHandler cgi-script .cgi
```

```
        Options +ExecCGI +Indexes

        AuthType Kerberos
        KrbAuthRealms DOMAIN.LOC
        KrbServiceName HTTP/host.domain.loc@DOMAIN.LOC
        Krb5Keytab /etc/apache2/http.keytab
        KrbMethodNegotiate on
        KrbMethodK5Passwd on
        KrbSaveCredentials on
        require valid-user
    </Directory>


    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

«host.domain.loc» необходимо заменить на имя сервера.

После завершения настройки необходимо перезапустить сервис HTTP-сервера следующей командой:

```
sudo systemctl restart apache2
```

4.2. Установка комплекса средств управления

Для установки комплекса средств управления необходимо развернуть приложение комплекса средств управления под управлением HTTP-сервера Apache.

Необходимо установить пакеты зависимостей основного приложения комплекса средств управления:

```
sudo ./install-dependencies-ksu.sh
```

Необходимо добавить пользователей системы/домена: потребуется пользователь который будет исполнять функции администратора (для примера пользователь с именем «administrator»), отдельный пользователь для планировщика (для примера пользователь с именем «ksu_shed»), а также остальные пользователи системы в зависимости от потребностей.

Пользователи добавляются средствами ОС либо контроллера домена. После добавления рекомендуется зайти в ОС штатными средствами, чтобы в системе создались необходимые пользовательские объекты.

Пользователь комплекса средств управления administrator добавляется во время установки комплекса средств управления, остальные пользователи добавляются через механизмы управления пользователями комплекса средств управления.

4.2.1. Настройка СУБД PostgreSQL

Необходимо предоставить доступ участникам группы к папке с логами:

```
sudo chmod 0750 /var/lib/postgresql/11/main/
sudo chmod 0750 /var/lib/postgresql/11/main/pg_log/
```

Создать базу данных для средств управления:

```
sudo -u postgres psql -c 'create database "hv_manager";'
```

Для управления ролями СУБД нужно добавить административного пользователя и выдать ему соответствующие права:

```
sudo -u postgres psql hv_manager -c 'create user "administrator";'
sudo -u postgres psql hv_manager -c 'alter user "administrator" createrole;'
sudo -u postgres psql hv_manager -c 'grant pg_read_server_files to administrator;'
```

По окончанию работ (после проведения миграций) необходимо выдать администратору права на созданные объекты базы данных:

```
sudo -u postgres psql -c 'grant all on database hv_manager to "administrator" with grant option;'
sudo -u postgres psql hv_manager -c 'grant all on all tables in schema public to "administrator" with grant option;'
sudo -u postgres psql hv_manager -c 'grant all on all sequences in schema public to "administrator" with grant option;'
```

4.2.1.1. Настройка СУБД для локальной аутентификации пользователей

В случае, если для аутентификации в комплексе средств управления будут использоваться аутентификационные данные локальных пользователей ОС сервера комплекса средств управления, рекомендуется ограничить доступ для внешних пользователей путем удаления или комментирования строк в файле /etc/postgresql/11/main/pg_hba.conf следующим образом:

```
#host    all     all     0.0.0.0/0     md5
#host    all     all     ::1/128       md5
```

При выполнении данной настройки подключение к СУБД будет возможно только от локальных процессов сервера СУБД, в данной конфигурации серверное приложение комплекса средств управления и СУБД должны устанавливаться на одну машину.

При использовании механизма мандатного разграничения доступа необходимо выдать пользователю postgres разрешения подсистемы parsec следующей командой:

```
sudo ./set-parsec.sh
```

В файле /etc/postgresql/11/main/postgresql.conf изменить следующие опции:

```
ac_ignore_socket_maclabel = false
ac_enable_grant_options = true
log_destination = 'stderr,csvlog'
log_file_mode = 0664
lc_messages = 'C'
```

После завершения настроек необходимо произвести перезагрузку сервиса PostgreSQL командой:

```
sudo systemctl restart postgresql
```

4.2.1.2. Настройка СУБД для аутентификации по протоколу Kerberos

Для работы СУБД в домене необходимо пройти аутентификацию учетной записью администратора домена:

```
sudo kinit admin
```

Создать принципала службы СУБД (на примере имени сервера СУБД «hv-manager.domain.loc»):

```
sudo ipa service-add postgres/hv-manager.domain.loc@DOMAIN.LOC
```

После прохождения аутентификации необходимо получить таблицу ключей для СУБД (на примере имени контроллера домена «freeipa.domain.loc»):

```
sudo ipa-getkeytab -s freeipa.domain.loc -k /etc/postgresql/11/main/krb5.keytab -p postgres/hv-manager.domain.loc@DOMAIN.LOC
```

После получения файла с таблицей ключей установить файлу необходимые атрибуты:

```
sudo chown postgres:postgres /etc/postgresql/11/main/krb5.keytab
```

```
sudo chmod 600 /etc/postgresql/11/main/krb5.keytab
```

Для обеспечения возможности чтения прав доступа пользователей из служб контроллера домена необходимо добавить uid пользователя postgres в список разрешенных пользователей службы sssd следующим образом:

- получить uid пользователя postgres командой:

```
id postgres
```

- добавить полученный uid в список в значении параметра allowed_uids блока настроек ifp в файле /etc/sssd/sssd.conf:

```
[ifp]
allowed_uids = 0, 33, 114, 115, 999
```

- перезапустить службу sssd:

```
sudo systemctl restart sssd
```

Необходимо изменить параметры СУБД, добавив в файл /etc/postgresql/11/main/postgresql.conf следующие параметры:

```
ac_ignore_socket_maclabel = false
listen_addresses = '0.0.0.0,localhost'
krb_server_keyfile = '/etc/postgresql/11/main/krb5.keytab'
ac_enable_grant_options = true
log_destination = 'stderr,csvlog'
log_file_mode = 0664
lc_messages = 'C'
```

Также необходимо скорректировать параметры аутентификации в файле /etc/postgresql/11/main/pg_hba.conf:

заменить:

```
host    all         all         0.0.0.0/0           md5
host all  all  ::1/128           md5
```

на:

```
host    all         all         0.0.0.0/0           gss include_realm=0
```

После завершения настроек необходимо произвести перезагрузку сервиса PostgreSQL командой:

```
sudo systemctl restart postgresql
```

4.2.2. Настройка Apache HTTP Server

4.2.2.1. Настройка конфигурации HTTP-сервера для локальной аутентификации

Для управления серверным приложением комплекса средств управления необходимо включить модуль авторизации следующим образом:

```
sudo a2enmod authnz_pam
```

Отключить стандартный сайт следующей командой:

```
sudo a2dissite 000-default.conf
```

Для управления приложением комплекса средств управления hv-manager необходимо создать новый конфигурационный файл сайта и включить его:

```
sudo touch /etc/apache2/sites-available/hv-manager.conf
sudo a2ensite hv-manager.conf
```

Добавить в конфигурационный файл сайта /etc/apache2/sites-avialable/hv-manager.conf следующее содержимое:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /opt/hv-manager
    ServerName hv-manager.domain.loc
    Alias /favicon.ico /opt/hv-manager/static/favicon.ico
    Alias /static/ /opt/hv-manager/static/
    Alias /media/ /opt/hv-manager/media/
    Alias /assets/images/ /opt/hv-manager/static/assets/images/

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /opt/hv-manager/>
      AuthType Basic
      AuthName "PAM authentication"
      AuthBasicProvider PAM
      AuthPAMService apache2
      Require valid-user
    </Directory>
```

```
    WSGIScriptAlias / /opt/hv-manager/backend/backend/wsgi.py
    WSGIApplicationGroup %{GLOBAL}
</VirtualHost>
WSGIPythonPath /opt/hv-manager/backend/:/usr/local/lib/python3.7/dist-packages/
```

«hv-manager.domain.loc» необходимо заменить на имя сервера.

После завершения настройки необходимо перезапустить сервис HTTP-сервера следующей командой:

```
sudo systemctl restart apache2
```

4.2.2.2. Настройка конфигурации HTTP-сервера для аутентификации по протоколу Kerberos

Для выполнения настроек необходимо пройти аутентификацию учетной записью администратора домена:

```
sudo kinit admin
```

Создать принципала службы HTTP-сервера (на примере имени сервера HTTP «hv-manager.domain.loc»):

```
sudo ipa service-add HTTP/hv-manager.domain.loc@DOMAIN.LOC
```

После прохождения аутентификации необходимо получить таблицу ключей для HTTP-сервера (на примере имени контроллера домена «freeipa.domain.loc»):

```
sudo ipa-getkeytab -s freeipa.domain.loc -k /etc/apache2/http.keytab -p HTTP/hv-manager.domain.loc@DOMAIN.LOC
```

Для управления серверным приложением комплекса средств управления необходимо включить модуль авторизации следующим образом:

```
sudo a2enmod auth_kerb
```

Отключить стандартный сайт следующей командой:

```
sudo a2dissite 000-default.conf
```

Для управления приложением комплекса средств управления hv-manager необходимо создать новый конфигурационный файл сайта и включить его:

```
sudo touch /etc/apache2/sites-available/hv-manager.conf
sudo a2ensite hv-manager.conf
```

Установить файлу ключей необходимые атрибуты:

```
sudo chown www-data:www-data /etc/apache2/http.keytab
```

```
sudo chmod 600 /etc/apache2/http.keytab
```

Добавить в файл /etc/apache2/sites-avialable/hv-manager.conf следующее содержимое:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /opt/hv-manager
    ServerName hv-manager.domain.loc
    Alias /favicon.ico /opt/hv-manager/static/favicon.ico
    Alias /static/ /opt/hv-manager/static/
    Alias /media/ /opt/hv-manager/media/
    Alias /assets/images/ /opt/hv-manager/static/assets/images/

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /opt/hv-manager/>
                AuthType Kerberos
                KrbAuthRealms DOMAIN.LOC
                KrbServiceName HTTP/hv-manager.domain.loc@DOMAIN.LOC
                Krb5Keytab /etc/apache2/http.keytab
                KrbMethodNegotiate on
                KrbMethodK5Passwd off
                KrbSaveCredentials on
                require valid-user
    </Directory>

    WSGIScriptAlias / /opt/hv-manager/backend/backend/wsgi.py
    WSGIApplicationGroup %{GLOBAL}
</VirtualHost>
WSGIPythonPath /opt/hv-manager/backend/:/usr/local/lib/python3.7/dist-packages/
```

«hv-manager.domain.loc» необходимо заменить на имя сервера.

После завершения настройки необходимо перезапустить сервис HTTP-сервера следующей командой:

```
sudo systemctl restart apache2
```

4.2.3. Установка ПО комплекса средств управления

Для установки дистрибутива необходимо выполнить скрипты из корневого каталога дистрибутива комплекса средств управления следующими командами:

```
sudo ./install-packages.sh
sudo ./install-dependencies.sh
./install-ksu.sh
```

Назначить метки МРД на файлы и папки:

```
pdpl-file 3:0:0:ccnr /opt/hv_manager/
pdpl-file 3:0:0:ccnr /opt/hv_manager/media
pdpl-file 3:0:0:ccnr /opt/hv_manager/backend
touch /opt/hv_manager/error.log
pdpl-file 0:0:0: /opt/hv_manager/error.log
touch /opt/hv_manager/error_1-0.log
pdpl-file 1:0:0: /opt/hv_manager/error_1-0.log
touch /opt/hv_manager/error_2-0.log
pdpl-file 2:0:0: /opt/hv_manager/error_2-0.log
touch /opt/hv_manager/error_2-0.log
pdpl-file 3:0:0: /opt/hv_manager/error_3-0.log
```

Средствами ОС либо контроллера домена добавить пользователя планировщика для комплекса средств управления и пользователей для операторов.

Назначить права на файлы для всех пользователей:

```
sudo find /opt/hv-manager/ -type d -exec chmod 777 {} \;
sudo find /opt/hv-manager/ -type f -exec chmod 666 {} \;
sudo find /opt/hv-manager/ -name "*.sh" -exec chmod 777 {} \;
sudo find /opt/hv-manager/ -name "*.js" -exec chmod 777 {} \;
```

4.2.3.1. Настройка конфигурации для локальной аутентификации

Для настройки окружения необходимо создать файл /opt/hv-manager/backend/.env.local и добавить в него следующее содержимое:

```
DATABASE_URL=postgres:///hv_manager
DEBUG=False
SERVE_STATIC=False
ALLOWED_HOSTS=localhost,127.0.0.1,0.0.0.0,hv-manager.domain.loc
ERROR_LOG_FILE=/opt/hv-manager/error.log
```

```
ERROR_LOG_DIR=/opt/hv-manager
MAC_ENABLE=true
WEBSOCKIFY_RUNNER=/opt/websockify/run
HOST_BACKUPS_DIR=/home/backup
```

«hv-manager.domain.loc» необходимо заменить на имя сервера.

Для обновления информационной модели комплекса средств управления необходимо выполнить миграции:

```
cd /opt/hv-manager/backend
sudo -u postgres python3 manage.py migrate
python3 manage.py collectstatic
```

Для работы планировщика в условиях механизма МРД предусмотрен запуск нескольких процессов планировщика на каждый уровень конфиденциальности. Для этого необходимо создать файлы /etc/systemd/system/ksu_N.service, где N - уровень конфиденциальности в условиях механизма МРД, и добавить в них следующее содержимое:

```
[Unit]
Description=KSU scheduler MAC N
After=network.target postgresql.service

[Service]
Type=simple
User=ksu_shed
PDPLabel=N:0:0
WorkingDirectory=/opt/hv-manager/backend/
ExecStart=python3 manage.py runapscheduler
ExecStop=kill -SIGINT $(ps ax | grep "runapscheduler" | grep -v grep | awk '{print $1}')

[Install]
WantedBy=multi-user.target
```

«N» в параметрах Description и PDPLabel необходимо заменить на соответствующий уровень конфиденциальности.

В параметре «User» нужно указать имя пользователя планировщика, ранее добавленного в ОС и комплекс средств управления как обычного пользователя.

После выполнения настроек необходимо обновить службы:

```
sudo systemctl daemon-reload
sudo systemctl enable ksu
sudo systemctl restart ksu
```

4.2.3.2. Настройка конфигурации для доменной аутентификации

Для настройки окружения необходимо создать файл /opt/hv-manager/backend/.env.local и добавить в него следующее содержимое:

```
DATABASE_URL=postgres://hv-manager.domain.loc/hv_manager
DEBUG=False
SERVE_STATIC=False
ALLOWED_HOSTS=localhost,127.0.0.1,0.0.0.0, hv-manager.domain.loc
ERROR_LOG_FILE=/opt/hv-manager/error.log
ERROR_LOG_DIR=/opt/hv-manager
MAC_ENABLE=true
KERBEROS_ENABLED=True
WEBSOCKIFY_RUNNER=/opt/websockify/run
HOST_BACKUPS_DIR=/home/backup
```

«hv-manager.domain.loc» необходимо заменить на имя сервера.

Для обновления информационной модели комплекса средств управления необходимо выполнить миграции:

```
cd /opt/hv-manager/backend
sudo -u postgres python3 manage.py migrate
python3 manage.py collectstatic
```

Для работы планировщика в условиях механизма МРД предусмотрен запуск нескольких процессов планировщика на каждый уровень конфиденциальности. Для этого необходимо создать файлы /etc/systemd/system/ksu_N.service, где N - уровень конфиденциальности в условиях механизма МРД, и добавить в них следующее содержимое:

```
[Unit]
Description=KSU scheduler MAC N
After=network.target sssd.service postgresql.service

[Service]
```

```
Type=simple
User=ksu_shed
PDPLabel=N:0:0
WorkingDirectory=/opt/hv-manager/backend/
RuntimeMaxSec=12h
Restart=on-abnormal
ExecStartPre=/usr/bin/kinit -k -t /opt/hv-manager/ksu_shed.keytab ksu_shed@DOMAIN.LOC
ExecStart=python3 manage.py runapscheduler
ExecStop=kill -SIGINT "$(ps ax | grep runapscheduler | grep -v grep | awk '{print $1}')"

[Install]
WantedBy=multi-user.target
```

«N» в параметрах Description и PDPLabel необходимо заменить на соответствующий уровень конфиденциальности.

В параметре «User» нужно указать имя пользователя планировщика, ранее добавленного в ОС и комплекс средств управления как обычного пользователя. Используя реквизиты пользователя, нужно хотя бы раз войти в ОС, чтобы создались пользовательские структуры.

Для настройки необходимо пройти аутентификацию учётной записью администратора домена:

```
sudo kinit admin
```

Затем создать таблицу ключей, чтобы планировщик мог самостоятельно подключаться к СУБД для обновления данных:

```
ipa-getkeytab -s freeipa.domain.loc -k /opt/hv-manager/ksu_shed.keytab -p ksu_shed@DOMAIN.LOC
chown ksu_shed:ksu_shed /opt/hv-manager/ksu_shed.keytab
chmod 600 /opt/hv-manager/ksu_shed.keytab
```

После создания таблицы ключей пользователь утратит возможность входа в систему по паролю.

После выполнения настроек необходимо обновить службы:

```
sudo systemctl daemon-reload
sudo systemctl enable ksu
sudo systemctl restart ksu
```

4.3. Установка ПО Терминала

ПО тонкого клиента загружается, используя технологию Linux Terminal Server Project (LTSP). Для подготовки серверной части ТК необходимо настроить сервер с TFTP, NFS, DHCP серверами.

Для установки LTSP необходимо установить нужные пакеты, для этого выполнить команду.

```
sudo apt install ltsp-server-standalone
```

Для установки пакетов TFTP и DHCP выполнить команду.

```
sudo apt install isc-dhcp-server tftpd-hpa
```

Для создания конфигурационного файла DHCP сервера выполнить команду.

```
sudo ltsp-config --overwrite isc-dhcp-server
```

Скопировать получившийся файл.

```
sudo cp /etc/ltsp/dhcpd.conf /etc/dhcp/dhcpd.conf
```

Привести конфигурационный файл /etc/dhcp/dhcpd.conf к следующему виду, при необходимости заменив настройки IP адресации.

```
subnet 192.168.67.0 netmask 255.255.255.0 {
range 192.168.67.20 192.168.67.250;
option domain-name "example.com";
option domain-name-servers 192.168.67.1;
option broadcast-address 192.168.67.255;
option routers 192.168.67.1;
next-server 192.168.67.1;
option subnet-mask 255.255.255.0;
option root-path "/opt/ltsp/amd64";
}
```

Настроить TFTP-сервер, отредактировав конфигурационный файл /etc/default/tftpd-hpa:

```
TFTP_DIRECTORY="/var/lib/tftpboot"
```

Создать образ тонкого клиента, выполнив команду.

```
sudo ltsp-build-client --dist 1.7_x86-64 --mirror "https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-main" \
```

```
--early-mirror "https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-base" \
--purge-chroot --components contrib,main,non-free \
--kernel-packages linux-image-5.10-generic,dpkg --mode 2
```

Доустановить в загрузочный образ необходимые пакеты, выполнив команду:

```
ltsp-chroot -m apt install vncviwer libgtk3
```

Скопировать необходимые библиотеки POCO в каталог /usr/lib/x86_64-linux-gnu, а исполняемый файл tk_connector в каталог /usr/local/bin образа LTSP.

Обновить созданный образ:

```
sudo ltsp-update-image
```

Настроить NFS-сервер? отредактировав конфигурационный файл /etc/exports:

```
/opt/ltsp/amd64 *(rw,sync,no_root_squash)
```

Выполнив команду, применить изменения:

```
exportfs
```

Настроить TFTP-сервер для использования загрузки ядра и загрузочного образа initrams, отредактировав конфигурационный файл default в каталоге /srv/tftp/pxelinux.cfg.

Настроить конфигурационный файл TFTP сервера /etc/default/tftpd-hpa:

```
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/srv/tftp"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure  -l -v -m /etc/tftpd.remap"
```

Настроить автозапуск X сервера и автоматический запуск исполняемого файла tk_connector.

5. НАСТРОЙКА МЕНЕДЖЕРА ВИРТУАЛЬНЫХ МАШИН И КОНФИГУРАЦИЯ ВМ

5.1. Конфигурационные файлы менеджера виртуальных машин

Большинство конфигурационных файлов менеджера виртуальных машин, доступных в Dom0, хранятся в каталоге /etc/xen (включая файлы конфигурации домена). В подкаталоге скриптов /etc/xen/scripts хранятся сценарии для обработки задач: к примеру, задачи создания виртуальных устройств.

В каталоге /etc/init.d хранятся сценарии для запуска и остановки сервисов, используемых в работе менеджера виртуальных машин.

Сценарий /etc/inid.d/xendomains автоматически сохраняет домены при выключении системы и восстанавливает их при загрузке. Сценарий /boot/grub/menu.lst сообщает загрузчику GRUB о необходимости загрузить ядро менеджера виртуальных машин, сместив ядро Dom0 к модульной линии. В этом файле переопределяются параметры загрузки и менеджера виртуальных машин, и самой гостевой ОС. К примеру, если требуется уточнить распределение фиксированной памяти для Dom0 с помощью опции менеджера виртуальных машин dom0_mem или увеличить количество сетевых петлевых устройств с помощью опции ядра управляющего домена nloopbacks.

Примеры конфигурационных файлов для различных виртуальных машин приведены в п. 3.7.

5.2. Инструмент управления менеджера виртуальных машин — команда «xl»

5.2.1. Описание команды xl

Команда xl, основанная на библиотеке libxenlight — это инструмент управления менеджером виртуальных машин.

Команда xl управляет гостевыми доменами: создаёт, приостанавливает и выключает домены. Также, с помощью команды xl можно вывести список текущих

доменов, запустить и закрепить виртуальные процессоры (vCPU), подсоединить или отсоединить виртуальные блочные устройства, PCI устройства, и многое другое.

Базовая структура команды:

подкоманда xl [опции] domain-id.

где domain-id — это численный идентификатор или имя домена, а опции — конкретные параметры подкоманды.

При работе с xl следует помнить о следующем:

- большинство операций xl основаны на работе сервисов xenstored и xenconsoled, соответственно, во время загрузки dom0 обязательно должен быть запущен скрипт /etc/ init.d/xencommons, ответственный за инициализацию этих сервисов;
- в большинстве сетевых конфигураций, для создания рабочей сети, необходимо настроить мост с именем xenbrX в Dom0;
- если объем памяти Dom0 указан посредством параметра domO_mem при загрузке менеджера виртуальных машин, следует отключить (приравнять к нулю) параметр autoballoon в файле /etc/ xen/xl.conf;
- большинство подкоманд xl требуют запуска с правами администратора.

5.2.2. Опции и команды xl

К общим опциям xl относятся:

- v — повысить детализацию вывода команды;
- n — холостой прогон команды;
- f — выполнить принудительно; обеспечивает запуск даже небезопасных команд;
- t — всегда использовать перевод строки CR/LF для сообщений о выполнении печати без прокрутки экрана (иначе это выполняется только в случае, если поток stderr является устройством вывода консоли).

5.2.2.1. Подкоманды домена

С помощью подкоманд домена можно управлять доменами напрямую. В качестве первого параметра большинство подкоманд используют domain-id. К примеру, команда нажатие кнопки domain-id button обозначает нажатие кнопки интерфейса ACPI для домена («Включён» или «Спящий режим»).

Информация по подкоманде create:

1) синтаксис - create [configfile] [опции];

2) назначение - создает гостевой домен;

3) возвращение управления команде xl - как только запущен сам домен. Это не означает, что гостевая ОС в этом домене загружена или доступна для ввода.

4) аргументы - onfigfile — абсолютный путь к конфигурационному файлу домена (подробная информация о таких файлах содержится в файле xl.cfg). Если этот аргумент не задан или файл отсутствует, xl создаст домен, используя значения по умолчанию для каждой опции;

5) опции:

- -q --quiet — запретить консольный вывод;
- -p — приостановить домен после его создания;
- -V, --vncviewer — подключиться к VNC- серверу домена, используя vncviewer, запущенный в отдельном процессе;
- -A, --vncviewer-autopass — передать пароль для vncviewer с помощью стандартного ввода (stdin);
- -c — подсоединить консоль к домену после его запуска. Используется для обнаружения проблем, связанных с внезапным отключением домена, а также для общего удобства, например, для наблюдения за процессом загрузки домена;
- key=value — указать опции в формате опций конфигурационного файла. Такие пары ключ=значение перекрывают аналогичные, заданные в конфигурационном файле. Чтобы избежать неправильной интерпретации отдельных символов (например, скобок или кавычек) в параметрах

конфигурации, все дополнительные следует указывать через точку с запятой в одних общих кавычках.

Пример.

```
xl create DebianLenny
```

Эта команда создаст домен с файлом /etc/xen/DebianLenny и вернёт управление, как только он запустится.

Пример.

С дополнительными опциями:

```
xl create hvm.cfg 'cpus="0-3"
pci=["01:05.1","01:05.2"]'
```

Эта команда создаст домен с файлом конфигурации hvm.cfg, свяжет его с ЦП (CPU) 0-3 и предоставит прямой доступ к двум PCI-устройствам.

Информация по подкоманде console:

1) синтаксис - console [опции] domain-id;

2) опции:

- -t [pv\serial] подключить к PV-консоли или к эмулируемой последовательной консоли. PV- консоли доступны лишь для PV-доменов. HVM- домены могут иметь оба вида консолей. Если опция не указана, по умолчанию используются эмулированные последовательные консоли для HVM ОС и PV-консоли для PV ОС;
- -n NUM — подключить к консоли с номером NUM. Номера консолей начинаются с 0.

Информация по подкоманде destroy:

1) синтаксис - destroy [опции] domain-id;

2) назначение - немедленно завершает работу домена; ОС домена не успевает среагировать. В большинстве случаев рекомендуется использовать подкоманду выключения shutdown;

3) опции - -f — разрешить уничтожение Dom0. Поскольку домен не может уничтожить сам себя, выполнение опции возможно только с использованием

разделённой структуры и наиболее полезно тогда, когда аппаратный домен отделен от Dom0.

Информация по подкоманде list:

1) синтаксис - list [опции] [domain-id …];

2) назначение - выводит список с информацией об одном или нескольких доменах. Если имена доменов не указаны, выводит информацию обо всех доменах;

3) опции:

- -l, --long — вывести список не в виде таблицы, а в виде структуры данных JSON;
- -Z, --context — вывести дополнительно метки защиты;
- -v, --verbose — вывести дополнительно UUID домена, причины выключения и метки защиты.

Формат вывода списка:

Name ID Mem VCPUs State Time(s) Domain-0 0 750 4
r----- 11794.3 win 1 1019 1 r----- 0.3 linux 2 2048 2 r----- 5624.2

В списке выведены следующие данные:

- Name — имя домена;
- ID — числовой идентификатор домена;
- Mem — объём памяти, предоставляемый домену;
- VCPUs — количество виртуальных ЦП, предоставленных домену;
- State — состояние работы;
- Time — общее время выполнения домена, учитываемое менеджером виртуальных машин.

В колонке State отображаются состояния домена:

- r — running, в настоящее время домен работает на ЦП;
- b — blocked, домен заблокирован, не работает или не способен работать; это может быть вызвано тем, что домен находится в ожидании ввода-вывода или в спящем режиме;

- p — paused, домен приостановлен, как правило, после использования команды xl pause; в приостановленном состоянии домен продолжает потреблять предоставленные ему ресурсы (например, память), но он не доступен планировщику менеджера виртуальных машин;
- s — shutdown, гостевая ОС выключена (был произведён вызов SCHEDOP_shutdown), но домен ещё не выключен полностью;
- c — crashed, работа домена была завершена принудительно (как правило, домен настроен на перезапуск после принудительного завершения);
- d — dying, домен находится в процессе выключения, но не полностью выключен, или вышел из строя.

Информация по подкоманде migrate:

1) синтаксис - migrate [опции] domain-id host;

2) назначение - переносит домен на другую хост-машину (по умолчанию, с помощью SSH);

3) опции:

- -s sshcommand — использовать sshcommand вместо SSH;
- -e — не ждать в фоновом режиме полного выключения домена на новом хосте;
- -C config — заменить файл конфигурации новым файлом config;
- -debug — вывести информацию об отладке в процессе миграци;
- -c — подсоединить консоль к домену после его запуска. Используется для обнаружения проблем, связанных с внезапным отключением домена, а также для общего удобства, например, для наблюдения за процессом загрузки домена.

Информация по подкоманде remus:

1) синтаксис - remus [опции] domain-id host;

2) назначение - задействует технологию Remus HA на домене (по умолчанию, с помощью SSH);

3) опции:

- -i MS — создавать контрольные точки памяти домена каждую миллисекунду (по умолчанию 200 мс);
- -u — отключить сжатие контрольных точек памяти;
- -s sshcommand — использовать sshcommand вместо SSH;
- -e — не ждать в фоновом режиме полного выключения домена на новом хосте;
- -N netbufscript — использовать netbufscript для настройки сети буферизации вместо сценария по умолчанию (/etc/xen/scripts/remus- netbuf-setup);
- -F — запустить Remus в небезопасном режиме;
- -b — продублировать контрольные точки памяти в /dev/null (черная дыра). Полезная опция для отладки. Требует включения небезопасного режима;
- -n — отключить сетевую буферизацию вывода. Требует включения небезопасного режима$
- -d — отключить репликацию диска. Требует включения небезопасного режима;
- pause domain-id — приостановить домен. В приостановленном состоянии домен продолжает потреблять предоставленные ему ресурсы (например, память), но он не доступен планировщику менеджера виртуальных машин.

Информация по подкоманде reboot:

1) синтаксис - reboot [опции] domain-id;

2) назначение - перезагружает домен. Использование подкоманды равносильно перезагрузке из консоли. Для HVM- доменов потребуются PV-драйверы, установленные в гостевой ОС. Если драйверы отсутствуют, но ОС настроена правильно, для имитации нажатия кнопки перезапуска можно использовать опцию -f. Процессы, сопутствующие перезапуску, настраиваются с помощью параметра on_reboot в конфигурационном файле домена;

3) возвращение управления команде xl - подкоманда возвращает управление в момент перезагрузки (до фактической загрузки домена);

4) опции - -f — включает откат к отправке события включения ACPI в случае, если гостевая ОС не поддерживает управление перезагрузкой с помощью PV. Требует соответствующей настройки ОС.

Информация по подкоманде restore:

1) синтаксис - Restore [опции] [ConfigFile] CheckpointFile;

2) назначение - создаёт домен из файла состояния, созданного с помощью подкоманды save;

3) опции:

- -p — не активировать домен после его восстановления;
- -e — не ждать в фоновом режиме полного выключения домена на новом хосте;
- -d — включить отладочные сообщения;
- -V, --vncviewer — подключиться к VNC- серверу домена, используя vncviewer, запущенный в отдельном процессе;
- -A, --vncviewer-autopass — передать пароль для vncviewer с помощью стандартного ввода (stdin).

Информация по подкоманде save:

1) синтаксис - save [опции] domain-id CheckpointFile [configfile];

2) назначение - сохраняет работающий домен для последующего восстановления. По умолчанию, домен не будет использоваться. Через параметр configfile передаётся конфигурационный файл домена;

3) опции:

- -c — после создания копии домен продолжает работать;
- -p — после создания копии домен переводится в приостановленное состояние;
- sharing [domain-id] — вывести список количества общих страниц.

Информация по подкоманде shutdown:

1) синтаксис - shutdown [опции] -a|domain-id;

2) назначение - аккуратно отключает домен в координации с ОС домена. Для HVM-доменов потребуются PV- драйверы, установленные в гостевой ОС.

Если драйверы отсутствуют, но ОС настроена правильно, для имитации нажатия кнопки перезапуска можно использовать опцию - f. Процессы, сопутствующие перезапуску, настраиваются с помощью параметра on_shutdown в конфигурационном файле домена.

Информация по подкоманде vncviewer:

1) синтаксис - vncviewer [опции] domain-id;

2) назначение - Подключает к VNC-серверу домена, запущенному в отдельном процессе;

3) опции:

- --autopass — передать пароль для vncviewer с помощью стандартного ввода;
- -w, --wait — ожидать выключения домена перед возвратом из команды;
- -f — использовать отправку события включения ACPI для выключения домена, если гостевая ОС не поддерживает управление перезагрузкой через PV- драйвер. Требует соответствующей настройки ОС.

Дополнительные подкоманды домена:

- mem-max domain-id mem — указать максимальный объём памяти, который может использовать домен. Приставки: «t» для терабайт, «g» для гигабайт, «m» для мегабайт, «k» для килобайт и «b» для байт. Значение mem-max может не соответствовать реальной памяти, используемой доменом, поскольку часть места используется ОС;
- mem-set domain-id mem — настроить используемую память домена с помощью драйвера balloon. Данная операция требует поддержки со стороны ОС домена. Гарантий ее выполнения нет. Операция не сработает, если у домена нет требуемого PV-драйвера. Не существует проверенного способа заранее узнать, какое количество mem-set приведет к нестабильности и падению домена;
- domid domain-name — преобразовать имя домена в идентификатор домена;

- domname domain-id — преобразовать идентификатор домена в имя домена;
- rename domain-id new-name — изменить имя домена domain- id на new-name;
- dump-core domain-id [filename] — создать образ памяти виртуальной машины для указанного домена с именем filename без приостановки ее работы. Файл образа будет записан в директорию для файлов образов (например, /var/lib/xen/dump);
- help [--long] — вывести короткое справочное сообщение (например, распространённые команды). Опция --long выводит полный набор подкоманд xl, сгруппированных по функциям;
- sysrq domain-id letter — отправить Magic System Request домену, при наличии PV-драйверов в гостевой ОС. Может использоваться для отправки SysRq-запросов гостевым ОС Linux. Подробное описание запросов доступно в файле sysrq.txt в исходных файлах ядра Linux;
- trigger domain-id nmi|reset|init|power|sleep|s3resume [VCPU] — отправить домену триггер (nmi, reset, init, power или sleep). Дополнительно, определённое число виртуальных ЦП (VCPU) может быть передано в качестве аргумента. Эта команда доступна только для HVM-доменов;
- unpause domain-id — вывести домен из приостановленного состояния. Позволяет ранее остановленным доменам иметь право на планирование посредством менеджера виртуальных машин;
- vcpu-set domain-id vcpu-count — количество ЦП в vcpu-count для указанного домена. Как и mem-set, эта команда выделяет только максимальное число виртуальных ЦП, заданное при загрузке домена. Если vcpu-count меньше, чем текущее количество активных виртуальных ЦП, лишние виртуальные ЦП будут автоматически удалены. Это сказывается на процессе привязки vCPU к физическим CPU. Попытка установить VCPU в количестве большем, чем изначально указано в настройках, вызовет ошибку. Попытка установить VCPU меньше 1 будет проигнорирована. Некоторым

гостевым ОС может потребоваться привести вновь добавленный ЦП в активное состояние после использования vcpu-set;

- vcpu-list [domain-id] — вывести список с информацией о виртуальных ЦП для конкретного домена. Если домен не указан, выводит информацию о виртуальных ЦП для всех доменов;

- vcpu-pin domain-id vcpu cpus hard cpus soft — установить «жёсткую» и «мягкую» привязку для виртуального ЦП vcpu домена с идентификатором domain-id. Обычно виртуальные ЦП перемещаются между доступными физическими ЦП в соответствии с командами менеджера виртуальных машин. «Жёсткая» привязка ограничивает этот процесс, чтобы убедиться, что конкретные виртуальные процессоры могут работать только на определённых физических процессорах. «Мягкая» привязка задаёт приоритетный набор процессоров. «Мягкая» привязка нуждается в специальной поддержке со стороны планировщика (доступна только в варианте credit1). Ключевое слово all можно использовать, чтобы применить как «жёсткую», так и «мягкую» привязку для всех виртуальных ЦП в домене. Символ «-» можно использовать, чтобы оставить только «жёсткую» или только «мягкую» привязку. Например, команда xl vcpu-pin 0 3 — 6-9 установит «мягкую» привязку для виртуального ЦП 3 домена Dom0 к физическими ЦП 6, 7, 8 и 9, оставляя «жёсткую» привязку незатронутой. Команда xl vcpu-pin 0 3 3, 4 6-9 установит параметры как «жесткой», так и «мягкой» привязки, первый к физическим ЦП 3 и 4, второй — к ЦП 6, 7, 8 и 9;

- vm-list — вывести информацию о гостях. Этот список не включает информацию о сервисе или вспомогательных доменах, таких как Dom0 и stub-домены. Формат для списка выглядит следующим образом:

UUID ID name 59e1cf6c-6ab9-4879-90e7-adc8d1c63bf5 2;
win 50bc8f75-81d0-4d53-b2e6-95cb44e2682e 3 linux.

5.2.2.2. Подкоманды хост-системы

Информация по подкоманде debug-keys:

1) синтаксис - debug-keys keys;

2) назначение - отправляет отладочную последовательность нажатия клавиш в менеджере виртуальных машин. Имеет тот же эффект, что тройное нажатие conswitch (по умолчанию, Ctrl+A) и затем последовательное нажатие клавиш keys.

Информация по подкоманде dmesg:

1) синтаксис - dmesg [-c];

2) назначение - читает буфер сообщений менеджера виртуальных машин, аналогично dmesg в ОС Linux. Буфер содержит информационные сообщения, предупреждения и сообщения об ошибках, созданные во время процесса загрузки менеджера виртуальных машин;

3) опции - -c, --clear — очищает буфер сообщений менеджера виртуальных машин.

Информация по подкоманде info:

1) синтаксис - info [-n, --numa];

2) назначение - выводит информацию о хосте менеджера виртуальных машин в формате имя : значение;

3) опции - -n, --numa — вывести список информации о топологии хоста NUMA.

Пример.

host : scarlett release : 3.1.0-rc4+ version : #1001 SMP Wed Oct 19 11:09:54 UTC 2011 machine : x86_64 nr_cpus : 4 nr_nodes : 1 cores_per_socket : 4 threads_per_core : 1 cpu_mhz : 2266 hw_caps : bfebfbff:28100800:00000000:00003b40:009ce3bd:00000000:00000001:00000000 virt_caps : hvm hvm_directio total_memory : 6141 free_memory : 4274 free_cpus : 0 outstanding_claims : 0 xen_major : 4 xen_minor : 2 xen_extra : -unstable xen_caps : xen-3.0-x86_64 xen-3.0-x86_32p hvm-3.0-x86_32 hvm-3.0-x86_32p hvm-3.0-x86_64 xen_scheduler : credit xen_pagesize : 4096 platform_params : virt_start=0xffff800000000000 xen_changeset : Wed Nov 02 17:09:09 2011 +0000 24066:54a5e994a241 xen_commandline : com1=115200,8n1 guest_loglvl=all dom0_mem=750M console=com1 cc_compiler : gcc version 4.4.5 (Debian 4.4.5-8) cc_compile_by : sstabellini cc_compile_domain : uk.xensource.com cc_compile_date : Tue Nov 8 12:03:05 UTC 2011 xend_config_format : 4.

Некоторые из указанных полей:

- hw_caps — вектор, показывающий, какие аппаратные возможности поддерживаются процессором; эквивалентно полю флагов в /proc/cpuinfo на обычной машине с ОС Linux;
- free_memory — доступная память (Мб), не выделенная под менеджер виртуальных машин или другие домены, или затребованная доменами;
- outstanding_claims — увеличение общего значения на 1, если сделан запрос на резервирование (в файле xl.conf) и установлен заказ на определённое количество страниц. Когда память домена заполняется, общее значение (outstanding_claims) уменьшается и постепенно приближается к нулю. Большую часть времени значение будет равно нулю. Но, если запущено несколько гостей и claim_mode включен, это значение может увеличиться/уменьшиться. Значение также влияет на free_memory — оно отражает свободную память в менеджере виртуальных машин, за вычетом вспомогательных страниц памяти для гостей;
- xen_caps — версия Xen и архитектуры. Значение архитектуры может быть одним из следующих: x86_32, x86_32p (т. е. включая PAE), x86_64, ia64;
- xen_changeset — номер ревизии исходного кода в системе контроля версий. Позволяет определить, из чего конкретно была скомпилирована версия менеджера виртуальных машин.

Информация по подкоманде top:

1) синтаксис — top;

2) назначение - выполняет команду xentop, которая обеспечивает мониторинг доменов в реальном времени.

Информация по подкоманде claims:

1) синтаксис — claims;

2) назначение - выводит информацию о дополнительной памяти гостей. Предоставляет информацию о подсчитанных дополнительных блоках памяти и выделенной памяти для гостей. Эти значения в сумме отражают общее значение запрошенной памяти, которое предоставляется с помощью info,

выделенной памяти для гостей. Эти значения в сумме отражают общее значение запрошенной памяти, которое предоставляется с помощью info, и значения outstanding_claims. Колонка Mem имеет совокупное значение запрошенной памяти и общего объёма памяти, которое было в данный момент выделено для гостей.

Пример.

Формат для списка:

```
Name ID Mem VCPUs State Time(s) Claimed Domain-0 0 2047
4 r       19.7 0 OL5 2 2048 1 --p--- 0.0 847 OL6 3 1024 4 r 5.9 0
Win_XP 4 2047 1 --p--- 0.0 1989
```

Гостю OL5 доступны 847 Мб заявленной памяти (из общих 2048 Мб, 1191 Мб из которых был выделен для гостя).

5.2.2.3. Команды планировщика

Гипервизор поставляется с набором планировщиков домена, который может быть установлен во время загрузки с параметром sched= в командной строке менеджера виртуальных машин. По умолчанию, кредитный лимит (credit) используется для планирования.

Информация по подкоманде sched-credit:

1) синтаксис - sched-credit [опции].

2) назначение - устанавливает или получает параметры кредитного планировщика. Кредитный планировщик соответствует планировщику ЦП, выделяющему ресурсы равными долями, работающему на симметричных многопроцессорных системах. Каждому домену выделяющему ресурсы равными долями, работающему на симметричных многопроцессорных системах. Каждому домену (включая Dom0) присваиваются вес и верхний лимит;

3) опции:

- -d DOMAIN, --domain=DOMAIN — указать домен, для которого необходимо изменить или восстановить параметры планировщика. Опция обязательна для изменения параметров планировщика;

- -w WEIGHT, --weight=WEIGHT — домен с весом 512 получит в два раза больше ЦП, чем домен с весом 256 на утвержденном хосте. Разрешенный вес варьируется от 1 до 65535, по умолчанию равен 256;

- -c CAP, --cap=CAP — верхний лимит дополнительно фиксирует максимальное количество ресурсов ЦП, которое домен сможет использовать, даже если на хост-системе есть «пустое» количество циклов процессора. Лимит задается в процентах одного физического ЦП: 100 — это 1 физический ЦП, 50 — это половина ЦП, 400 — это 4 ЦП и т.д. Число по умолчанию, 0, означает, что ограничение отсутствует. Во многих системах есть функции, уменьшающие вычислительную мощность процессора, который задействован не на 100%. Это может происходить как в самой операционной системе, так и уровнем ниже — в BIOS. Если параметр cap установлен таким образом, что отдельные ядра загружены менее, чем на 100%, это может оказать дополнительное влияние на эффективность рабочей загрузки. Например, если процессор работает на частоте 2 ГГц и установлено ограничение ВМ на 50%, система управления питанием может уменьшить тактовую частоту до 1 ГГц; в результате ВМ будет получать лишь 25% от доступной мощности (50% от 1 ГГц), а не 50% (50% от 2 ГГц);

- --p CPUPOOL, --cpupool=CPUPOOL — ограничить вывод к доменам в указанном cpupool;

- -s, --schedparam — добавить в список или настроить параметры планировщика, относящиеся ко всему пулу;

- -t TSLICE, --tslice_ms=TSLICE — указать планировщику, сколько по времени должны работать виртуальные машины до принудительного возврата управления. По умолчанию, 30 мс. Допустимый диапазон варьируется от 1 мс до 1000 мс. Продолжительность кванта времени (мс) должна быть больше, чем продолжительность RLIMIT;

- -r RLIMIT, --ratelimit_us=RLIMIT — параметр RLIMIT пытается ограничить количество запланированных действий в секунду. Оно устанавливает минимальное количество времени (мс), которое должна

отработать виртуальная машина, прежде чем планировщик позволит ВМ с более высоким приоритетом получить контроль. Значение по умолчанию составляет 1 мс. Допустимый диапазон составляет от 100 до 500 000 мкс. Величина RLIMIT должна быть меньше, чем продолжительность кванта времени TSLICE.

Сочетания приведённых выше параметров:

- d [domid] — вывести параметры домена для домена [domid];
- d [domid] [params] — установить параметры домена для домена [domid];
- p [pool] — перечислить все домены и параметры планировщика для [pool];
- s — перечислить все параметры планировщика для пула с poolid 0;
- s [params] — установить параметры планировщика для пула с poolid 0;
- p [pool] -s — перечислить все параметры планировщика для [pool];
- p [pool] -s [params] — установить параметры планировщика [params] для [pool]. Если параметры не указаны, будет выведен список параметров всех доменов и параметры планировщика из всех пулов.

Информация по подкоманде sched-credit2:

1) синтаксис - sched-credit2 [опции];

2) назначение - устанавливает или получает параметры планировщика credit2. Планировщик credit2 соответствует планировщику ЦП, выделяющему ресурсы равными долями, работающему на симметричных многопроцессорных системах. Каждому домену (включая Dom0) присваиваются вес и верхний лимит;

3) опции:

- -d DOMAIN, --domain=DOMAIN — указать домен, для которого необходимо изменить или восстановить параметры планировщика. Опция обязательна для изменения параметров планировщика;
- -w WEIGHT, --weight=WEIGHT — домен с весом 512 получит в два раза больше ЦП, чем домен с весом 256 на утверждённом хосте. Разрешённый вес варьируется от 1 до 65535, по умолчанию равен 256;

- -p CPUPOOL, --cpupool=CPUPOOL — ограничить вывод к доменам в указанном cpupool.

Информация по подкоманде sched-sedf:

1) синтаксис - sched-sedf [опции];

2) назначение - устанавливает или получает параметры планировщика Simple EDF. Этот планировщик предоставляет взвешенный ЦП-обмен «интуитивно» и использует алгоритмы реального времени для предоставления гарантий времени. Дополнительная информация доступна в файле docs/misc/sedf_scheduler_mini- HOWTO.txt дистрибутива;

3) опции:

- -d DOMAIN, --domain=DOMAIN — указать домен, для которого необходимо изменить или восстановить параметры планировщика. Опция обязательна для изменения параметров планировщика;
- -p PERIOD, --period=PERIOD — стандартное использование EDF планирования в миллисекундах;
- -s SLICE, --slice=SLICE — стандартное использование EDF-планирования в миллисекундах;
- -l LATENCY, --latency=LATENCY — масштабированный период времени, если домен выполняет интенсивный ввод /вывод;
- -e EXTRA, --extra=EXTRA — флаг, разрешающий домену работать в дополнительное время (0 или 1);
- -w WEIGHT, --weight=WEIGHT — установка кванта времени ЦП;
- -c CPUPOOL, --cpupool=CPUPOOL — ограничить вывод к доменам в указанном cpupool.

Информация по подкоманде sched-rtds:

1) синтаксис - sched-rtds [опции];

2) назначение - устанавливает или получает параметры планировщика rtds (Real Time Deferrable Server). Этот планировщик использует алгоритм планирования в реальном времени Preemptive Global Earliest Deadline First для планирования процессоров VCPU в системе. Каждый виртуальный ЦП имеет

выделенный период и бюджет. Виртуальные ЦП (VCPU) в одном домене имеют один и тот же период и бюджет. Во время планирования VCPU «сжигает» свой бюджет. Бюджет VCPU пополняется в начале каждого периода; неиспользованный бюджет «списывается» в конце каждого периода;

3) опции:

- -d DOMAIN, --domain=DOMAIN — указать домен, для которого необходимо изменить или восстановить параметры планировщика. Опция обязательна для изменения параметров планировщика;
- -p PERIOD, --period=PERIOD — период времени в микросекундах, в течение которого необходимо пополнить бюджет;
- -b BUDGET, --budget=BUDGET — количество времени в микросекундах, которое VCPU будет разрешено работать каждый период;
- -c CPUPOOL, --cpupool=CPUPOOL — ограничить вывод к доменам в указанном cpupool.

5.2.2.4. Команды управления пулами ЦП

Гипервизор может группировать физические процессоры сервера в группы, или «пулы» ЦП, для предоставления возможности использовать различные планировщики для разных ВМ.

Каждый физический процессор назначается максимум на один пул ЦП, каждый домен также ограничен одним пулом ЦП. Планирование не пересекает границы пула ЦП: каждому пулу ЦП назначен собственный планировщик. Физические процессоры и домены могут быть перемещены из одного пула ЦП в другой только с помощью описанной ниже команды. Пулы ЦП идентифицируются либо по имени, либо по ID.

Информация по подкоманде scpupool-create:

1) синтаксис - cpupool-create [опции] [configfile] [Variable=Value …];

2) назначение - создаёт пул ЦП на базе конфигурации из configfile или параметров командной строки. Значения переменных из configfile могут быть

изменены посредством установки новых или дополнительных назначений в командной строке;

3) опции:

- -f=FILE, --defconfig=FILE — использовать данный файл конфигурации;
- cpupool-list [-c|--cpus] [cpu-pool] — вывести список пулов ЦП на хосте. Если -c указано, xl выведет список процессоров, используемых пулом ЦП;
- cpupool-rename cpu-pool <newname> — переименовать пул ЦП в newname;
- cpupool-cpu-add cpu-pool cpu-nr|node:node-nr — добавить ЦП или все процессоры;
- cpupool-cpu-remove cpu-nr|node:node-nr — удалить ЦП или все процессоры узла NUMA из пула ЦП;
- cpupool-migrate domain cpu-pool — переместить домен, обозначенный с помощью domain-id или domain-name, в пул ЦП;
- cpupool-numa-split— разделить машину таким образом, что на каждый узел NUMA приходится один пул ЦП.

5.2.2.5. Команды управления виртуальными устройствами

Большинство виртуальных устройств можно добавлять или удалять во время работы ВМ, если гостевая ОС поддерживает подобный функционал. Для гостя эффект такой операции равнозначен событию «горячего подключения».

5.2.3. Блочные устройства

Информация по подкоманде block-attach:

1) синтаксис - block-attach domain-id disc-spec-component(s);

2) назначение - создаёт новое виртуальное блочное устройство;

3) опции:

- domain-id — идентификатор домена гостевого домена, к которому будет подключено устройство;
- disc-spec-component — спецификация диска в формате, аналогичном используемому для параметра disk в конфигурационном файле домена.

Информация по подкоманде block-dettach:

1) синтаксис - block-detach domain-id devid [--force];

2) назначение - отсоединяет виртуальное блочное устройство домена. devid — имя или числовой идентификатор устройства, присвоенный устройству доменом Dom0. Чтобы определить число, требуется запустить xl block- list. Отсоединение устройства требует взаимодействия с доменом. Если домен не позволяет отключить устройство (в том случае, если домен «завис» или всё еще использует данное устройство), отсоединиться не удастся. Параметр --force отсоединит устройство принудительно, но, возможно, вызовет ошибки ввода- вывода в домене.

Информация по подкоманде block-list:

1) синтаксис - block-list domain-id;

2) назначение - выводит список виртуальных блочных устройств для домена.

Информация по подкоманде cd-insert:

1) синтаксис - cd-insert domain-id VirtualDevice target;

2) назначение - вставляет cdrom в существующий виртуальный CD-привод гостевого домена. Виртуальный привод может быть пустым, но должен существовать. Работает только с HVM-доменами;

3) опции:

- dVirtualDevice — указать, каким образом устройство должно быть показано гостевому домену (например, hdc);
- target — путь назначения в back-end домене (обычно Dom0), который следует экспортировать (блочное устройство, файл и т. д.). Пример доступен в файле docs/misc/xl-disk-configuration.txt.

Информация по подкоманде cd-eject:

1) синтаксис - cd-eject domain-id VirtualDevice target;

2) назначение - извлекает cdrom из виртуального CD-привода гостя. Работает только с HVM-доменами;

3) опции - VirtualDevice — указать, каким образом устройство должно быть показано гостевому домену (например, hdc).

5.2.4. Сетевые устройства

Информация по подкоманде network-attach:

1) синтаксис - network-attach domain-id network-device;

2) назначение - создаёт новое сетевое устройство для домена domain-id. network-device задаёт устройство, которое следует присоединить (используя тот же формат, что и vif в файле конфигурации домена).

Информация по подкоманде network-detach:

1) синтаксис - network-detach domain-id devid|mac;

2) назначение - удаляет сетевое устройство из домена domain-id. devid — номер устройства виртуального интерфейса в домене (например, 3 в vif22.3). В качестве альтернативы для выбора виртуального интерфейса, который следует отсоединить, можно использовать MAC-адреса.

Информация по подкоманде network-list:

1) синтаксис - network-list domain-id;

2) назначение - выводит список виртуальных сетевых интерфейсов для домена.

5.2.5. Канальные устройства

Информация по подкоманде channel-list:

1) синтаксис - channel-list domain-id;

2) назначение - выводит список виртуальных канальных интерфейсов для домена.

5.2.5.1. Прямой доступ к PCI устройствам

Список команд управления прямым доступом к PCI устройствам:

- pci-assignable-list — вывести список всех PCI-устройств, доступных для прямого доступа из ВМ. Они привязаны к PCI backend драйверу, а не к реальному драйверу ядра dom0;
- pci-assignable-add BDF — подготовить устройство для прямого доступа из ВМ, определяемое PCI Bus/Device/Function (BDF). Эта команда свяжет устройство с драйвером pciback. Если устройство уже связано с каким либо драйвером, исходный драйвер будет откреплён от него и сохранен таким

образом, чтобы возможно было привязать устройство к этому драйверу повторно. Если устройство подготовлено, xl возвращает код success. Устройство будет непригодно для использования доменом Dom0, пока оно не будет возвращено командой xl pci-assignable-remove. Следует с осторожностью манипулировать устройствами, такими как дисковые и RAID контроллеры, сетевые интерфейсы или GPU, использующиеся в данный момент;

- pci-assignable-remove [-r] BDF — вернуть устройство, идентифицируемое Bus/Device/Function (BDF). Данная команда открепит устройство от pciback-драйвера. Если указана опция - r, будет предпринята попытка перепривязать устройство к его исходному драйверу, делая его вновь пригодным для использования доменом Dom0. Если устройство не привязано к драйверу pciback, xl возвращает код success;
- pci-attach domain-id BDF — «горячее» подключение нового PCI- устройства для прямого доступа в определённом домене(BM). BDF — идентификатор PCI устройства, ранее подготовленного для прямого доступа;
- pci-detach [-f] domain-id BDF — «горячее» отключение ранее подключённого PCI-устройства от домена, определяемого BDF. Если указана опция - f, xl принудительно удалит устройство, без взаимодействия с гостем;
- pci-list domain-id — вывести список PCI-устройств для прямой передачи для домена.

5.2.5.2. Пулы памяти tmem

Информация по подкоманде tmem-list:

1) синтаксис - tmem-list I[<-l>] domain-id;

2) назначение - выводит список пулов TMEM. Если указана опция -l, выводит также статистику TMEM.

Информация по подкоманде tmem-freeze:

1) синтаксис - tmem-freeze domain-id;

2) назначение - замораживает пулы TMEM.

Информация по подкоманде tmem-thaw:

1) синтаксис - tmem-thaw domain-id;

2) назначение - размораживает пулы TMEM.

Информация по подкоманде tmem-set:

1) синтаксис - tmem-set domain-id [опции];

2) назначение - меняет настройки TMEM;

3) опции:

- -w WEIGHT — вес (целое);
- -c CAP — верхний лимит (целое);
- -p COMPRESS — сжать (целое).

Информация по подкоманде tmem-shared-auth:

1) синтаксис - tmem-shared-auth domain-id [опции];

2) назначение - де/аутентифицирует поделённые пулы TMEM;

3) опции:

- -u UUID — определить UUID;
- --a AUTH — 0=auth, 1= deauth.

Информация по подкоманде tmem-freeable:

1) синтаксис — tmem-freeable;

2) назначение - выводит информацию о том, сколько свободной памяти (Мбайт) использует TMEM.

5.2.5.3. Система безопасности Flask

Flask — это система безопасности, которая определяет обязательную политику контроля доступа, обеспечивая гибкий контроль над доменами менеджера виртуальных машин и позволяя политике записи определять, какие взаимодействия между доменами, устройствами и менеджером виртуальных машин не допускаются.

Пример.

Пример применения системы безопастности Flask:

- запрет общения двух доменов через каналы событий и разделяемые страницы;

- контроль того, какие домены могут использовать прямой доступ к устройству (и какие устройства);
- ограничение или проверка операций, выполняемых привилегированными доменами;
- защита привилегированного домена от произвольного отображения страниц из других доменов.

Список команд Flask:

- getenforce — определить, загружен ли модуль безопасности Flask и выполняются ли его процедуры;
- setenforce 1|0|Enforcing|Permissive — включить или отключить контроль доступа Flask. По умолчанию включен; это поведение можно изменить с помощью опции flask_enforcing в командной строке менеджера виртуальных машин при его загрузке;
- loadpolicy policy-file — добавить политику Flask из заданного файла политики. Первичная политика предоставляется менеджеру виртуальных машин в качестве мультизагрузочного модуля. Эта команда позволяет осуществлять обновление работающей политики.

Загрузка новой политики безопасности сбросит изменения во время исполнения для меток устройств.

5.3. Синтаксис конфигурационного файла домена

5.3.1. Опции и команды xl

Для создания виртуальной машины (домена) с помощью команды xl необходим файл конфигурации создаваемого домена. Как правило, конфигурационный файл доступен по пути /etc/xen/DOMAIN.cfg, где DOMAIN — это имя домена.

Файл конфигурации домена состоит из последовательностей пар ключ=значение (KEY=VALUE). Пары могут быть разделены либо символом новой строки, либо точкой с запятой.

Ключи бывают следующих типов:

- обязательные ключи;
- общие параметры, применимые к любому типу домена;
- ключи, применяемые к определенным типам доменов (например, доменам PV или HVM).

Значения бывают следующих типов:

- ”STRING” — строка, заключённая в одинарные или двойные кавычки;
- NUMBER — десятичное, восьмеричное (с использованием префикса 0) или шестнадцатеричное (с использованием префикса 0x) число;
- BOOLEAN — число, интерпретируемое как «ложно» (0) или «истинно» (любое другое значение);
- [VALUE, VALUE, ...] — список значений указанных выше типов. Списки являются гомогенными и не сгруппированными. Семантика каждого ключа определяет, какое значение требуется.

5.3.2. Опции конфигурационного файла

5.3.2.1. Обязательные ключи

name="NAME" — задаёт имя домена. Имена доменов на одном хосте должны быть уникальными.

builder="value" — выбор типа гостя: value = generic для PV-домена (значение по умолчанию), value = hvm для HVM- домена.

5.3.2.2. Общие параметры

5.3.2.2.1. Распределение ЦП

pool="CPUPOOLNAME" — помещает гостевые ЦП в указанный пул ЦП.

vcpus=N — запускает домен с N доступными виртуальными ЦП (VCPU).

maxvcpus=M — разрешает гостю довести до максимума M число VCPU. Если vcpus=N меньше, чем maxvcpus, будет выделено N виртуальных процессоров, а оставшаяся часть будет находиться в резерве.

cpus="CPU-LIST" — список ЦП, которые гостю разрешается использовать. По умолчанию пиннинга нет совсем.

Пример.

all — разрешить всем VCPU гостя запустить все ЦП на хосте.

0-3,5,^1— разрешить всем виртуальным процессорам гостя работать на процессорах 0, 2, 3, 5. Можно совместить это с опцией all: all^7, которая означает работу на всех процессорах на хосте, кроме ЦП 7.

odes:0-3,node^2 — разрешить всем виртуальным процессорам гостя работать на процессорах NUMA узлов 0, 1, 3 хоста. Таким образом, если процессоры 0-3 относятся к узлу 0, процессоры 4-7 относятся к узлу 1 и процессоры 8-11 к узлу 3, все виртуальные процессоры гостя будут работать на процессорах 0-3, 8-11. Возможно сочетание этих обозначений с приведённым выше. Например, 1,node:2^6 означает, что все виртуальные процессоры гостя будут работать на процессоре 1 и на всех процессорах NUMA узла 2, но не на процессоре 6. Основываясь на примере, приведённом выше, это будут процессоры 1, 4, 5, 7.

«2», «3-8, ^5» — запросить определённый маппинг для VCPU. В данном примере VCPU 0 гостя будет работать на процессоре 2 хоста и VCPU 1 гостя будет работать на процессорах 3, 4, 6, 7, 8 хоста. Также могут быть использованы более сложные обозначения, как это описано выше.

Если эта опция не указана, то привязки VCPU к физическому ЦП не задаётся, и виртуальные процессоры гостя могут работать на всех ЦП хоста. Если эта опция указана, пересечение пиннинговой маски VCPU, приведённой здесь, с маской схожести ПО, приведённой через cpus_soft (если таковые имеются) используется для вычисления узловой привязки домена и для управления распределением памяти.

cpus_soft="CPU-LIST" — указывает на мягкую привязку, а не пиннинг, как в предыдущей команде. При использовании кредитного планировщика эта опция покажет, какие ЦП предпочитают процессоры VCPU домена. CPU-LIST определяется так же, как и в случае с cpus.

Если эта опция не указана, виртуальные процессоры гостя не будут иметь предпочтений при выборе того, какие ЦП должны работать. Если опция указана,

пересечение пиннинговой маски VCPU, приведённой здесь, с маской схожести ПО, приведённой через cpus (если таковые имеются), используется для вычисления узловой привязки домена и для управления распределением памяти.

Если не указаны ни cpus_soft=, ни cpus=, libxl автоматически попытается разместить домены на минимально возможным количестве узлов. Эвристический подход используется для выбора наилучшего узла (или набора узлов) с целью максимизации производительности для гостя и в то же время для достижения эффективного использования хост-процессоров и памяти. В этом случае мягкая привязка всех виртуальных ЦП домена будет установлена в pcpus, относящиеся к NUMA-узлам, выбранным при размещении.

5.3.2.2.2. Планирование ЦП

cpu_weight=WEIGHT — домен с весом 512 получит в два раза больше ЦП, чем домен с весом 256 на конфликтном хосте. Разрешенный вес находится в диапазоне от 1 до 65535; по умолчанию равен 256. Поддерживается планировщиками credit, credit2 и sedf.

cap=N — верхний предел, дополнительно фиксирует максимальное количество ЦП, которые домен сможет использовать, даже если хост-система имеет пустые циклы процессора. cap выражается в процентах одного физического процессора: 100 — один физический процессор, 50 — половина процессора, 400 — 4 процессора и т. д. Значение по умолчанию, 0, означает, что нет верхнего предела. Поддерживается планировщиками credit и credit2. Многие системы обладают характеристиками, уменьшающими вычислительную мощность процессора, который используется не на 100%. Это может происходить как в ОС, так и уровнем ниже, в BIOS. Если cap установлен таким образом, что отдельные ядра загружены менее, чем на 100%, это может оказать влияние на производительность рабочей нагрузки сверх воздействия крышки. Например, если процессор работает на частоте 2 ГГц, и ВМ будет «закрыта» на 50%, система управления питанием может также уменьшить тактовую частоту до 1 ГГц; эффект будет таким, что ВМ будет получать 25% от доступной мощности (50% от 1 ГГц), а не 50% (50% от 2 ГГц).

period=NANOSECONDS — нормальное EDF-использование планирования в наносекундах. Каждый период домен получает процессорное время, определенное в фрагменты. Поддерживается планировщиком sedf.

slice=NANOSECONDS — нормальное EDF-использование планирования в наносекундах. Определяет время, получаемое доменом каждый период времени. Поддерживается планировщиком sedf.

atency=N — пересчитанный период, если домен выполняет интенсивный ввод-вывод. Поддерживается планировщиком sedf.

extratime=BOOLEAN — флаг позволяет запустить домен в дополнительное время. Поддерживается планировщиком sedf.

5.3.2.2.3. Распределение памяти

memory=MBYTES — запустить домен с выделенными MBYTES оперативной памяти.

maxmem=MBYTES — максимальный объем памяти, видимый для гостя. Значение maxmem= должно быть равно или больше, чем memory=.

В сочетании с memory= эта опция запустит гостя с зарезервированным объемом памяти (pre-ballooned), если значения memory= и maxmem= различаются. HVM-гостю с зарезервированным объемом памяти необходим драйвер balloon — без него произойдёт сбой.

5.3.2.2.4. События

on_poweroff="ACTION" — действия домена при отключении.

Действия могут быть следующими:

- destroy — уничтожить домен;
- restart — уничтожить домен и создать новый домен с такой же конфигурацией;
- rename-restart — переименовать домен, который перестал работать, и создать новый домен с той же конфигурацией;
- preserve — сохранить домен. Он может быть проверен и позже уничтожен с помощью опции xl destroy.

- coredump-destroy — записать дамп ядра домена в /var/xen/ dump/NAME и затем уничтожить домен.
- coredump-restart — записать дамп ядра домена в /var/xen/ dump/NAME и затем перезапустить домен.

Значение по умолчанию: destroy.

on_reboot="ACTION" — действие при отключении домена с указанием кода причины, требующей перезагрузки. Значение по умолчанию: restart.

on_watchdog="ACTION" — действие при выключении домена по тайм-ауту менеджера виртуальных машин. Значение по умолчанию: destroy.

on_crash="ACTION" — действие при сбое работы домена. Значение по умолчанию: destroy.

5.3.2.2.5. Прямая загрузка ядра

Прямая загрузка ядра разрешает непосредственную загрузку ядра Linux и начального образа файловой системы initrd, расположенных в файловой системе хоста, что позволяет передавать аргументы командной строки напрямую. Прямая загрузка ядра поддерживается в PV-режиме. Поддержка прямой загрузки ядра в HVM-режиме ограничена (поддерживается при использовании qemu-xen и BIOS по умолчанию — seabios; не поддерживается в случае stubdom-dm и старых rombios).

kernel="PATHNAME" — загрузить указанный файл как образ ядра.

ramdisk="PATHNAME" — загрузить указанный файл как ram disk.

cmdline="STRING" — добавить STRING к командной строке ядра. Значение данной опции зависит от конкретного гостя. Эта опция является предпочтительной и может заменять root="STRING" плюс extra="STRING". Если cmdline="STRING" установлена, root="STRING" и extra="STRING" будут проигнорированы.

root="STRING" — добавить root="STRING" к командной строке ядра. Значение данной опции зависит от конкретного гостя.

extra="STRING" — добавить STRING к командной строке ядра. Значение данной опции зависит от конкретного гостя.

5.3.2.2.6. Дополнительные опции

uuid="UUID" — определяет UUID домена. Если опция не указана, будет создан новый уникальный UUID.

seclabel="LABEL" — назначить метку безопасности XSM для этого домена.

init_seclabel="LABEL" — указать метку безопасности XSM, временно используемую для этого домена во время его «сборки». XSM-метка домена будет изменена на исполняемую метку безопасности (заданную с помощью seclabel), как только создание домена завершится, перед началом активизации домена. При правильно построенной политике безопасности (например, nomigrate_t), можно использовать метку для создания домена, память которого недоступна для набора утилит домена.

nomigrate=BOOLEAN — отключить миграцию этого домена. Это даёт возможность задействовать некоторые определенные функции, которые несовместимы с миграцией. На текущий момент ограничено инвариантным флагом функции TSC команды cpuid, в случае, если TSC не эмулируется.

driver_domain=BOOLEAN — указать, что данный домен является ведущим доменом драйвера. Это задействует некоторые функции, необходимые для запуска домена драйвера.

5.3.2.2.7. Опции устройств

Следующие опции определяют, какие паравиртуальные, эмулированные и физические устройства будет содержать домен:

- disk=["DISK_SPEC_STRING", "DISK_SPEC_STRING", ...] — определяет диски (как эмулированные диски, так и виртуальные блочные устройства менеджера виртуальных машин), которые должны быть предоставлены гостю, и то, в каких объектах они должны отображаться;
- vif=["NET_SPEC_STRING", "NET_SPEC_STRING", …] определяет сетевое обеспечение (как эмулированные сетевые адаптеры, так и виртуальные интерфейсы менеджера виртуальных машин), которое должно быть

предоставлено гостю (подробная информация в файле docs/misc/xl-network-configuration.markdown);

- vfb=["VFB_SPEC_STRING", "VFB_SPEC_STRING", …] определяет паравиртуальные устройства фреймбуфера, которые должны предоставляться домену.

Эта опция не контролирует эмулируемую видеокарту, предоставленную HVM-гостю. Если опции Emulated VGA Graphics Device используются в конфигурации PV-доменов, xl подхватит vnc, vnclisten, vncpasswd, vncdisplay, vncunused, sdl, opengl и keymap для создания паравиртуального устройства кадрового буфера для доменов.

Каждый VFB_SPEC_STRING представляет собой разделённые запятой настройки KEY=VALUE из следующего списка:

- vnc=BOOLEAN — разрешить доступ к дисплею через протокол VNC. Это позволяет активировать другие настройки, относящиеся к VNC. По умолчанию разрешено;
- vnclisten="ADDRESS[:DISPLAYNUM]" — IP-адрес (и опционально VNC-номер дисплея). Если номер дисплея указан здесь, vncdisplay не потребуется;
- vncdisplay=DISPLAYNUM — номер дисплея VNC для использования. Фактическое число TCP-порта будет равно DISPLAYNUM+5900;
- vncunused=BOOLEAN — запросить VNC-поиск настройки VNC-дисплея для использования свободного порта TCP. Реально используемый дисплей может быть доступен с xl vncviewer;
- vncpasswd="PASSWORD" — пароль для VNC-сервера;
- sdl=BOOLEAN — указывает, что дисплей должен быть представлен через окно X (используя Simple DirectMedia Layer). По умолчанию этот режим не включён;
- opengl=BOOLEAN — обеспечивает OpenGL ускорение дисплея SDL. Работает только на машинах с device_model_version="qemu-xen-traditional" и только в том случае, если модель устройства обеспечивает поддержку OpenGL. По умолчанию эта опция отключена;

- keymap="LANG" — настройка раскладки для использования клавиатуры, относящейся к этому дисплею. Если метод ввода не поддерживает необработанные коды клавиш (например, при использовании VNC), эта опция позволит правильно преобразовать нажатия клавиш в корректные коды для гостевой системы. Конкретные допустимые значения определяются версией модели устройства. По умолчанию, en-us;
- channel=["CHANNEL_SPEC_STRING", "CHANNEL_SPEC_STRING", ...] — определяет виртуальные каналы, которые должны предоставляться гостю. Канал — это двунаправленный поток байтов с низкой пропускной способностью, который напоминает линию последовательной передачи данных. Типичные области применения для каналов включают в себя передачу конфигурации ВМ после загрузки и взаимодействие с программами внутри гостя (подробная информация в файле docs/ misc/channels.txt).

Каждый CHANNEL_SPEC_STRING представляет собой разделенные запятой настройки KEY=VALUE. Начальные и конечные пробелы игнорируются как в ключах, так и в значениях. Ни ключ, ни значение не могут содержать символы ',', '=' или "". Определяемые значения приведены далее:

- backend=DOMAIN — имя или ID back-end. Параметр не является обязательным. Если этот параметр опущен, будет применяться управляющий домен;
- name=NAME — имя строки для устройства. Параметр является обязательным. Рекомендуется использовать хорошо известное название для конкретных приложений (например, гостевого агента), для использования front end при подключении приложения к правильному канальному устройству. Не существует официального реестра названий каналов, поэтому авторам приложений рекомендуется создавать уникальные имена, включая доменное имя и номер версии в строке (например, org.mydomain.guestagent.1);
- connection=CONNECTION — указать, как будет реализован back end. connection=SOCKET — back end подключится к сокету домена Unix (на пути, указанном в path=PATH), вызывая, слушая и принимая соединения. Back end

будет функционировать как служба прокси между каналом и присоединенным сокетом. connection=PTY — back end создаст pty и proxy данных между каналом и главным устройством. Можно использовать команду xl channel-list, чтобы обнаружить назначенное ведомое устройство;

- pci=["PCI_SPEC_STRING", "PCI_SPEC_STRING", ...] — определяет PCI-устройство хоста для прямой передачи данному гостю. Каждый PCI_SPEC_STRING имеет форму [DDDD:]BB:DD.F[@VSLOT], KEY=VALUE, KEY=VALUE, ... где:

- DDDD:BB:DD.F — синтаксис определяет устройство PCI с точки зрения хоста в домене (DDDD), шины (BB), устройства (DD) и функции (F). Та же схема используется в выводе lspci для рассматриваемого устройства. По умолчанию lspci опускает домен (DDDD), если он равен нулю; здесь этот параметр (DDDD) тоже не является обязательным. Можно указать функцию (F) с помощью *, чтобы обозначить все функции;

- @VSLOT — определяет виртуальное устройство, с которого гостю будет видно данное устройство. Эквивалентно DD, которое видит домен. Внутри гостя DDDD и BB 0000:00;

- permissive=BOOLEAN — по умолчанию, pciback позволяет только PV-гостям записать известные безопасные значения в пространство конфигурации PCI. Многие устройства требуют записи в другие области пространства конфигурации для того, чтобы работать должным образом. Эта опция сообщает pciback- драйверу, что можно разрешить любую запись в пространство конфигурации PCI данного устройства этому домену. Опцию стоит включать с осторожностью: она дает гостю больше контроля над устройством, что может иметь последствия для безопасности или стабильности работы устройства. Рекомендуется включать эту опцию только для доверенных виртуальных машин под управлением администратора. Только для PV-доменов;

- msitranslate=BOOLEAN — указывает, что трансляция MSI-INTx должна быть включена для устройства PCI. При включении этой опции трансляция

MSI-INTx всегда разрешена MSI на устройстве PCI независимо от того, использует ли домен INTx или MSI. Некоторые драйвера устройств, такие как NVIDIA, обнаруживают несоответствия и не работают, если эта опция включена. Поэтому значение по умолчанию является ложным (0);

- seize=BOOLEAN — заставляет xl автоматически попытаться назначать устройство для pciback повторно, если оно ещё не назначено. Если опция установлена, xl тут же переназначит критическое устройство системы, такое как сеть или контроллер диска, используемое Dom0, без подтверждения;
- power_mgmt=BOOLEAN — указывает, что ВМ должна иметь возможность программировать состояния управления питанием D0-D3 для устройства PCI. Ложно (0) по умолчанию. Только для HVM-доменов;
- pci_permissive=BOOLEAN — изменяет значение по умолчанию permissive для всех устройств PCI, переданных этой ВМ. Только для PV-доменов;
- pci_msitranslate=BOOLEAN — изменяет значение по умолчанию msitranslate для всех устройств PCI, переданных этой ВМ;
- pci_seize=BOOLEAN — изменяет значение по умолчанию seize для всех устройств PCI, переданных этой ВМ;
- pci_power_mgmt=BOOLEAN — изменяет значения по умолчанию power_mgmt для всех устройств PCI, переданных к этой ВМ. Только для HVM-доменов;
- gfx_passthru=BOOLEAN — включить прямой доступ к графическим устройствам PCI. Эта опция делает назначенную PCI-видеокарту базовой (исходной) видеокартой в ВМ. Эмулированный графический адаптер QEMU в данном случае отключён, а консоль VNC для ВМ не будет иметь графического вывода. Вся графика, в том числе сообщения о времени загрузки QEMU BIOS из виртуальной машины, будет выводиться через переданную физическую видеокарту.

PCI-устройство видеокарты для передачи выбирается опцией PCI, точно так же, как выполняется транзит/назначение обычных PCI-устройств. gfx_passthru не позволяет совместно использовать GPU: можно назначить графический процессор

только для одной ВМ в данный момент времени. gfx_passthru также позволяет передавать различные диапазоны устаревшей памяти VGA, BAR, MMIOs и ioports к ВМ, поскольку они необходимы для правильной работы VGA BIOS, текстового режима, VBE и т. д.

Включение опции gfx_passthru также копирует BIOS физической видеокарты в память гостя и выполняет VBIOS в госте для инициализации видеокарты. Большинство графических адаптеров требуют специфичных для производителя настроек для правильной работы графической подсистемы.

gfx_passthru в настоящее время поддерживается только традиционной моделью устройств QEMU. Старшие версии модели устройств QEMU не поддерживают gfx_passthru.

Некоторые графические адаптеры (AMD / ATI карты) не требуют опции gfx_passthru: можно использовать обычный транзит PCI для назначения видеокарты в качестве вторичной видеокарты на виртуальной машине. Эмулированная QEMU-видеокарта остаётся основной видеокартой, а вывод VNC доступен в эмулированном QEMU основном адаптере.

ioports=["IOPORT_RANGE", "IOPORT_RANGE", ...] — разрешить гостевой доступ к конкретным традиционным портам ввода/вывода. Каждый IOPORT_RANGE дается в шестнадцатеричном виде и может задавать или диапазон (например, 2f8-2ff включительно), или один порт ввода/вывода (например, 2F8).

Рекомендуется использовать эту опцию только для доверенных виртуальных машин под управлением администратора.

iomem=["IOMEM_START,NUM_PAGES[@GFN]", "IOMEM_START, NUM_PAGES[@GFN]", ...] — разрешить автоматически транслированным доменам доступ к страницам памяти ввода-вывода аппаратного обеспечения.

IOMEM_START — физический номер страницы. NUM_PAGES — количество страниц, начиная со START_PAGE для предоставления доступа. GFN — номер кадра доменов, где начнется трансляция в адресном пространстве домена DomU. Если GFN не указан, маппинг будет производиться с помощью IOMEM_START как начало в адресном пространстве домена DomU, поэтому трансляция будет 1:1 по

умолчанию. Все значения должны быть приведены в шестнадцатеричном формате. IOMMU не будет обновляться с отображениями, указанными с этой опцией. Данная опция не должна использоваться для транзитной передачи какого-либо IOMMU-защищённого устройства. Рекомендуется использовать эту опцию только для доверенных виртуальных машин под управлением администратора.

irqs=[NUMBER, NUMBER, ...] — разрешить гостевой доступ к определенным физическим прерываниям (IRQ). Рекомендуется использовать эту опцию только для доверенных виртуальных машин под управлением администратора.

max_event_channels=N — ограничить гостя в использовании максимального количества событий каналов, равного N (PV-прерываний). Гости пользуются ресурсами менеджера виртуальных машин для каждого канала событий, который они используют. По умолчанию, значение 1023 должно быть достаточным для обычных доменов. Максимальное значение зависит от того, что поддерживает домен. Гости, поддерживающие каналы событий на основе FIFO ABI, поддерживают до 131071 каналов событий. Другие гости ограничены значением 4095 (64-разрядная x86 и ARM) или 1023 (32-разрядная x86).

5.3.2.2.8. Особые опции PV-гостя

Следующие опции подходят только для паравиртуализированных доменов:

- bootloader="PROGRAM" — запуск программы с целью поиска образа ядра и RAM-диска для использования. Обычно программа использует pygrub, что является эмуляцией grub/grub2/syslinux. Для PV-доменов должно быть указано или ядро (kernel), или загрузчик (bootloader);
- bootloader_args=["ARG", "ARG", ...] — добавить аргументы (ARG) к аргументам в программе загрузчика (bootloader). Если аргумент представляет собой простую строку, она будет разделена на слова пробелами (рекомендуется именно такой вариант);
- e82Q_host=BOOLEAN — выбирает, демонстрировать ли хост e820 (карта памяти) гостю с помощью виртуальной e820. Когда для этой опции установлено значение ложь (0), гостевое псевдофизическое адресное

пространство состоит из одного непрерывного диапазона памяти. Когда эта опция используется, виртуальный e820 отображает хост e820 и содержит те же области адресного пространства PCI. Общее количество RAM, представленное картой памяти, всегда одно и то же; эта опция настраивает только то, как оно распределено. Отображение e820 гостю даёт гостевому ядру возможность отложить необходимую часть псевдофизического адресного пространства в целях обеспечения адресного пространства для отображения переданных устройств PCI. Нужна ли эта опция, зависит от гостевой ОС; в частности, она необходима при использовании современного ядра Linux (pvops). Эта опция по умолчанию выставлена на true (1), если есть любые передаваемые PCI устройства, и на false (0) — в противном случае. Если не настраивается передача каких-либо устройств во время создания домена, но hotplug-устройства будут использованы позднее, следует установить эту опцию;

- pvh=BOOLEAN — выбирает, следует ли запустить данного PV- гостя с использованием аппаратной поддержки HVM. Значение по умолчанию: 0.

5.3.2.2.9. Особые опции HVM-гостя

Следующие опции применимы только к HVM-гостям.

5.3.2.2.10. Устройство загрузки

boot=[c\d\n] — выбирает эмулируемое виртуальное устройство для загрузки. Вариантами здесь являются жёсткий диск (c), cd-rom (d) или network/PXE (n). Может быть задано несколько вариантов; они будут приняты в том порядке, в каком были указаны. Например, dc для загрузки с компакт-диска, но с резервом на жёстком диске. Значение по умолчанию: cd.

5.3.2.2.11. Пейджинг

Следующие опции управляют механизмами, используемыми для виртуализации гостевой памяти. Значения по умолчанию выбраны таким образом, чтобы получить лучшие результаты для общих случаев. Как правило, следует оставить эти опции без изменений.

Перечень опций:

- hap=BOOLEAN — включает или выключает аппаратно поддерживаемый пейджинг (использование аппаратного вложенного страничного элемента таблицы). Выполняет функцию EPT, также расширенные таблицы страниц) в процессорах Intel и функцию NPT (англ. Nested Page Tables, также вложенные таблицы страниц) или RVI в процессорах AMD. Если опцию отключить, менеджер виртуальных машин запустит «гостя» в режиме «таблицы теневых страниц», где обновления таблицы страниц гостя и/или флеш TLB будут эмулированы. Использование HAP предусмотрено по умолчанию, когда доступно;
- oos=BOOLEAN — включает или выключает несинхронизованные таблицы памяти. При работе в режиме теневых таблиц страниц обновление таблицы страниц гостя может быть отложено, как указано в архитектуре руководства Intel/AMD. Это может привести к неожиданным ошибкам в госте или нахождению ошибок в менеджере виртуальных машин: можно отключить эту функцию. Использование несинхронизованных таблиц памяти в случаях, когда менеджер виртуальных машин считает это целесообразным, является значением по умолчанию;
- shadow_memory=MBYTES — посчитать количество мегабайт, отведённое для теневого копирования гостевых таблиц страниц (эффективно действующее в качестве кэш-памяти переведённых страниц), или использовать для состояния HAP. По умолчанию, это 1 МБ на VCPU гостя плюс 8 Кбайт на 1 Мбайт гостевой оперативной памяти. Как правило, не возникает необходимости менять это значение. Однако, при использовании аппаратно поддерживаемого пейджинга (например, в теневом режиме) и гостевой рабочей нагрузки, состоящей из очень большого количества подобных процессов, увеличение этого значения может повысить производительность.

5.3.2.2.12. Ключевые характеристики процессора и платформы

Следующие опции позволяют различным функциям на уровне процессора и платформы быть скрытыми или отображаться для гостя. Опции используются в работе старых гостевых ОС, которые могут вести себя неправильно при выполнении современных функций. Рекомендуется принять значения по умолчанию для этих параметров там, где это возможно.

bios="STRING" — выбрать виртуальную прошивку BIOS, которая будет показана гостю. По умолчанию, предположение делается на основе модели устройства.

rombios — загружает ROMBIOS, 16-разрядный x86- совместимый BIOS. Используется по умолчанию при device_model_version=qemu-xen-traditional. Это единственная опция BIOS, поддерживаемая при device_model_version=qemu-xen-traditional.

seabios — загружает SeaBIOS, 16-разрядный x86- совместимый BIOS. Это используется по умолчанию с device_model_version=qemu-xen.

ovmf — загружает OVMF, стандартную прошивку UEFI по проекту Tianocore. Требуется device_model_version=qemu-xen.

pae=BOOLEAN — скрыть или отобразить IA32 расширения физических адресов. Эти расширения позволяют 32-разрядным гостевым ОС получить доступ к более чем 4 Гб оперативной памяти. Включение PAE также включает другие функции, такие как NX. PAE необходим для запуска 64-разрядных гостевых ОС. Рекомендуется оставить опцию включённой и разрешить гостевой операционной системе выбрать, следует ли использовать PAE. Поддерживается только в x86 ОС.

acpi=BOOLEAN — отображает спецификацию ACPI (усовершенствованный интерфейс конфигурирования системы и управления энергопитанием) таблиц из виртуальной прошивки на гостевой ОС. Поддержку ACPI требует большинство современных гостевых ОС. Эта опция включена по умолчанию. Может потребоваться отключить ACPI для совместимости с некоторыми гостевыми ОС.

acpi_s3=BOOLEAN — включить S3 (suspend-to-ram) состояние в таблицу виртуальной прошивки ACPI. Значение по умолчанию 1.

acpi_s4=BOOLEAN — включить S4 (suspend-to-disk) состояние в таблицу виртуальной прошивки ACPI. Значение по умолчанию: 1.

apic=BOOLEAN — включить информацию, касающуюся APIC (англ. Advanced Programmable Interrupt Controller — расширенный программируемый контроллер прерываний) в таблицах прошивки / BIOS, на одном гостевом процессоре. Это приводит к экспорту MP (многопроцессорные) и PIR (PCI Маршрутизация прерываний) таблиц виртуальной прошивкой. Опция не оказывает никакого эффекта на гостя с несколькими виртуальными процессорами, поскольку они всегда должны включать эти таблицы. Опция включена по умолчанию. Может потребоваться отключить эти таблицы прошивки при использовании определённых старых гостевых ОС. Таблицы были заменены на более новые конструкции в таблицах ACPI. Поддерживается только в x86 ОС.

nx=BOOLEAN — скрыть или отобразить возможности No-eXecute. Позволяет гостевой ОС помечать страницы памяти так, чтобы они не могли быть выполнены. Эта опция требует, чтобы PAE была включена. Поддерживается только в x86 ОС.

hpet=BOOLEAN — включает или отключает HPET (англ. High Precision Event Timer, высокоточный таймер событий). Опция включена по умолчанию. Может потребоваться отключить HPET в целях улучшения совместимости с гостевой ОС. Поддерживается только в x86 ОС.

nestedhvm=BOOLEAN — включение или отключение функции предоставления гостевого доступа к возможностям аппаратной виртуализации, что позволяет, например, гостевой ОС также функционировать в качестве менеджера виртуальных машин. Опция по умолчанию отключена.

cpuid="LIBxl_STRING" — настроить значение, возвращаемое при выполнении гостем инструкции CPUID. Синтаксис libxl — это список разделенных запятой пар ключ=значение, которому предшествует слово host. Несколько ключей принимают числовое значение, все остальные принимают единственный символ, который описывает, что делать со служебным битом.

Возможные значения для отдельного служебного бита:

- '1' -> заставить соответствующий бит 1;

- '0' -> принудить к 0;
- 'x' -> получить безопасное значение (транзит и маска с политикой по умолчанию);
- 'k' - > пройти через битное значение хоста;
- 's' -> как 'k', но сохранить через сохранить/восстановить и миграцию (не реализовано).

Список ключей, принимающих значения: apicidsize, brandid, clflush, family, localapicid, maxleaf, maxhvleaf, model, nc, proccount, procpkg, stepping. Список ключей, принимающих символы: 3dnow, 3dnowext, 3dnowprefetch, abm, acpi, aes, altmovcr8, apic, avx, clfsh, cmov, cmplegacy, cmpxchg16, cmpxchg8, cntxid, dca, de, ds, dscpl, dtes64, est, extapic, f16c, ffxsr, fma4, fpu, fxsr, htt, hypervisor, ia64, ibs, lahfsahf, lm, lwp, mca, mce, misalignsse, mmx, mmxext, monitor, movbe, msr, mtrr, nodeid, nx, osvw, osxsave, pae, pagelgb, pat, pbe, pclmulqdq, pdcm, pge, popcnt, pse, pse36, psn, rdtscp, skinit, smx, ss, sse, sse2, sse3, sse4_l, sse4_2, sse4a, ssse3, svm, svm_decode, svm_lbrv, svm_npt, svm_nrips, svm_pausefilt, svm_tscrate, svm_vmcbclean, syscall, sysenter, tbm, tm, tm2, topoext, tsc, vme, vmx, wdt, x2apic, xop, xsave, xtpr.

Более подробную информацию об инструкции к CPUID можно найти в описании к процессору.

acpi_firmware="STRING" — указать путь к файлу, который содержит дополнительные таблицы прошивки ACPI для прохода к гостю. Файл может содержать несколько таблиц в двоичной AML-форме, объединённых вместе. Каждая таблица самостоятельно описывает свою длину; дополнительной информации не требуется. Эти таблицы будут добавлены к таблице ACPI, установленной в госте. Существующие таблицы не могут быть отменены этой функцией. Например, нельзя использовать эту функцию для переопределения таких таблиц, как DSDT и FADT.

smbios_firmware="STRING" — указать путь к файлу, который содержит дополнительные структуры прошивки SMBIOS для прохода к гостю. Файл может содержать набор предопределенных DMTF- структур, которые позволят изменить внутренние настройки по умолчанию. Не все предопределенные структуры могут быть отменены, только следующие типы: 0, 1, 2, 3, 11, 22, 39. Файл может содержать

любое количество определённых поставщиком SMBIOS структур (типа 128 - 255). Поскольку SMBIOS структуры не демонстрируют свой полный размер, каждой записи в файле должно предшествовать 32b целое число, указывающее на размер следующей структуры.

ms_vm_genid="OPTION" — предоставить гостю VM generation ID. VM generation ID — это 128-битное случайное число, которое используется гостем для определения, был ли домен был восстановлен из предыдущего снимка или клонирован. Опция необходима для Microsoft Windows Server 2012 (и более поздних) контроллеров домена. acpi_firmware="STRING" — указать путь к файлу, который содержит дополнительные таблицы прошивки ACPI для прохода к гостю. Файл может содержать несколько таблиц в двоичной AML-форме, объединённых вместе. Каждая таблица самостоятельно описывает свою длину; дополнительной информации не требуется. Эти таблицы будут добавлены к таблице ACPI, установленной в госте. Существующие таблицы не могут быть отменены этой функцией. Например, нельзя использовать эту функцию для переопределения таких таблиц, как DSDT, FADT и т. д.

Допустимые варианты:

- generate — генерировать случайный VM generation ID каждый раз при создании или восстановлении домена;
- none — не предоставлять VM generation ID.

5.3.2.2.13. Механизмы управления виртуальным временем гостя

tsc_mode="MODE" — определяет, как TSC (англ. Time Stamp Counter) должен быть представлен гостю. Установка этой опции в виде числа устарела. Поддерживается только в x86 ОС.

Варианты MODE:

- default — режим по умолчанию: домен rdtsc/p выполняется с гарантированной монотонностью работы, в частности с помощью эмуляции (с частотой, масштабируемой при необходимости);

- always_emulate — режим, в котором домен rdtsc/ p всегда эмулируется с частотой 1 ГГц. Домен всегда эмулируется. Виртуальные кванты времени выглядят для гостя увеличивающимися на фиксированной скорости 1 ГГц, независимо от скорости физического ЦП или состояния питания. Дополнительные затраты, связанные с эмуляцией, не влияют на основную производительность процессора;
- native — режим, в котором домен rdtsc всегда выполняется без гарантий монотонности/частоты; домен rdtscp эмулируется в исходной частоте, если не поддерживается аппаратно;
- native_paravirt — аналогично режиму native, за исключением того, что менеджер виртуальных машин управляет регистром TSC_AUX, чтобы домен мог определить, когда произошло восстановление/миграция, и предполагает, что домен получает/ использует механизм, подобный pvclock, для корректировки монотонности и частоты изменений (подробное описание в файле доступно в файле docs/misc/tscmode.txt).

localtime=BOOLEAN — установить часы реального времени на местное время или время по Гринвичу. Значение по умолчанию: 0 — значение по Гринвичу.

rtc_timeoffset=SECONDS — установить часы реального времени с отступом (задержкой) в секундах. Значение по умолчанию: 0.

vpt_align=BOOLEAN — указывает, что периодические таймеры виртуальной платформы должны быть приведены в соответствие, чтобы уменьшить прерывания доменов. Включение этой опции позволяет снизить потребление энергии, особенно, если домен использует значения высокого таймера частоты прерываний (Гц). Значение по умолчанию: 1.

timer_mode="MODE" — режим виртуальных таймеров.

Варианты MODE:

- delay_for_missed_ticks — приостановить пропущенные кванты времени. Не продвигать (не опережать) время VCPU за рамки правильного времени доставки для прерываний, которое было упущено из-за вытеснения. Доставить

пропущенные прерывания при перепланировании VCPU и продвинуть виртуальное время VCPU поэтапно для каждого из них;

- no_delay_for_missed_ticks — не приостанавливать пропущенные кванты времени. Пропущенные прерывания будут доставлены, но при этом гостевое время всегда отслеживает стандартное (реальное) время;

- no_missed_ticks_pending — пропущенные прерывания не находятся в режиме ожидания. Вместо этого обеспечивается доставка квантов времени на какой-либо ненулевой скорости; если обнаружены пропущенные кванты и VCPU вытесняется в течение следующего периода, флаг обработки не отключается;

- one_missed_tick_pending — один пропущенный квант в режиме ожидания. Пропущенные прерывания собираются вместе и доставляются как один «опоздавший квант». Гостевое время всегда отслеживает стандартное (реальное) время.

5.3.2.2.14. Распределение памяти

mmio_hole=MBYTES — определяет размер разрыва для MMIO ниже 4 Гбайт. Действительно только для device_model_version = "qemu-xen". Не может быть меньше 256. Не может быть больше 3840. Довольно распространённое большое значение равно 3072.

5.3.2.2.15. Поддержка паравиртуализации

Следующие опции позволяют паравиртуализированным ключевым характеристикам (например, устройствам) отображаться в ОС HVM-гостя, что улучшает производительность:

xen_platform_pci=BOOLEAN — включить или отключить PCI- устройство платформы менеджера виртуальных машин. Наличие этого виртуального устройства позволяет гостевой ОС (при условии наличия подходящих драйверов) использовать функции паравиртуализации, такие как дисковые и сетевые устройства. Включение этих драйверов повышает производительность и настоятельно рекомендуется при их

наличии. Установка xen_platform_pci=0 со значением по умолчанию device_model "qemu-xen" требует QEMU 1.6 и выше.

viridian=["GROUP", "GROUP", ...] — группы расширений Microsoft Hyper-V (Viridian), отображаемые для гостя. Варианты GROUP:

- base — группа включает в себя гипервызовы MSR, индекс виртуального процессора MSR и доступ APIC MSR. Эти расширения могут улучшить производительность Windows Vista и Windows Server 2008 года, поэтому настоятельно рекомендуется установить эту опцию для таких доменов. Эта группа также является необходимой базой для всех остальных;
- freq — группа включает в себя TSC и APIC frequency MSRs. Эти расширения могут улучшить производительность Windows 7 и Windows Server 2008 R2 года и более поздних версий;
- time_ref_count — группа включает в себя Partition Time Reference Counter MSR. Это расширение может улучшить производительность Windows 8 и Windows Server 2012 года и более поздних версий;
- all — специальное значение для использования всех доступных групп.

Группы могут быть отключены при добавлении к имени префикса !. Так, например, чтобы задействовать все группы, кроме freq, укажите: viridian=["all", "! freq"]. Также, Viridian-опция может быть указана как boolean. Значение истина (1) эквивалентно списку ["defaults"], а значение ложь (0) эквивалентно пустому списку.

5.3.2.2.16. Эмулированные графические устройства VGA

Следующие опции управляют ключевыми характеристиками эмулированных графических устройств. Многие из этих опций подобны эквивалентному ключу в VFB_SPEC_STRING для конфигурирования виртуального устройства буфера кадров.

videoram=MBYTES — количество оперативной памяти, которое будет доступно эмулированной видеокарте, что, в свою очередь, ограничивает доступное разрешение и глубину цвета.

При использовании традиционной модели устройств QEMU значением по умолчанию, а также минимальным количеством видеопамяти для stdvga, являются 8 Мбайт, что вполне достаточно для, например, 1600x1200 на 32bpp. Для более поздних традиционных моделей устройств QEMU значение по умолчанию и минимум равны 16 Мбайт.

При использовании эмулированной видеокарты Cirrus (vga="cirrus") и традиционной модели устройств QEMU объем видеопамяти фиксируется на 4 Мбайт, что достаточно для 1024x768 при 32 bpp. Для более поздних моделей устройств QEMU значение по умолчанию и минимум равны 16 Мбайт.

stdvga=BOOLEAN — выбрать стандартную VGA-карту с VBE (англ. VESA BIOS Extensions) в качестве эмулированного графического устройства. По умолчанию является ложным (0), что означает эмуляцию видеокарты Cirrus Logic GD5446. Если ОС домена поддерживает VBE 2.0 или более позднюю версию (например, Windows XP и выше), рекомендуется включить эту опцию.

vga="STRING" — выбрать эмулированную видеокарту (none\ stdvga\cirrus). Значение по умолчанию: cirrus.

vnc=BOOLEAN — разрешить доступ к дисплею через VNC-протокол. Это активирует другие настройки VNC. По умолчанию включено.

vnclisten="ADDRESS[:DISPLAYNUM]" — IP-адрес и, если требуется, номер дисплея VNC.

vncdisplay=DISPLAYNUM — определяет номер дисплея VNC. Фактическое число TCP порта будет DISPLAYNUM+5900.

vncunused=BOOLEAN — запрашивает автоматическую настройку номера дисплея VNC, используя свободный порт TCP. Доступ к фактически используемому дисплею можно получить через xl vncviewer.

vncpasswd="PASSWORD" — пароль для сервера VNC.

keymap="LANG" — настройка раскладки для использования клавиатуры, относящейся к этому дисплею. Если метод ввода не поддерживает необработанные коды клавиш (например, при использовании VNC), эта опция позволит преобразовать нажатия клавиш в корректные коды для гостевой системы.

Конкретные допустимые значения определяются версией модели устройства. Значение по умолчанию: en-us.

sdl=BOOLEAN — указывает, что дисплей должен быть представлен через окно X SDL (англ. Simple DirectMedia Layer). Значение по умолчанию: не включать этот режим.

localtime=BOOLEAN — установить часы реального времени на местное время или время по Гринвичу. Значение по умолчанию: 0 — значение по Гринвичу.

opengl=BOOLEAN — активирует OpenGL ускорение дисплея SDL. Эффективно только для машин, использующих «device_model_version="qemu-xen-traditional"» и только тогда, когда модель устройства была собрана с поддержкой OpenGL. Значение по умолчанию: 0.

nographic=BOOLEAN — включить или отключить виртуальное графическое устройство. По умолчанию, графическое устройство VGA предоставляется, но можно использовать эту опцию, чтобы отключить его.

5.3.2.2.17. Поддержка графики SPICE

Следующие опции управляют ключевыми характеристиками SPICE.

spice=BOOLEAN — разрешить доступ к дисплею через протокол SPICE. Это позволяет использовать другие настройки SPICE.

spicehost="ADDRESS" — указать адрес интерфейса для прослушивания, если он дан, или любой другой интерфейс.

spiceport=NUMBER — указать порт для прослушивания на сервере SPICE, если SPICE включен.

spicetls_port=NUMBER — указать безопасный порт для прослушивания на сервере SPICE, если SPICE включён. Должен быть предоставлен по крайней мере один из spiceport или spicetls_port, если SPICE включён. Опции, связанные с spicetls_port, не поддерживаются.

spicedisable_ticketing=BOOLEAN — разрешить подключение клиента без пароля. При отключении должен быть установлен spicepasswd. Значение по умолчанию 0.

spicepasswd="PASSWORD" — указать ticket password, который используется клиентом для подключения.

spiceagent_mouse=BOOLEAN — указать, используется ли агент SPICE для клиентского режима работы мыши. Значение по умолчанию: 1.

spicevdagent=BOOLEAN — включает spice vdagent. Spice vdagent — это дополнительный компонент для повышения пользовательского опыта и выполнения задач управления, ориентированных на гостя. Его характеристики включают клиентский режим работы мыши (нет необходимости «захватывать» мышь клиентом, нет отставания мыши), автоматическую настройку разрешения экрана, копирование и вставку (текста и изображений) между клиентом и DomU. Также для работы требуется vdagent-услуги, установленные на ОС DomU. Значение по умолчанию: 0.

spice_clipboard_sharing=BOOLEAN — позволить совместное использование буфера обмена Spice (копировать/вставить). Требуется включение spicevdagent. Значение по умолчанию: 0.

spiceusbredirection=NUMBER — включить spice USB redirection. Создаёт число (NUMBER) USB-redirection каналов для перенаправления до 4 USB-устройств от spice-клиента к QEMU домена DomU. Требуется контроллер USB. В случае, если не определён конкретный контроллер, будет автоматически добавлен контроллер USB2. Значение по умолчанию 0.

5.3.2.2.18. Различное эмулированное аппаратное оборудование

serial=["DEVICE", "DEVICE", ...] — перенаправить виртуальные последовательные порты к устройствам (DEVICE) . По умолчанию, vc в графическом режиме и stdio, если используется nographics=1. Для обратной совместимости также принимается форма serial=DEVICE.

soundhw=DEVICE — выбрать виртуальную звуковую карту для гостя. Допустимые устройства определяются конфигурацией модели устройства. По умолчанию не экспортируется никакое звуковое устройство.

usb=BOOLEAN — включить или отключить эмуляцию USB-шин в госте.

usbversion=NUMBER — тип эмулированной USB-шины в госте. 1 для usb1, 2 для usb2 и 3 для usb3; доступно только для последней версии устройств QEMU. Из-за ограничений реализации несовместимо с параметрами USB и usbdevice. Значение по умолчанию: 0 (не определен контроллер USB).

usbdevice=["DEVICE", "DEVICE", ...] — добавляет устройства (DEVICE) к эмулируемой шине USB. Шина USB также должна быть включена с помощью usb=1. Наиболее распространенное использование этой опции — usbdevice=['tablet'], что добавляет указатель устройства, используя абсолютные координаты. Такие устройства работают лучше, чем относительные координаты устройства (например, стандартная мышь), поскольку многие методы экспорта гостевой графики (например, VNC) лучше работают в этом режиме. Это не зависит от фактического указателя устройства, который вы используете на хосте / на стороне клиента.

Хост-устройства могут быть также переданы подобным образом с указанием хоста: USBID, который имеет вид xxxx:yyyy. USBID можно выявить с помощью lsusb или USB-устройств. Для использования формата host:bus.addr необходимо удалить все ведущие '0' в шине и адресе. Например, для устройства USB на шине 008 устройства 002, host:8.2. Форма uusbdevice=DEVICEтакже используется для совместимости в обратном направлении.

vendor_device="VENDOR_DEVICE" — выбирает, какой вариант QEMU Xen-pvdevice следует использовать для этого гостя. Допустимые значения:

- none — Xen-pvdevice следует пропустить. Значение по умолчанию;
- xenserver — будет указан xenserver-вариант Xen-pvdevice (device-id=COOO), позволяющий использовать PV-драйверы XenServer в госте.

Этот параметр вступает в силу только при device_model_version=qemu-xen (подробное описание доступно в файле docs/misc/pci-device-reservations.txt).

5.3.2.2.19. Опции для моделей устройств

Следующие опции управляют выбором модели устройства. Компонент обеспечивает эмуляцию виртуальных устройств для HVM-гостя. Для PV- гостя

модель устройства иногда используется для обеспечения back end для некоторых PV-устройств (чаще всего для виртуального устройства кадрового буфера).

device_model_version="DEVICE-MODEL" — выбирает, какой вариант модели устройства следует использовать для этого гостя. Допустимые значения:

- qemu-xen — использовать модели устройств, объединенные в вышестоящие версии проекта QEMU. Эта модель устройства выбрана по умолчанию для Linux Dom0;
- qemu-xen-traditional — использовать модель устройства, основанную на «вилке» QEMU менеджера виртуальных машин. Эта модель устройства выбрана по умолчанию для NetBSD Dom0. Рекомендуется принять значение по умолчанию для новых доменов.

device_model_override="PATH" — переопределение пути в двоичную систему для использования в качестве модели устройства. Двоичная система, представленная здесь, должна быть согласована с указанной device_model_version. Как правило, не требуется указывать эту опцию.

device_model_stubdomain_override=BOOLEAN — override- использование модели устройства на основе stub-домена. Как правило, эта опция автоматически выбирается на основе других характеристик и параметров, выбранных ранее.

device_model_stubdomain_seclabel="LABEL" — назначить метки безопасности XSM stub-домену.

device_model_args=["ARG", "ARG", ...] — передать дополнительные произвольные параметры командной строке модели устройства. Каждый элемент в списке передаётся в качестве опции к модели устройства.

device_model_args_pv=["ARG", "ARG", ...] — передать дополнительные произвольные параметры командной строке только PV-модели устройства. Каждый элемент в списке передается в качестве опции к модели устройства.

device_model_args_hvm=["ARG", "ARG", …] — передать дополнительные произвольные параметры командной строке только HVM-модели устройства. Каждый элемент в списке передаётся в качестве опции к модели устройства.

5.3.2.2.20. Сочетания клавиш

Доступные сочетания клавиш определяются моделью используемого устройства. Обычно применяются следующие сочетания:

ar de-ch es fo fr-ca hu ja mk no pt-br sv da en-gb et fr

fr-ch is It nl pi ru th de en-us fi fr-be hr it lv nl-be pt si tr

Значения по умолчанию: en-us.

5.4. Конфигурация сети в xl.cfg

5.4.1. Опции конфигурационного файла

В данном разделе описаны опции конфигурационного файла xl для конфигурации VIF-устройств. Конфигурация имеет следующую форму:

vif = ['<vifspec>', '<vifspec>', …]

где каждый vifspec представлен в форме [<key>=<value>\ <flag>,].

Пример конфигурации.

mac=00:16:3E:74:3d:76,model=rtl8139,bridge=xenbr0'
'mac=00:16:3E:74:34:32"

В конфигурационном файле домена эти MAC-адреса указываются следующим образом:

vif = ['mac=00:16:3E:74:34:32',
'mac=00:16:3e:5f:48:e4,bridge=xenbr1']

Более формально строка представляет собой последовательность разделённых запятой пар ключ=значение. Все ключевые слова не являются обязательными.

У каждого устройства есть DEVID, который является его индексом в списке VIF-устройств, начиная с 0.

5.4.2. Ключевые слова

5.4.2.1. MAC-адрес

Ключевое слово mac задаёт MAC-адрес гостя данного VIF-устройства. Значение является 48-разрядным числом, представленным в виде шести групп двух шестнадцатеричных чисел, разделённых двоеточием.

По умолчанию, если это ключевое слово не указано, MAC-адрес автоматически генерируется внутри пространства, предназначенного для уникального идентификатора организации OUI (00:16:3e).

При выборе MAC-адреса рекомендуется придерживаться одного из следующих алгоритмов.

5.4.2.1.1. Алгоритм 1

Алгоритм 1 выглядит следующим образом:

1) создайте случайную последовательность из 6 байт;

2) установите локально управляемый бит (бит 2 первого байта);

3) очистите бит многоадресной передачи (бит 1 первого байта). Первый байт должен иметь битовый паттерн xxxxxxlO (x — бит, созданный произвольно — random), а остальные 5 байт генерируются случайным образом.

5.4.2.1.2. Алгоритм 2

Выделите адрес внутри пространства, ограниченного OUI, следуя при этом указаниям вашей организации.

5.4.2.1.3. Алгоритм 3

Выделите адрес внутри пространства, ограниченного OUI. Следует избегать конфликта с другими пользователями сегмента физической сети, в котором будет располагаться VIF-устройство.

При наличии OUI рекомендуется использовать его. В противном случае рекомендуется создать случайный MAC-адрес и установить локально управляемый бит: это позволит генерировать последовательность битов в более произвольном порядке, чем при использовании OUI.

5.4.2.2. Сетевой мост

Ключевое слово bridge задает имя сетевого моста, к которому должен быть добавлено VIF-устройство. По умолчанию, xenbr0. Мост должен быть настроен с помощью инструментов конфигурации сети имеющегося дистрибутива.

5.4.2.3. Шлюз

Ключевое слово gateway задает имя сетевого интерфейса, которому назначен IP-адрес и который находится в сети, с которой должно взаимодействовать VIF-устройство. Параметр используется в хосте с помощью скрипта горячего подключения vif-route.

5.4.2.4. Тип

Ключевое слово type определяет тип устройства. Действительно только для HVM-гостей. Допустимые значения:

- ioemu (по умолчанию) — устройство будет предоставлено гостю в качестве эмулируемого устройства, а также в качестве паравиртуализированного устройства, которое гость может выбрать для использования при наличии подходящих драйверов;
- vif — устройство будет предоставлено только в качестве паравиртуализированного.

5.4.2.5. Модель

Ключевое слово model действительно для гостевых HVM-устройств только с type=ioemu.

Определяет тип устройства для эмуляции для данного гостя. Допустимые значения:

- rtl8139 (по умолчанию) — Realtek RTL8139;
- elQQQ — Intel E1000;
- другая модель, поддерживаемая устройством.

5.4.2.6. Имя back-end устройства

Ключевое слово vifname определяет имя back-end устройства для виртуального устройства. Для HVM-доменов связанное эмулированное устройство типа tap будет иметь суффикс -emu.

По умолчанию, именем виртуального устройства является vifDOMID.DEVID, а имя для типа tap — vifDOMID.DEVID-emu.

5.4.2.7. Скрипт

Ключевое слово script указывает скрипт запуска горячего подключения для настройки VIF-устройств (например, чтобы добавить его в соответствующий мост). По умолчанию, находится в XEN_SCRIPT_DIR/vif-bridge.

5.4.2.8. IP-адрес

Ключевое слово ip указывает IP-адрес устройства. По умолчанию, IP- адрес не указан. Как правило, в результате указания адреса меняются правила межсетевой защиты: можно разрешить гостю использовать только указанный IP-адрес и заблокировать остальные.

5.4.2.9. Back-end домен

Ключевое слово backend определяет back-end домен, к которому подсоединяется данное VIF-устройство. По умолчанию, Dom0. Указание другого домена требует создания отдельного домена для драйвера.

5.4.2.10. Скорость

Ключевое слово rate указывает скорость, при которой исходящий трафик будет ограничен. По умолчанию, скорость не ограничена.

Скорость может быть указана как /s или дополнительно /s@.

RATE выражается в байтах и может принимать суффиксы:

- GB, MB, KB, B для байт;
- Gb, Mb, Kb, b для бит.

INTERVAL выражается в микросекундах и может принимать суффиксы ms, us, s. Это задаёт частоту, на которой пополняются кредитные передачи VIF-устройства. По умолчанию это 50ms.

Лимит VIF-скорости — кредитный: для 1MB/s@20ms доступный кредит будет равен трафику, который используется на 1Мбайт/сек за 20 мс. В результате кредит составляет 20000 байт, пополняемые каждые 20000 микросекунд.

Пример.

- rate=10Mb/s — до 10 мегабит каждую секунду;

- rate=250KB/s — до 250 килобайт каждую секунду;
- rate=1MB/s@20ms — 20000 байт за каждый 20-миллисекундный период.

Фактические пределы ограничения скорости зависят от исходной netback-реализации.

5.4.3. Конфигурация параметров дисков в xl.cfg

Этот документ описывает опцию конфигурационного файла xl для задания формата дисков. Она имеет следующую форму:

```
disk = [ '<diskspec>', '<diskspec>', ... ]
```

где каждый diskspec выражен в такой форме:

```
[<key>=<value>|<flag>,]*,
[<target>, [<format>, [<vdev>, [<access>]]]],
[<key>=<value>|<flag>,]*
[target=<target>]
```

Например, эти строчки эквивалентны:

```
/dev/vg/guest-volume,,hda
/dev/vg/guest-volume,raw,hda,rw
format=raw, vdev=hda, access=rw, target=/dev/vg/guest-volume
```

Как и эти:

```
/root/image.iso,,hdc,cdrom
/root/image.iso,,hdc,,cdrom
/root/image.iso,raw,hdc,devtype=cdrom
format=raw, vdev=hdc, access=ro, devtype=cdrom, target=/root/image.iso
```

Они могут быть указаны в конфигурационном файле домена следующим образом:

```
disk = [ '/dev/vg/guest-volume,,hda', '/root/image.iso,,hdc,cdrom' ]
```

Более формально строка представляет собой последовательность разделённых запятой пар «ключевое слово/значение», флагов и позиционных параметров. Параметры, которые не являются параметризованными ключевыми словами и не содержат знак «=», относятся к позиционным параметрам, порядок указания которых указан ниже. Позиционные параметры также можно эксплицитно задать по имени.

Каждый параметр может быть указан не более одного раза: либо в виде позиционного параметра, либо в виде именованного параметра. Значения по умолчанию применяются, если параметр не указан, или же указан с пустым значением (позиционно или эксплицитно).

Пробелы могут появляться перед каждым параметром и будут проигнорированы.

5.4.4. Позиционные параметры

Ниже приведены позиционные параметры.

5.4.4.1. Target

Описание: Блочное устройство или путь к файлу образа. При использовании в качестве пути будет добавлен /dev, если путь не начинается с «/».

Поддерживаемые значения: N/A Устаревшие значения: N/A.

Значение по умолчанию: Отсутствует. Путь задается во всех случаях, но существует исключение: для устройства с cdrom отсутствие этого атрибута подразумевает пустой привод.

Если этот параметр задан по имени, то есть с синтаксисом «target=» в конфигурационном файле, то он воспринимает весь остальной <diskspec>, в том числе и пробелы. Поэтому в таком случае он должен стоять последним. Это допустимо, даже если пустое значение для задания было указано в качестве позиционного параметра. Это единственный способ указать строку задания, содержащую метасимволы, такие как запятые и (в некоторых случаях) двоеточия, которые иначе были бы поняты неверно.

Имена последующих параметров и флагов будут начинаться с ascii буквы и содержать только буквы и цифры кода ascii, дефисы и подчёркивания, и не будут допустимыми как vdevs. Цели, которые могут соответствовать этому синтаксису, не должны указываться в качестве позиционных параметров.

5.4.4.2. Format

Описание: Определяет формат файла образа.

Поддерживаемые значения: raw, qcow, qcow2, vhd.

Устаревшие значения: нет.

Значение по умолчанию: raw.

5.4.4.3. Vdev

Описание: Виртуальное устройство, каким его видит «гость» (также упоминается как «гостевое обозначение диска» в некоторых спецификациях).

Поддерживаемые значения: hd[x], xvd[x], sd[x] etc.

Для дополнительной информации обратитесь к спецификации выше.

Устаревшие значения: нет.

Значение по умолчанию: отсутствует. Этот параметр является обязательным.

5.4.4.4. Access

Описание: Информация по управления доступом. От этого атрибута зависит, предоставляется ли блочное устройство “гостю” в режиме только для чтения или в режиме чтения и записи.

Поддерживаемые значения: ro|r (режим только для чтения), rw|w (режим чтения и записи).

Устаревшие значения: нет.

Значение по умолчанию: rw (За исключением случаев, когда devtype=cdrom, тогда всегда «r»).

5.4.5. Другие параметры и флаги

5.4.5.1. devtype=<devtype>

Описание: Определяет тип виртуального устройства. Поддерживаемые значения: cdrom.

Устаревшие значения: отсутствуют.

Обязательный параметр: нет.

5.4.5.2. backend=<domain-name>

Описание: обозначает бэкенд домен для устройства. Поддерживаемые значения: Допустимые имена доменов.

Обязательный параметр: нет.

Определяет бэкенд домен, к которому должно быть подсоединено данное устройство. По умолчанию это domain 0. Для задания другого домена требуется настройка драйвер-домена.

5.4.5.3. backendtype=<backend-type>

Описание: определяет реализацию бэкенда для использования Поддерживаемые значения: phy, tap, qdisk.

Обязательный параметр: нет.

Значение по умолчанию: автоматически определить, какой бэкенд должен использоваться.

Не влияет на то, каким «гость» видит устройство. Опция лишь контролирует, какая реализация ПО бэкенд драйвера Xen здесь используется. Не все бэкенд драйверы поддерживают все комбинации других опций. Например, «phy» не поддерживает никакие другие форматы кроме «raw».

Обычно эта опция не должна указываться; в этом случае libxl автоматически определит наиболее подходящий бэкенд.

5.4.5.4. script=<script>

Указывает, что <target> не является обычным путем хоста, а представляет собой, скорее, информацию, которую должна понять исполняемая программа <script> (ищется в /etc/xen/scripts, если путь не содержит слэш).

Обычно эти скрипты называются «block-<script>».

5.4.5.5. direct-io-safe

Описание: Disables non-O_DIRECT workaround.

Поддерживаемые значения: absent, present.

Обязательный параметр: нет.

Значение по умолчанию: absent.

Этот параметр влияет на производительность и масштабирование и необходим только в том случае, если основным устройством является не сетевая файловая

система, а локальное устройство. В противном случае было бы целесообразно отключить эту опцию.

Важно отметить, что, если вы храните диск VM на сетевой файловой системе или на сетевом блочном устройстве (NFS или ISCSI), то использование данной опции может быть небезопасным. В других ситуациях это безопасно, и может способствовать повышению производительности.

5.4.5.6. discard / no-discard

Описание: запросить, чтобы бэкенд уведомлял фронтенд о поддержке discard.

Поддерживаемые значения: discard, no-discard.

Обязательный параметр: нет.

Значение по умолчанию: discard.

Рекомендуемый параметр для бэкенд драйвера, определяющий, следует ли уведомлять фронтенд о поддержке discard (TRIM, UNMAP). Преимуществом этой опции является возможность принудительно ее выключить, а не включить. Можно использовать это для отключения «перфорации» для бэкендов на базе файлов, которые были намеренно созданы неразреженными, чтобы избежать фрагментации файла.

5.5. Планировщик реального времени RTDS

RTDS (англ. Real-Time Deferrable Server) представляет собой планировщик ЦП в режиме реального времени, предоставляющий гостевым ВМ на хостах SMP гарантированную мощность процессора.

Каждому виртуальному ЦП (VCPU) каждого домена выделяются бюджет и период. Бюджет пополняется в начале каждого периода. В процессе планирования VCPU «сжигает» свой бюджет. VCPU должен успеть израсходовать свой бюджет окончания каждого периода; в конце периода неиспользованной бюджет сбрасывается. Если VCPU расходует весь бюджет до окончания периода, он вынужден ждать следующего периода.

Каждый VCPU реализован в виде допускающего задержку сервера. В ходе выполнения задачи в VCPU его бюджет непрерывно сжигается. Если у VCPU не остается задач, но остаётся бюджет, этот бюджет сохраняется.

RTDS опирается на теорию Preemptive Global Earliest Deadline First (EDF) в поле реального времени (англ. real-time field) для планирования процессоров VCPU. В любой момент планирования VCPU с более ранним окончанием периода имеет более высокий приоритет. Планировщик всегда выбирает VCPU с наивысшим приоритетом для работы на подходящем PCPU. PCPU подходит для VCPU в случае, если PCPU находится в режиме ожидания или имеет работающий на нем VCPU с более низким приоритетом.

В RTDS используется схема очередности global run queue и global depletedq для каждого пула процессора. Схема run queue содержит все готовые к выполнению процессоры VCPU с имеющимся бюджетом и отсортированные по окончанию периода; схема depletedq содержит все неотсортированные VCPU без бюджета.

Подразумевается, что процессор VCPU будет работать с бюджетом (<budget>) в каждый период (<period>) - необязательно непрерывно. Виртуальные процессоры одного и того же домена имеют одинаковые параметры в данный момент.

Команда xl sched-rtds может использоваться для настройки параметров гостевого планировщика для каждой ВМ:

- xl sched-rtds -d <domain> - вывести список параметров указанного <domain>;
- xl sched-rtds -d <domain> -p <period> -b <budget< - установить бюджет каждого VCPU на <budget< и период на <period> для указанного <domain>.

5.6. Примеры конфигураций виртуальных машин

5.6.1. Конфигурация ВАРМ без PCI passthrough

```
builder=’hvm’
memory = 8192
# Область для MMIO диапазонов устройств. Для ВМ без PCI passthrough добавлять данную опцию нет нужды.
#mmio_hole=1300
vcpus = 4
name = “vm01”
```

```
# Используем Q35 как чипсет. Без указания данной опции по умолчанию используется чипсет i440
machine=’q35’

disk = [
# AoE (ATA over Ethernet)
#    ‘phy:/dev/etherd/e0.1,hda,w’,

# Локальный образ диска
#    ‘phy:/dev/zvol/storage/vm01,hda,w’,

# Диск через iSCSI
   ‘phy:/dev/disk/by-path/ip-192.168.0.2:3260-iscsi-iqn.2017-05.ru.russoft-nn.macavity:win-vm01-lun-0,hda,w’

# Локальные ISO - один с инсталляторомм ОС, другой - с дровами PV, NVidia, ATI, тестовым софтом
#    ‘file:/home/ISO/WIN8.1_ENT.ISO,hdc:cdrom,r’,
#    ‘file:/home/ISO/xen_drivers.iso,hdd:cdrom,r’
]

# DHCP знает, что этот MAC - vm01, и выдает соответствующий адрес.
# Соответственно, ВМ доступна по имегни в DNS, имеет reverse DNS запись и корректный hostname.
vif = [’bridge=xenbr0, mac=00:16:3e:38:3c:01’]

boot = ‘cd’

# Что делаем в каком случае
on_xend_stop = ‘shutdown’
on_poweroff=’destroy’
#on_reboot=’destroy’
on_reboot=’restart’
on_crash=’destroy’

vga=’stdvga’
sdl=0
vnc=1
vncdisplay=”1”
# Доступно снаружи, а не только localhost.
vnclisten=”0.0.0.0”
# vncpasswd не задаем - вход без пароля.

serial=[’stdio’]
keymap=”en-us”

usb=1
usbdevice=[’tablet’]

localtime=1
```

```
# Пусть знает, что в виртуалке, включаем часть ABI Hyper-V
viridian=[ “all” ]
# Используем EPT
hap=1
# Управление питанием
acpi=1
# не нужны standby / hibernate
acpi_s3=0
acpi_s4=0
apic=1
hpet=1
pae=1
nx=1
pci_power_mgmt=1
pci_msitranslate=0

# Пока не нужно
#pci_passthrough=1
#pci_permissive=1

# Если не ставить PV драйверы, то можно выключить platform_pci, чтобы не болтался в device manager-е Винды. Для Linux HVM лучше оставлять всегда, чтобы автоматом работали PV драйвера.
xen_platform_pci=1
xen_extended_power_mgmt=1
```

5.6.2. Secondary VGA - NVidia

```
builder=’hvm’
memory = 8192
mmio_hole=1300
vcpus = 4
name = “vm10”

machine=’q35’
        disk = [’phy:/dev/disk/by-path/ip-192.168.0.2:3260-iscsi-iqn.2017-05.ru.russoft-nn.macavity:win-vm01-lun-0,hda,w’,
   ‘phy:/dev/disk/by-path/ip-192.168.0.2:3260-iscsi-iqn.2017-05.ru.russoft-nn.macavity:win-vm02-lun-0,xvdb,w’]

vif = [’bridge=xenbr0, mac=00:16:3e:38:3c:10’]

boot = ‘cd’

on_xend_stop = ‘shutdown’
        on_poweroff=’destroy’
#on_reboot=’destroy’
on_reboot=’restart’
on_crash=’destroy’

vga=’stdvga’
```

```
sdl=0

vnc=1
vncdisplay="10"
vnclisten="0.0.0.0"

serial=['stdio']
keymap="en-us"

usb=1
usbdevice=['tablet']

localtime=1

viridian=[ "all" ]

hap=1
acpi=1
acpi_s3=0
acpi_s4=0
apic=1
hpet=1
pae=1
nx=1
pci_power_mgmt=1
pci_msitranslate=0
pci_passthrough=1
pci_permissive=1
xen_platform_pci=0
xen_extended_power_mgmt=0

# Используем usb tablet от QEMU, USB можно не пробрасывать.
#rdm = "strategy=host,policy=relaxed"

# Не перенаправляем QEMU VGA
gfx_passthru=0

# Quadro M4000 как secondary
pci = ['04:00.0', '04:00.1']
# nvidia_xen_hiding - экспериментальная опция, позволяющая использовать не Quadro/
GRID/Tesla, а в том числе и бытовые видеокарты.
```

5.6.3. Конфигурация для вспомогательной ВМ с пробросом RAID контроллера (driver domain)

```
name = "vm12-pvdd"
kernel = "/home/kernels/vmlinuz-4.9.0-ogun1-amd64"
initrd = "/home/kernels/initrd.img-4.9.0-ogun1-amd64"

# Чтобы консоль работала, ее надо прописать в /etc/inittab
extra = "debug earlyprintk=xen root=/dev/xvda2 console=hvc0"
```

```
memory = 16384
vcpus = 6

# CPU Affinity - критично, чтобы ВМ была на том-же процессоре, к которому подключен контроллер. В данном случае - CPU0
cpus=”6-11”

# В PVH режиме проброс PCI не работает. Без проброса, и с ядрами 3.18+ - PVH крайне рекомендуется.
#pvh=1

vif = [ ‘mac=00:16:3e:38:3c:12’ ]

# Основные диски (ротационные) - через RAID, они определятся как /dev/sd{ a,b,cd}
# Используем системные SSD для кэша - ZFS ZIL/ARC2 - с SATA контроллера на мат. плате, их пробрасываем из хост-системы как паравиртуальные (xvd*)
# Загрузочный образ системы - на iSCSI
disk = [ ‘phy:/dev/disk/by-path/ip-192.168.0.2:3260-iscsi-iqn.2017-02.ru.iqint.macavity:linux-vm-lun-1,xvda,rw’,
    ‘phy:/dev/disk/by-id/ata-OCZ-VECTOR150_OCZ-HBF02PHL48L905FK-part5,xvdb,rw’,
    ‘phy:/dev/disk/by-id/ata-OCZ-VECTOR150_OCZ-RM2YYN7049S1MXTK-part5,xvdc,rw’,
    ‘phy:/dev/disk/by-id/ata-OCZ-VECTOR150_OCZ-HBF02PHL48L905FK-part6,xvdd,rw’,
    ‘phy:/dev/disk/by-id/ata-OCZ-VECTOR150_OCZ-RM2YYN7049S1MXTK-part6,xvde,rw’ ]

on_poweroff=’destroy’
on_reboot=’restart’
on_crash=’destroy’

sdl=0
vnc=0

localtime=1

# Пробразываем RAID контроллер:
# 07:00.0 Serial Attached SCSI controller: Intel Corporation C602 chipset 4-Port SATA Storage Control Unit (rev 06)
pci=[’07:00.0’]

# Внимание!!! Adapted RAID/HBA - не работают в PV/PVH режимах,в том числе и в Dom0!! (И, кстати, даже исключены из HAL Citrix-а).
# Работают только в HVM, т.е. доступ к его дискам можно организовать лишь по сети.
# Связано это с использованием привилегированных команд, которые драйвер использовать не должен, и которые игнорируются в PV режиме.
```

5.6.4. Пример работы с Storage driver domain (PVH ВМ)

Работа HVM со сторадж доменами возможна только если диски подключены к хост-системе или STUB домену.

```
name = “vm10-dd-test”
pvh = 1
kernel = “/home/kernels/vmlinuz-4.9.0-ogun1-amd64”
```

```
initrd = “/home/kernels/initrd.img-4.9.0-ogun1-amd64”
extra = “debug earlyprintk=xen root=/dev/xvda2”
memory = 4096
vcpus = 2

vif = [ ‘mac=00:16:3e:38:3c:10’ ]
disk = [ ‘backend=vm12-pvdd,/dev/zvol/storage/stubdom-disk,raw,xvda,w’,
    ‘backend=vm12-pvdd,/dev/zvol/storage/vm-dd-test,raw,xvdb,w’,
    ‘file:/home/ISO/debian-live-8.7.1-amd64-standard.iso,hdd:cdrom,r’]

on_poweroff=’destroy’
on_reboot=’restart’
on_crash=’destroy’
```

5.6.5. PCI Passtrough - Primary VGA (NVidia)

Для NVidia Primary VGA требует чтобы видеокарта стояла в PCIE слоте CPU0, иначе будут конфликты с перехватом чипсетом обращений к VGA и перенаправлением их к видеоадаптеру IPMI (Matrox/AST).

Также, соответственно, нужно выставить CPU Affinity.

Рекомендуется подключать видеокарту в уже поставленную систему, но в приципе можно попробовать и инсталлировать с пробросом.

Не работает с драйвером nouveau в Linux - его нужно либо поместить в blacklist, либо запускать систему с “nomodeset” до установки проприетарного драйвера NVidia. В противном случае на нем виснет. Но вообще, с Linux все проще - там нет разницы, primary или secondary VGA - одинаково хорошо и так и так работает - главное правильно настроить XOrg (указать устройство GPU и драйвер).

```
builder=’hvm’
memory=8192
mmio_hole=1300
vcpus = 4

# Критично, пробрасываемая карта - на CPU0
# 26.07.2017, AG: главное значение имеют настройки BIOS Setup хоста (какая карта выбрана там основной) или физический слот, в который вставлена карта, поэтому на данный момент привязка к NUMA-ноде более не актуальна (кроме производительности)?
cpus=”0-11”
name = “vm10”

machine=’q35’

#disk = [
    ‘phy:/dev/zvol/storage/vm10-c-quadro,hda,w’,
    ‘phy:/dev/zvol/storage/vm10-d,hdb,w’]
```

```
    'file:/home/ISO/WIN8.1_ENT.ISO,hdc:cdrom,r',
    'file:/home/ISO/drivers_win.iso,hdd:cdrom,r']

vif = ['bridge=xenbr0, mac=00:16:3e:38:3c:10']

boot = 'cd'

on_xend_stop = 'shutdown'
on_poweroff='destroy'
# Для отладки, и если нужно конфиг поправить после перезагрузки - например, отключить ISO.
on_reboot='destroy'
#on_reboot='restart'
on_crash='destroy'

# Не используем VNC, работает видеокарта.
vga='none'

sdl=0
vnc=0

serial=['stdio']
keymap="en-us"

localtime=1

viridian=[ "all" ]

hap=1
acpi=1
acpi_s3=0
acpi_s4=0
apic=1
hpet=1
pae=1
nx=1
#pci_power_mgmt=1
pci_msitranslate=0
pci_passthrough=1
pci_permissive=1
xen_platform_pci=1

# Необходимо для проброса RDM устройств, а именно - USB контроллера
rdm = "strategy=host,policy=relaxed"

# Режим Primary VGA (Используем NVidia, vga=none и gfx_passthru=1)
gfx_passthru=1

# Пробрасываем видеокарту и оба USB контроллера с мат. платы (можно один)
pci = ['04:00.0,pci_conf_fix=1', '04:00.1','00:1d.0', '00:1a.0']
```

5.6.6. Primary VGA - ATI

У ATI сделано куда прямее, чем в NVidia. Работает и с Primary и с Secondary VGA без проблем (для NVidia пришлось делать специальную обработку). Но все также: под linux с драйвером radeon не работает! Нужно его запрещать, или использовать nomodeset, а в систему ставить проприетарные ATI/AMD драйверы (новые! старые с новыми ядрами не работают). Протестировано на Ubuntu, также есть драйвера на сайте для RedHate.

```
builder=’hvm’
memory=8192
mmio_hole=1300
vcpus = 4
# CPU affinity с ATI не критично, но сильно влияет на скорость работы - устройство 82:00 - на PCIe шине CPU1
cpus=”12-23”
name = “vm11”

machine=’q35’

disk = [
   ‘phy:/dev/zvol/storage/vm10-ati,hda,w’]
      ‘phy:/dev/disk/by-path/ip-192.168.0.2:3260-iscsi-iqn.2017-02.ru.iqint.macavity:zfs-lun-1,hdb,w’]

vif = [’bridge=xenbr0, mac=00:16:3e:38:3c:11’]

boot = ‘cd’

on_xend_stop = ‘shutdown’
on_poweroff=’destroy’
on_reboot=’destroy’
#on_reboot=’restart’
on_crash=’destroy’

vga=’none’

sdl=0
vnc=0

serial=[’stdio’]
keymap=”en-us”

localtime=1

viridian=[ “all” ]

hap=1
acpi=1
```

```
acpi_s3=0
acpi_s4=0
apic=1
hpet=1
pae=1
nx=1
pci_power_mgmt=1
pci_msitranslate=0
pci_passthrough=1
pci_permissive=1

# Если не ставить PV драйверы, то можно выключить platform_pci, чтобы не отображался в device manager-e Windows. Для Linux лучше оставлять всегда, чтобы автоматически работали PV драйвера.
xen_platform_pci=0
xen_extended_power_mgmt=1

rdm = “strategy=host,policy=relaxed”

# Используем ATI/nvidia как основной адаптер
gfx_passthru=1

# ATI R290X как Primary VGA и USB с мат. платы
# 00:1d.0 USB controller: Intel Corporation C600/X79 series chipset USB2 Enhanced Host Controller #1 (rev 06)
# 82:00.0 VGA compatible controller: Advanced Micro Devices, Inc. [AMD/ATI] Hawaii XT [Radeon R9 290X]
# 82:00.1 Audio device: Advanced Micro Devices, Inc. [AMD/ATI] Device aac8

pci = [’82:00.0,pci_conf_fix=1,vbios_workaround=1’, ‘82:00.1’,  ‘00:1d.0’]
```

5.6.7. Secondary VGA - ATI

```
builder=’hvm’
memory = 8192
mmio_hole=1300
vcpus = 4
name = “vm11”

machine=’q35’

disk = [phy:/dev/zvol/storage/vm10-ati,hda,w’]

vif = [’bridge=xenbr0, mac=00:16:3e:38:3c:11’]

boot = ‘cd’

on_xend_stop = ‘shutdown’
on_poweroff=’destroy’
#on_reboot=’destroy’
on_reboot=’restart’
on_crash=’destroy’
```

```
vga='stdvga'

sdl=0

vnc=1
vncdisplay="11"
vnclisten="0.0.0.0"
vncpasswd=""

serial=['stdio']
keymap="en-us"

usb=1
usbdevice=['tablet']

localtime=1

viridian=[ "all" ]

hap=1
acpi=1
acpi_s3=0
acpi_s4=0
apic=1
hpet=1
pae=1
nx=1
pci_power_mgmt=1
pci_msitranslate=0
pci_passthrough=1
pci_permissive=1
xen_platform_pci=1
xen_extended_power_mgmt=1

gfx_passthru=0

pci = ['04:00.0', '04:00.1']
```

6. НАСТРОЙКА КОМПОНЕНТОВ УПРАВЛЯЮЩЕГО ДОМЕНА

Ниже приводится описание настройки компонентов для кросс-платформенного взаимодействия в информационной системе с применением программного модуля «Технология «тонкого клиента» «Синергия-ТК» и ссылки на документацию. В связи с тем, что используются в основном стандартные для большинства ОС семейства UNIX компоненты, подробно описываются только изменения, специфичные для программного модуля «Технология «тонкого клиента» «Синергия-ТК». Для компонентов, включаемых без изменения порядка их настройки, приводятся ссылки на публично доступную документацию.

Также приводится последовательность действий по созданию HVM виртуальных машин из образа Dom0 (или любого другого снимка системы, по аналогии) для использования в качестве доменов ввода-вывода, миграции физических ВМ в виртуальные, и т.д.

6.1. Начальный загрузчик GRUB2

Начальный загрузчик GRUB используется для первичной загрузки менеджера виртуальных машин, контроля целостности образов программного модуля «Технология «тонкого клиента» «Синергия- ТК» и задания параметров загрузки как менеджера ВМ, так и управляющего домена.

Параметры загрузки системы задаются в конфигурационом файле /boot/grub/grub.cfg. Шаблон конфигурационного файла, используемого по умолчанию, приведён ниже:

```
 1
 2  terminal_input console
 3 terminal_output console
 4
 5 insmod gzio
 6 insmod xzio
 7 insmod lzopio
 8 insmod part_gpt
 9 insmod part_msdos
10 insmod ext2
11 insmod search
12
13 set default=”0”
```

```
14 set timeout=15
15 set gfxmode=640x480
16 set menu_color_normal=cyan/blue
17 set menu_color_highlight=white/blue
18 set menutext1=”Load XEN-based Hypervisor”
19 set menutext2=”Load dom0 image only”
20 set loading_msg=”Loading”
21 set file_msg=”File”
22 set corrupted_msg=”is corrupted.”
23 set consistent_msg=”is consistent.”
24
25 if [ x$feature_platform_search_hint = xy ]; then
26   search --no-floppy --fs-uuid --set=root --hint-bios=hd0,gpt1 --hint-efi=hd0,gpt1 --hint-
baremetal=ahci0,gpt1 @BOOT_UUID@
27 else
28   search --no-floppy --fs-uuid --set=root @BOOT_UUID@
29 fi
30
31 if [ x$feature_all_video_module = xy ]; then
32   insmod all_video
33 else
34   insmod vbe
35   insmod vga
36   insmod video_bochs
37   insmod video_cirrus
38 fi
39 insmod gfxterm
40
41 if terminal_output gfxterm ; then
42  # Грузим модули графики. Если все нормально - подгружаем фон.
43  if [ -f /grub/background.png ] ; then
44   insmod png
45   background_image /grub/background.png
46  fi
47
48  # Грузим шрифт. Если все нормально - устанавливаем русские сообщения.
49  if loadfont /grub/fonts/unicode.pf2 ; then
50   set locale_dir=/grub/locale
51   set lang=ru
52   insmod gettext
53   # Если удалось переключиться в графический режим при
54   # загруженом шрифте - заголовки покажем по русски.
55   set menutext1=”Загрузить Гипервизор на базе XEN”
56   set menutext2=”Загрузить только образ dom0”
57   set loading_msg=”Загружаем”
58   set file_msg=”Файл”
59   set corrupted_msg=”поврежден.”
60   set consistent_msg=”не поврежден.”
61  else
62   terminal_output console
63  fi
64 fi
```

```
65
66 function check_sum {
67  if hashsum --hash sha512 --check ${ 1} .sha512 ; then
68   echo “$file_msg $1 $consistent_msg”
69  else
70   echo “$file_msg $1 $corrupted_msg”
71   sleep 10
72   halt
73  fi
74 }
75
76 set superusers=”root”
77 password_pbkdf2 root @GRUB_PASSWD@
78
79 menuentry “$menutext1” --unrestricted {
80  check_sum /hypervisor.gz
81  echo “$loading_msg hypervisor.gz”
82  multiboot /hypervisor.gz conring_size=51200K loglvl=all guest_loglvl=all iommu=1,verbose
com2=115200,8n1 console=vga,com2 console_timestamps=datems cpufreq=xen:performance
dom0_mem=4G dom0_max_vcpus=4 dom0_vcpus_pin flask=permissive
83  check_sum /tk2019kernel
84  echo “$loading_msg tk2019kernel”
85  module /tk2019kernel placeholder root=/dev/ram0 ramdisk_size=536870912 ro
net.ifnames=0 biosdevname=0 nomodeset earlyprintk=xen
86  check_sum /tk2019image
87  echo “$loading_msg tk2019image”
88  module --nounzip   /tk2019image
89  module /policy.mls
90  }
91
92 menuentry “$menutext2” --unrestricted {
93  echo “$loading_msg tk2019kernel”
94  linux /tk2019kernel root=/dev/ram0 ramdisk_size=536870912 ro net.ifnames=0
biosdevname=0 nomodeset
95  echo “$loading_msg tk2019image”
96  initrd /tk2019image
97 }
98
```

При установке программного модуля «Технология «тонкого клиента» «Синергия-ТК» скрипт-инсталлятор создаёт файловые системы и определяет их UUID, вычисляет контрольные суммы загрузочных файлов и подставляет эти значения в шаблон.

Конфигурационный файл GRUB2, шаблон которого приведён выше, является командным скриптом, согласно которому загрузчик выполняет следующие действия:

- устанавливает тип вывода на экран и задаёт текст выводимых сообщений;

- производит попытки перевода консоли в графический режим и загрузки русского экранного шрифта. В случае успеха — текст выводимых сообщений переопределяется на русскоязычные фразы;
- производит проверку целостности (рассчет хэш-сумм) файлов менеджера виртуальных машин, ядра и образа Dom0. В случае несовпадения с заданными значениями, загрузка останавливается;
- предлагает пользователю меню загрузки с выбором из 2-х пунктов — полноценный запуск программного модуля «Синергия-ТК» или запуск только управляющего домена. По умолчанию загружаются все компоненты программного модуля «Синергия-ТК», запуск dom0 отдельно доступен только для Администратора (защищён логином и паролем).

Загрузчик GRUB2 позволяет осуществлять редактирование пунктов меню, но для этого необходимо указать логин и пароль Администратора.

Подробная публично доступная документация на компонент GRUB2 находится по адресу: https://www.gnu.org/software/grub/manual/grub/.

6.2. Ядро Linux

В управляющем домене dom0 используется стандартное ядро GNU Linux. Ядро может получать параметры от начального загрузчика GRUB. Параметры, принимаемые ядром, стандартны и описаны в общедоступной документации ядра Linux на сайте https://kernel.org.

Версия ядра, используемого в данный момент в программном модуле «Синергия-ТК», определяется командой uname.

Пример.

```
root@tk2019image:~# uname -a;
Linux tk2019image 4.9.88 #1 SMP Mon Jun 25 08:00:00 MSK 2018 x86_64 GNU/Linux.
```

Описание использования некоторых специфичных для менеджера виртуальных машин параметров, например, для организации прямого доступа к PCIe устройствам, приведено в разделах ниже.

6.3. Настройка сети для использования с ВМ

6.3.1. Виртуальные сетевые интерфейсы

6.3.1.1. Паравиртуализированные сетевые устройства

Обычно, у гостевой ОС есть доступ к одному или нескольким сетевым PV-интерфейсам. Эти интерфейсы обеспечивают быструю и эффективную коммуникацию для доменов, без затрат на эмуляцию реальных сетевых устройств. По умолчанию, драйверы PV-интерфейсов доступны в большинстве PV-ядер гостевых ОС. Кроме того, драйверы доступны для различных гостевых ОС при работе в HVM-режиме: к примеру, на HVM-драйверах в Linux или PV-драйверах GPL в Windows.

Каждое PV-устройство состоит из front-end устройства в гостевом домене и back-end устройства в основном домене (обычно Dom0).

Front-end устройства, работающие в Linux, связаны с драйвером xen- netfront и представлены устройством ethN.

В названии back-end устройства обычно указаны ID гостевого домена и устройства. По умолчанию, в Linux такие устройства называются vifDOMID.DEVID, где DOMID — это ID домена, а DEVID — ID устройства.

Устройства front-end и back-end связаны виртуальным каналом связи. Гостевая сеть устанавливается путём организации трафика, переходящего от back-end устройства на более широкую сеть с помощью, к примеру, бриджинга (см. п. 4.3.2), маршрутизации или преобразования сетевых адресов (NAT).

Схема доменов PV-устройства приведена на рис. 1.

Рисунок 1

6.3.1.2. Эмулированные сетевые устройства

Как и в случае с устройствами PV, гостевые ОС, работающие в HVM- режиме, могут поддерживать несколько сетевых устройств. Эти устройства эмулируют реальные части оборудования и нужны в случае, когда в гостевой ОС не установлены или ещё не доступны PV-драйверы (например, во время установки ОС).

Эмулированное сетевое устройство, как правило, работает в паре с PV-устройством с тем же MAC-адресом и конфигурацией. Это позволяет гостевой ОС плавно переходить от эмулированного к PV-устройству, как только драйвер становится доступным. Эмулированное сетевое устройство предоставляется моделью устройства (англ. device model), работающей как процесс в Dom0 или в stub-домене.

В случае с Dom0, устройство можно обнаружить в back-end домене как сетевое устройство типа tap. Такие устройства также обозначаются как vifDOMID=DEVID, что подчёркивает парную связь PV-домена и эмулированного устройства.

В случае со stub-доменом, устройство можно обнаружить в Dom0 как сетевое PV-устройство, подключённое к stub-домену. Stub-домен обеспечивает передачу данных между эмулятором устройства и PV-устройством.

Термины PV-устройство и эмулированное устройство взаимозаменяемы.

6.3.1.3. MAC-адреса

Виртуализированным сетевым интерфейсам в доменах присваиваются MAC-адреса Ethernet. В зависимости от набора инструментов (англ. toolstack) менеджера виртуальных машин, это либо произвольный статический адрес, неизменный в ходе всего цикла жизни гостевой ОС (например, Libvirt или XAPI управляемых доменов), либо динамический, меняющийся при каждом новом запуске ОС (например, xl неуправляемых доменов).

В последнем случае, если требуется зафиксировать MAC-адрес (например, для использования DHCP), можно выполнить настройку с помощью опции mac с директивой конфигурации vif.

Например, vif = ['mac=aa:QQ:QQ:QQ:QQ:11'].

При выборе MAC-адреса могут быть использованы три ключевых стратегии. Стратегии перечислены ниже начиная с наиболее оптимальной:

1. Задать адрес из диапазона, связанного с уникальным идентификатором организации (OUI);
2. Создать случайную последовательность из 6 байт, установить локально управляемый бит (бит 2 первого байта) и очистить бит многоадресной передачи (бит 1 первого байта). Другими словами, первый байт должен иметь битовый шаблон xxxxxxlO, где x — случайно сгенерированный бит. Остальные 5 байт генерируются также случайным образом;
3. Задать случайный адрес внутри пространства QQ:16:3e:xx:xx:xx, где QQ:16:3e — OUI, назначенный для менеджера виртуальных машин, чтобы его пользователи могли задавать локальные адреса в этом пространстве.

MAC-адрес должен быть уникальным для всех сетевых устройств (как физических, так и виртуальных) в одном сегменте локальной сети (например, в локальной сети LAN, к которой подключён хост менеджера виртуальных машин). По этой причине, при отсутствии OUI, рекомендуется использовать случайно сгенерированный локально управляемый адрес (стратегия 2). В этом случае вы получите 46, а не 12, случайных бит, что значительно снижает шансы на столкновение.

6.3.2. Сетевой мост

По умолчанию, конфигурация менеджера виртуальных машин использует сетевой мост (также мост, бридж) в back-end домене. Таким образом, все домены в сети отображаются в виде отдельных хостов.

В этой конфигурации программный мост также создаётся в back-end домене. Back-end виртуальные сетевые устройства (vifDOMID.DEVID) добавляются к мосту наряду с физическим Ethernet-устройством — для обеспечения ещё одного канала связи, помимо хоста. Опуская физическое устройство, можно создать изолированную сеть, содержащую только гостевые домены.

Существуют две общепринятые схемы назначения имён при использовании моста:

- физическое устройство eth0 переименовывают в peth0, затем создают мост eth0;
- физическому устройству оставляют имя eth0, а мосту дают имя xenbr0 или br0etc.

На рис. 2 показаны примеры мостовых соединений.

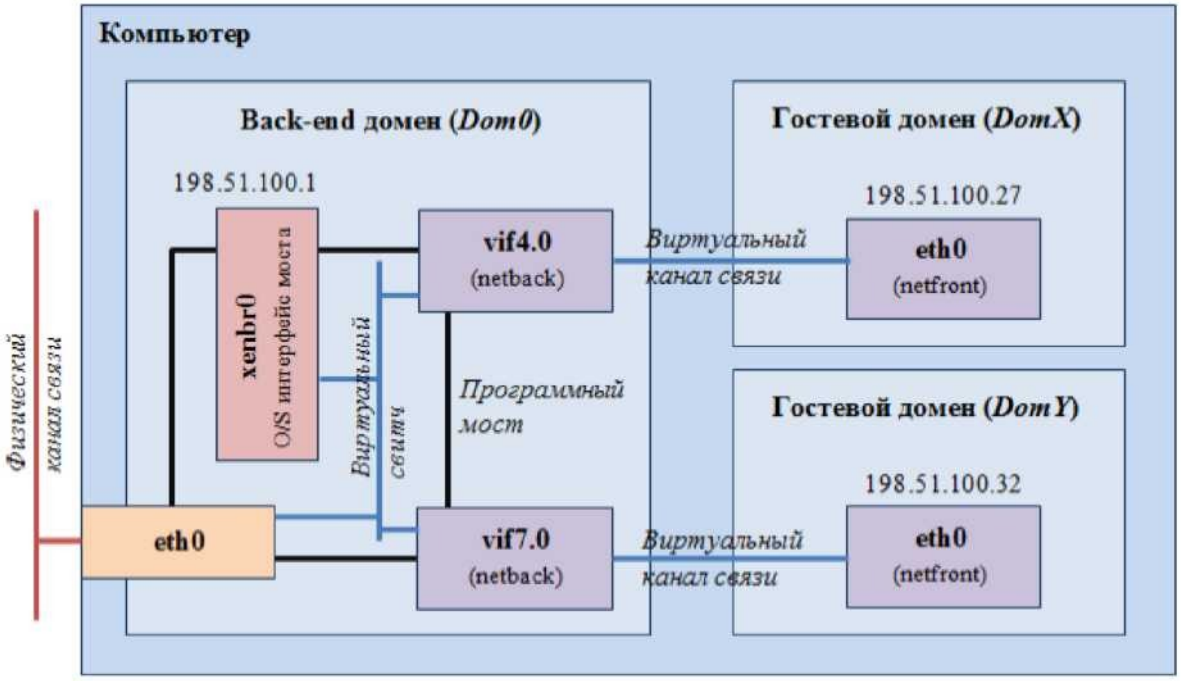

Рисунок 2

6.3.2.1. Настройка моста

Рекомендуется настраивать мост в соответствии с конфигурацией сети, специфичной для конкретной АСЗИ. Не рекомендуется использовать неустойчивые сценарии, в которых существующая конфигурация физической сети меняется на конфигурацию сетевого моста.

Пример настройки сетевого моста находится в стартовом скрипте /etc/rc.local.

Набор инструментов xl не меняет конфигурацию сети; предполагается, что за правильность настройки отвечает сетевой администратор.

6.3.2.2. Подключение виртуальных устройств

Одновременно с DomU запускается сценарий vif-bridge, который подсоединяет vifDOMID.DEVID к мосту и запускает его. С xl для каждого VIF-устройства можно создать мост с помощью ключа bridge:

```
vif=[ 'bridge=mybridge' ]
```

Или

```
vif=[ 'mac=00:16:3e:01:01:01,bridge=mybridge'
]
```

Также можно создать разные интерфейсы, присоединённые к разным мостам:

```
vif=[ 'mac=00:16:3e:70:01:01,bridge=br0',
'mac=00:16:3e:70:02:01,bridge=br1' ]
```

6.3.3. Мостовые петли

Отключение протокола STP на мостах менеджера виртуальных машин является распространённой (и рекомендуемой) практикой. Однако, если гостевые ОС могут самостоятельно объединять в мост два или более интерфейса, возникает риск создания мостовых петель.

При настройке гостевых ОС следует учитывать, что при включении протокола STP, передача первых пакетов трафика может происходить с задержкой, соответствующей настройкам STP на сетевом оборудовании (например, у оборудования Cisco такая задержка по умолчанию составляет 120 сек.), что может

вызывать проблемы при использовании статических IP адресов и сетевых протоколов.

6.4. Организация прямого доступа ВМ к PCI/PCIe устройствам

6.4.1. Описание прямой передачи устройств шины ввода-вывода

Данный раздел затрагивает как конфигурацию управляющего домена, так и настройку параметров в конфигурационном файле виртуальной машины.

Прямая передача устройств шины ввода-вывода (PCI) позволяет передавать управление физическими устройствами гостевым доменам. Прямую передачу используют для присвоения устройств PCI (NIC, контроллер диска, HBA, контроллер USB, контроллер FireWire, звуковая карта и т.д.) виртуальной машине гостя, предоставляя ей полный и прямой доступ к устройству или группе устройств.

Существует несколько причин для применения данного способа передачи, в том числе следующие:

- работа с драйвер-доменами;
- прямая передача видеокарт для гостевых ВМ и 3D-ускорение, что крайне актуально для профессиональных пользователей CAD, выполнения рассчетов и моделирования с использованием GPGPU и вычислительных сопроцессоров.

Устройства PCI задаются с помощью записи BDF (англ. bus/device/ function), которая назначается при запуске утилиты lspci в Dom0.

Dom0 отвечает за все устройства в системе. Обычно при обнаружении устройств PCI он передаёт их драйверам в ядре ОС. Чтобы гость мог иметь доступ к устройству, оно должно быть прикреплено к особому драйверу Dom0 (драйвер xen-pciback в ядрах pvops). PV-гости получают доступ к устройству с помощью драйвера ядра xen-pcifront и подключаются к pciback. HVM-гости могут видеть устройство на эмулированной шине PCI, представленной QEMU.

Гости могут предоставить прямой доступ к памяти (DMA) для устройств: с помощью QEMU в HVM-гостях и с помощью pciback-драйвера в Dom0 в PV-гостях.

Обычно устройствам разрешён DMA к любой и из любой части физической памяти хоста. В связи с этим возникают две проблемы:

- гость с дефектным драйвером (англ. buggy driver) может непреднамеренно перезаписать некоторые части памяти менеджера виртуальных машин;
- домен, управляемый злоумышленником, может читать и записывать память других гостей;
- концепция распределения памяти у гостя виртуализирована, а концепция устройства — нет. PV-гости могут избежать этой проблемы, а HVM гости — нет.

Решением обеих проблем является блок управления памятью для операций ввода-вывода IOMMU.

IOMMU позволяет менеджеру виртуальных машин управлять, к какой именно памяти разрешать доступ устройству, и предоставлять устройству то же виртуализированное распределение памяти, что и гостю.

Режим IOMMU / VT-d отличается от HVM и не совместим с ним.

Настоятельно рекомендуется использовать прямую передачу только на системах с IOMMU. В системах без IOMMU устройства могут быть переданы доверенным PV-гостям, однако при этом теряются преимущества безопасности и стабильности (потери производительности при этом не происходит). Устройства не могут быть переданы HVM-гостям на системах без IOMMU.

6.4.2. Использование прямой передачи

6.4.2.1. Подготовка устройства к прямой передаче

Чтобы подготовить устройство к прямой передаче, определите его BDF (обычно, с помощью запуска lspci в Dom0).

Назначьте устройству pciback-драйвер вместо обычного драйвера в Dom0, чтобы сделать его доступным для передачи гостям. Это можно сделать как статически во время загрузки, так и динамически после загрузки системы. Статический метод менее гибок и требует перезагрузки системы после каждого изменения. Динамический способ является более длительным, но гибким процессом

и не требует перезагрузки. Если модуль xen-pciback содержится в ядре ОС, статический метод будет самым простым решением; если же xen-pciback скомпилирован как модуль, этот вариант, напротив, затратнее.

Ниже приведены опции прямой передачи в порядке убывания простоты их использования.

6.4.2.1.1. Статическое назначение для встроенного xen-pciback

Если pciback-драйвер встроен в ядро (т.е. представлен не в виде модуля), передайте записи BDF в «скрытую» опцию к модулю xen-pciback на командную строку ядра Dom0.

Например, если требуется передать устройства на BDF 08:00.0 и 08:00.1, внесите следующую запись в командную строку Linux ядра Dom0:

xen-pciback.hide=(08:00.0)(08.00.1)

Это скроет устройства от обычных драйверов гостей и назначит им pciback-драйвер при загрузке.

6.4.2.1.2. Динамическое назначение с xl

Если модуль xen-pciback загружен как модуль и не встроен в ядро, убедитесь, что в Dom0 загружен pciback-модуль:

modprobe xen-pciback.

Подготовьте устройство к назначению с помощью xl pci-assignable-add. Например, если требуется сделать устройство на BDF 08:00.0 доступным для гостей, добавьте следующее:

xl pci-assignable-add 08:00.0.

6.4.2.1.3. Динамическое назначение с sysfs

Если xen-pciback загружен как модуль и не встроен в ядро, можно назначить устройство вручную с помощью команд sysfs в Linux.

Убедитесь, что в Dom0 загружен pciback-модуль «modprobe xen-pciback»

Основные этапы:

1) отмените привязку к старому драйверу;

2) создайте новый слот в pciback для устройства;

3) создайте привязку к pciback.

Ниже приведены команды для BDF 08:00.0. Для sysfs домен требуется как часть BDF (почти всегда 0000).

```
echo 0000:08:00.0 >
/sys/bus/pci/devices/0000:08:00.0/driver/unbind echo 0000:08:00.0 > /sys/bus/pci/drivers/pciback/
new_slot echo 0000:08:00.0 > /sys/bus/pci/drivers/pciback/bind
```

Первая команда (отменить привязку) содержит BDF в указании пути, а также echo.

Кроме того, можно использовать следующий скрипт (pciback.sh):

```
#!/bin/bash
if [ $# -eq 0 ]; then
  echo “Require PCI devices in format:
  <domain>:<bus>:<slot>.<function>”
  echo “Eg: $(basename $0) 0000:00:1b.0”
  exit 1
fi
for pcidev in $@; do
  if [ -h /sys/bus/pci/devices/”$pcidev”/driver ]; then
    echo “Unbinding $pcidev from” $(basename $(readlink /sys/bus/pci/devices/”$pcidev”/driver))
    echo -n “$pcidev” > /sys/bus/pci/devices/”$pcidev”/driver/unbind
  fi
  echo “Binding $pcidev to pciback”
  echo -n “$pcidev” > /sys/bus/pci/drivers/pciback/new_slot
  echo -n “$pcidev” > /sys/bus/pci/drivers/pciback/bind
done
```

6.4.2.2. Проверка готовности устройства к прямой передаче

На этом этапе устройство готово к прямой передаче. Проверить это можно с помощью команды xl: xl pci-assignable-list 08:00.0.

6.4.2.3. Настройка домена

HVM-гости не требуют специальной настройки для гостевого ядра, так как все доступы эмулируются и виртуализируются аппаратным обеспечением IOMMU.

PV-гостям нужен модуль xen-pcifront. Кроме того, требуется включить swiotlb в командной строке гостевого ядра.

```
iommu=soft
```

Присвоить устройство гостю можно и во время создания ВМ с помощью файла конфигурации. Если гость поддерживает горячее подключение, устройства могут быть также добавлены динамически.

6.4.2.3.1. Файл конфигурации для DomU

Предположим, требуется передать устройства на BDF 08:00.0 и 08:00.1. Добавьте следующую строку в ваш конфигурационный файл:

```
pci=['08:00.0','08:00.1']
```

6.4.2.3.2. Горячее подключение

Команды для горячего подключения и отключения устройства в работающей ВМ приведены ниже:

```
xl pci-attach <domain-id> <pci device>
<guest virtual slot number> xl pci-detach <domain-id>
<pci device> <guest virtual slot number>
```

6.4.2.4. PV-гости и особенности PCI

Доступ к конфигурационному пространству PCI для PV-гостей контролируется через pciback-модуль. Часто pciback-модуль ограничивает некоторые действия, выдавая сообщение об ошибке.

```
pciback 0000:08:00.0: Driver tried to write to a read-only configuration space field at
offset 0xe0, size 2.
```

Наиболее простой способ обойти эту проблему — включить разрешающий режим.

Чтобы включить разрешающий режим для устройства с помощью xl, включите его для всех устройств данного домена в конфигурационном файле /etc/xen/<domain>: pci_permissive=1.

Также можно добавить опцию к BDF определённого устройства, когда оно передано, либо в конфигурационном файле:

```
pci=['08:00.0,permissive=1']
```

либо во время его «горячего» подключения:

```
xl pci-attach 5 '08:00.0,permissive=1'
```

6.4.3. Дополнительная информация и часто задаваемые вопросы

6.4.3.1. Xen dom0 pciback driver backend modes

В старшей версии Linux 3.1.0 (и более поздних, например, 4.9.x) можно установить PASS/VCPI в качестве опции модуля/драйвера при загрузке драйвера.

Можно использовать такую опцию в командной строке ядра Linux Dom0 в grub.conf, если xen-pciback встроен в ядро:

```
xen-pciback.passthrough=1
```

или следующую, если загружаемый драйвер xen-pciback driver выступает в качестве модуля:

```
modprobe xen-pciback passthrough=1
```

6.4.3.2. Ограничения прямой передачи PCI Xen

Если в домене есть используемые для прямой передачи устройства PCI, такие действия, как сохранение/восстановление/миграция невозможны.

Следует по возможности отсоединить (unplug) устройство, предназначенное для передачи, перед сохранением, восстановлением или миграцией.

6.4.3.3. Ошибка «non-page-aligned MMIO BAR» при попытке запуска гостя

Добавьте следующее:

```
xen-pciback.permissive xen-pciback.hide=(08:05.0)(09:06.1)
pci=resource_alignment=08:05.0;09:06.1
```

При использовании GRUB2 и resource_alignment для нескольких устройств, необходимо оформить resource_alignment единичными кавычками:

```
'pci=resource_alignment=00:1a.7;00:1d.7'
```

В противном случае GRUB2 прочтёт строку неправильно.

6.4.3.4. Ошибка «Error: pci: 0000:02:06.0 must be co-assigned to the same guest with 0000:02:05.0» при запуске гостя

Обычно эта ошибка возникает при попытке передать только одну функцию многофункционального устройства (например, dual-port nic) или только одно из устройств за тем же мостом PCI. Это не разрешено VT- d спецификацией Intel.

Если устройство PCI является устройством с одной функцией (single-function device), попробуйте переместить его в другой слот PCI, чтобы обойти эту проблему.

6.4.3.5. Сообщение «Паника ядра - не синхронизируется: не удалось получить непрерывную область памяти для DMA из Xen» при попытке запуска гостя с iommu=soft

В данном случае может помочь ограничение максимальной памяти Dom0, устанавливаемое в параметрах загрузки менеджера виртуальных машин (/boot/grub/grub.cfg): ... dom0_mem=512M ...

Не забудьте запустить update-grub.

Если гость не запускается с iommu=SOft, попробуйте добавить earlyprintk=xen к параметрам гостевого ядра.

Гипервизор сообщает, что виртуализация IO отключена. Как включить более подробный сбор данных для поиска причины отключения?

Добавьте опцию iommu=verbose для менеджера виртуальных машин в grub.cfg и перезагрузите систему.

6.4.3.6. В оборудовании/плате имеется IOMMU, но менеджер виртуальных машин не включает аппаратно поддерживаемую виртуализацию IO

Многие материнские платы поставляются с неисправным BIOS (например, с неправильными таблицами ACPI DMAR, DRHD или RMRR), что приводит к отключению виртуализации IO в качестве меры безопасности или для предотвращения «падений» в будущем.

Если виртуализация IO отключена, но доступна на вашем оборудовании, попробуйте следующее для устранения этой проблемы:

- проверьте версию BIOS и установите последние обновления для BIOS/прошивки;
- включите IOMMU, виртуализацию IO или VT-d в BIOS и отключите питание, затем перезагрузите машину;
- установите загрузочную опцию iommu=verbose для менеджера виртуальных машин в grub.conf;
- прочтите сообщения о загрузке менеджера виртуальных машин, чтобы узнать, включена или отключена виртуализация IO;

- если менеджер виртуальных машин сообщает о неисправном BIOS, отправьте отчёт об этом поставщикам системы/платы.

6.4.3.7. Проверка поддержки устройством PCI FLR (Function Level Reset)

Запустите lspci -vv в Dom0 и проверьте, есть ли у устройства FLReset + в поле DevCap.

```
lspci -vv | grep -C 22 -i flreset+
```

Количество выдаваемых окружаемых строк следует изменить так, чтобы увидеть, к какому устройству относится флаг.

Например, в выводе команды видно, что 3 устройства в системе поддерживают FLR: 03:00.0, 04:00.0 и 03:00.1. Соответственно, при работе с ними будет использоваться соответствующий спецификациям PCI аппаратный механизм сброса.

6.4.3.8. Использование pci-stub

pci-stub может использоваться только при прямой передаче PCI с HVM-гостями, поэтому вместо него рекомендуется использовать pciback, который подходит для работы как с PV, так и с HVM-гостями.

Также в pciback реализованы специфичные для видеокарт механизмы сброса PCIe устройства, которые отсутствуют в pci-stub.

Тем не менее, остаётся один из вариантов использования pci-stub, когда нужно экранировать какое-либо устройство при загрузке без менеджера ВМ, т.е. только tk2019kernel и tk2019image.

6.4.3.9. Передача нескольких устройств PCI

При передаче нескольких устройств PCI необходимо объединить все подустройства до начала работы прямой передачи PCI.

6.5. Домены ввода-вывода

6.5.1. Драйвер-домены

Драйвер-домен — это непривилегированный домен ввода-вывода, который отвечает за определённую часть аппаратного обеспечения. Он работает на

минимальном ядре с одним конкретным аппаратным драйвером и back-end драйвером для данного класса устройств. При падении аппаратного драйвера остальные домены (включая Dom0) продолжат функционировать; после перезагрузки драйвер-домена они могут снова использовать аппаратное обеспечение.

Подробные примеры настройки драйвер-доменов приведены в п. 4.11.

6.5.1.1. Преимущества

6.5.1.1.1. Производительность

Размещение всех back-end устройств в Dom0 приводит к задержкам ответа от Dom0. Драйвер-домен позволяет разгрузить Dom0, который без него выступает в качестве «узкого места» (элемента, параметры которого ограничивают повышение производительности системы).

6.5.1.1.2. Повышенная надёжность

Аппаратные драйверы являются самой ненадёжной частью ОС, они часто подвергаются сбоям в работе. Целесообразно изолировать драйверы от других частей системы, чтобы при сбое их можно было перезапустить, не затрагивая остальные части машины.

6.5.1.1.3. Повышенная безопасность

Из-за особенностей сетевых протоколов и маршрутизации существует высокий риск возникновения ошибки в эксплуатации в сетевом пути (драйвер хоста, мост, фильтрация и т. д.). Размещая их в отдельном непривилегированном домене, можно ограничить влияние атаки на сетевой стек: даже при удачной попытке атаки у злоумышленников будет не больше доступа, чем у обычной непривилегированной ВМ.

6.5.1.1.4. Использование проприетарных драйверов и микропрограмм

Система, сертифицированная для работы с информацией, содержащей гостайну, не может включать в свой состав проприетарные драйверы или двоичные

прошивки для контроллеров. Решением этой проблемы является использование драйвер-доменов, запускаемых в режиме HVM. Данный режим является основным защищённым режимом, и используется для запуска недоверенного кода, например OS Windows. Он обеспечивает полноценную изоляцию к памяти CBT, к ресурсам PCI/PCIe, дисковым образам, и т.д. Соответственно, при запуске образа ВМ, содержащего необходимые драйвера устройств, ядра с поддержкой xen-*back модулей и библиотек пространства пользователя, можно использовать эту ВМ в качестве драйвер-домена.

6.5.1.2. Требования

Настоятельно рекомендуется устанавливать систему с современной IOMMU (AMD или VT-d версии 2). Без поддержки IOMMU не получится препятствовать драйвер-домену в использовании сетевой карты DMA или RAID-контроллера для чтения и записи любой области системной памяти. Кроме того, без поддержки IOMMU невозможно напрямую передавать устройство HVM-гостю (только PV-гостю).

В отсутствие поддержки IOMMU, можно использовать PV-домены, чтобы получить более высокую производительность, без дополнительных преимуществ стабильности и безопасности.

6.5.2. Storage-driver домены

Пример конфигурационного файла storage драйвер-домена.

```
name = "drvdom"
memory = 3072
kernel = "/home/drvdom-kernel"
cmdline = "debug console=hvc0 root=/dev/xvda init=/init"
hap = 1
disk = [ 'file:/home/drvdom.squash.img,xvda,rw']
# Intel SAS
pci=['07:00.0']
```

После запуска storage домена устройство подключается к нужной ВМ (включая dom0) с помощью команды:

```
xl block-attach 0 'format=raw, backendtype=phy, backend=drvdom, vdev=xvda, target=/dev/sda'
```

В примере выше ВМ «drvdom» используется как Storage домен, и устройство, доступное в drvdom как /dev/sda передаётся как /dev/xvda в Dom0 (0 — идентификатор домена).

6.5.3. USB-driver домен

USB-домен — непривилегированный домен, использующийся для изоляции работы с USB-устройствами по протоколу USBIP. По сути, это обычный драйвер-домен, использующий USB контроллер, и передающий поток данных с него в гостевую ВМ.

6.5.3.1. Создание конфигурационного файла домена

Конфигурационный файл USB-домена:

```
name = "usbdom"
kernel = "/boot/vmlinuz-4.9.0-ogun1-amd64"
extra = "debug earlyprintk=xen
root=/dev/xvda"
memory = 2048
vcpus = 2
vif = [ 'mac=00:16:3E:74:3d:76' ]
disk = [ 'file:/home/usbdom.img,xvda,rw' ]
rdm = "strategy=host,policy=relaxed"
pci=['00:1a.0,permissive=1','00:1d.0,permissive=1','00:14.0,permissive=1']
```

6.5.3.2. Определение идентификатора шины USB-устройства

Выведите список устройств с идентификаторами их шин.

```
usbip list -l - busid 1-1
(064f:0bd7) WIBU-Systems AG : BOX/U (064f:0bd7)
```

6.5.3.3. Предоставление доступа к USB-устройству

Предоставьте доступ к устройству, указав нужный идентификатор шины.

```
usbip bind --busid=1-1
```

6.5.3.4. Подключение USB-устройства со стороны Dom0

В гостевом домене с ОС Linux подключите USB-устройство с помощью команды, указав адрес сервера USBIP и идентификатор шины.

```
usbip attach --remote=<адрес_сервера> --busid=1-1
```

6.5.3.5. Отключение USB-устройства со стороны гостевой ВМ

usbip detach --port <порт>

6.5.4. Сетевой driver домен

Для организации сетевого драйвер-домена достаточно предоставить прямой доступ к PCI устройству сетевого контроллера и создать «мост» в драйвер-домене. Подключение гостевых машин к сетевому драйвер-домену выполняется аналогично дисковому — посредством указания параметра backend, например:

vif = ['bridge=br0, mac=00:16:3E:38:3c:01, model=e1000, backend=netdom']

6.5.5. Эмуляционный домен

Stub-домен (также эмуляционный домен) — это специализированный домен менеджера виртуальных машин, дизаггрегирующий управляющий домен Dom0. В stub-домене запускается модель устройств QEMU, связанная с доменом, работающим в режиме HVM.

Когда гостевой системе требуется выполнить операцию ввода-вывода, менеджер виртуальных машин отправляет событие модели устройств, которая выполняет эмуляцию ввода-вывода и отправляет результат эмуляции гостевой системе.

Для каждой гостевой системы в управляющем домене запускается своя модель устройств, что приводит к следующим проблемам:

- конкуренция за ресурсы между QEMU и службами Dom0;
- при наличии уязвимости в QEMU злоумышленник может получить привилегированный доступ к управляющему домену.

В схеме работы со stub-доменами QEMU выполняется в отдельном непривилегированном домене. Поэтому при наличии уязвимости в QEMU злоумышленник получит доступ только к stub-домену, который имеет такие же привилегии, что и гостевая система.

6.5.5.1. Преимущества

6.5.5.1.1. Безопасность

Stub-домен имеет меньше привилегий, чем управляющий домен Dom0. Stub-домен может взаимодействовать только со связанным с ним доменом, работающим в режиме HVM (XEN_DOMCTL_set_target).

6.5.5.1.2. Изоляция

Поскольку процессы, создаваемые QEMU для эмуляции устройств, работают в отдельном домене, они не конкурируют с другими процессами управляющего домена (моделями устройств, управляющими инструментами, обычными пользовательскими процессами). Такая изоляция позволяет снизить зависимость между производительностью гостевой системы и нагрузкой на управляющий домен.

6.5.5.1.3. Производительность

Если модель устройств работает в управляющем домене, это приводит не только к конкуренции за ресурсы с другими процессами управляющего домена, но и к отсроченному планированию: когда модели устройств требуется выполнить какую-либо работу, ей нужно дождаться, когда планировщик менеджера виртуальных машин передаст управление управляющему домену, а затем планировщик управляющего домена передаст управление процессу модели устройств.

6.5.5.1.4. Масштабируемость

Stub-домен не ограничен ресурсами, выделенными управляющему домену. Stub-домены, созданные на основе MiniOS — миниатюрной операционной системы — чрезвычайно ограничены по функциональности. Для создания безопасного, функционального и гибкого в настройке stub-домена используется система на основе Linux.

Конфигурация при использовании stub-доменов разбивается на 2 конфигурационных файла. Один файл отвечает за HVM-домен, а другой — за stub-домен. В конфигурационном файле, отвечающем за HVM-домен, при помощи

параметра stubdomain_config_file указывается имя конфигурационного файла stub-домена.

6.5.5.2. Конфигурационные параметры stub-домена

Таблица 4 - Конфигурационные параметры для stub-доменов

Параметр	Описание
kernel	Загрузить указанный файл в качестве образа ядра.
ramdisk	Загрузить указанный файл в качестве диска в памяти.
cmdline	Добавить указанную строку к строке команды ядра.
pvh	Выбирает, нужно ли запускать PV-гостя в HVM-контейнере.
hap	Включает/выключает функцию hardware assisted paging. Эта функция называется EPT (англ. Extended Page Tables) у Intel и NPT (англ. Nested Page Tables) или RVI (англ. Rapid Virtualisation Indexing) у AMD. Работает только для HVM-гостей. Если выключено, то гипервизор будет запускать гостя в режиме shadow page table, в котором обновления страничных таблиц гостя и/или операции flush над TLB будут эмулироваться. Когда HAP доступен, он будет использоваться по умолчанию.
vif	Указывает сетевые средства (как эмулируемые сетевые адаптеры, так и виртуальные интерфейсы гипервизора), предоставляемые гостю.
memory	Запустить гостя с указанным в мегабайтах количеством оперативной памяти.
maxmem	Указывает максимальное количество памяти, которое может увидеть гость. Значение параметра maxmem должно быть не меньше значения memory.
rdm	Описано в разделе 5.2 по xl.cfg
pci	Описано в разделе 5.2 по xl.cfg

Конфигурация при использовании stub-доменов разбивается на 2 конфигурационных файла. Один файл отвечает за HVM-домен, а другой — за stub-домен. В конфигурационном файле, отвечающем за HVM-домен, при помощи параметра stubdomain_config_file указывается имя конфигурационного файла stub-домена.

6.5.5.3. Конфигурационный файл HVM-гостя

builder = "hvm" name = "win" memory = 4096 vcpus = 2

```
vif = [ 'type=ioemu,ip=assigned-ip,mac=aa:00:00:00:00:11' ]
address = 'assigned-ip'
netmask = '255.255.255.XXX'
disk = [ '/dev/vg0/win10,raw,xvda,rw' ]
sdl = 0
vnc = 1
vncconsole = 1 vnclisten = "0.0.0.0"
# USB-устройства
usb = 1
usbdevice = [ 'tablet' ] device_model_stubdomain = 1 device_model_stubdomain_override = 1
device_model_version = 'qemu-xen' stubdomain_version = 'linux'
stubdomain_config_file = <путь_к_конфигурационному_файлу>
```

6.5.5.4. Конфигурационный файл stub-домена

```
memory = 1024 maxmem = 1024
stubdom_disk = "stubdom.img" kernel = "/boot/tk2019kernel"
cmdline = "debug console=hvc0 rootfstype=ext4 root=/dev/xvdz init=/init" vif =
[ 'mac=00:11:22:33:44:55' ]
```

6.5.5.5. Запуск виртуальной машины

Запуск виртуальной машины со stub-доменом осуществляется, как обычный запуск HVM-гостя:

```
xl create <путь_к_конфигурационному_файлу_hvm_гостя>
```

Скрипт, находящийся в файле rc.local в stub-домене, запустит QEMU с нужными параметрами.

6.5.6. Stub-домен в роли USB-домена для клавиатуры и мыши

При организации прямого доступа к USB-клавиатуре и USB-мыши в гостевой системе типа HVM stub-домен выполняет роль USB-домена. В stub-домене осуществляется прямой доступ к USB-контроллерам (как к PCI-устройствам), к которым будут подключаться необходимые устройства (клавиатура, мышь). Работа с этими устройствами осуществляется через QEMU (QEMU передаются параметры с идентификаторами нужных USB- устройств).

Чтобы организовать прямой доступ к PCI-устройству из stub-домена, укажите в конфигурационном файле stub-домена параметры pci и rdm, заменив указанные ниже идентификаторы устройств нужными:

```
rdm = "strategy=host,policy=relaxed"
pci = ['00:1a.0,permissive=1','00:1d.0,permissive=1','00:14.0,permissive=1' ]
```

6.6. Создание виртуальной машины на базе образа dom0

Для создания новой виртуальной машины на основе образа Dom0 (в данном случае будет использоваться имя «testvm01») необходимо выполнить следующие действия:

1) войти в систему под учётной записью Администратора (root):

```
Hypervisor Dom0 image
dom0image login: root
Password:
root@dom0image:~#
```

2) отключить режим SELinux «enforcing»:

```
root@dom0image:~# setenforce 0
```

3) создать том LVM для тестовой виртуальной машины:

```
root@dom0image:~# lvcreate -n testvm01 -l 256 vg0
  Logical volume "testvm01" created.
root@dom0image:~# lvscan
  ACTIVE       '/dev/vg0/home' [64,00 GiB] inherit
  ACTIVE       '/dev/vg0/testvm01' [1,00 GiB] inherit
```

4) произвести разбиение диска на разделы:

```
root@dom0image:~# parted -s /dev/vg0/testvm01 mklabel msdos
root@dom0image:~# parted -s /dev/vg0/testvm01 mkpart p ext2 1MiB 128MiB
root@dom0image:~# parted -s /dev/vg0/testvm01 mkpart p ext2 128MiB 100%
root@dom0image:~# parted -s /dev/vg0/testvm01 p
Model: Linux device-mapper (linear) (dm)
Disk /dev/dm-1: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number Start  End            Size            Type            File system Flags
  1       1049kB       134MB                133MB                primary
  2   134MB   1074MB       940MB                primary
```

5) создать файловую систему для загрузчика в первом разделе:

```
root@dom0image:~# mkfs.ext4 /dev/mapper/vg0-testvm01p1 mke2fs 1.44.2 (14-May-2018)
Discarding device blocks: done
Creating filesystem with 130048 1k blocks and 32512 inodes
Filesystem UUID: acd2dd78-f856-489f-a5d2-455aa89a7086
Superblock backups stored on blocks:
8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

6) скопировать образ dom0 (/boot/dom0image) во второй раздел и изменить размер ФС:

```
root@dom0image:~# lzcat /boot/dom0image >/home/dom0image
root@dom0image:~# dd if=/home/dom0image of=/dev/mapper/vg0-testvm01p2 oflag=direct
bs=1M
512+0 records in
512+0 records out
536870912 bytes (537 MB, 512 MiB) copied, 0,430438 s, 1,2 GB/s

root@dom0image:~# e2fsck -f /dev/mapper/vg0-testvm01p2
e2fsck 1.44.2 (14-May-2018)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 3A: Optimizing directories
Pass 4: Checking reference counts
Pass 5: Checking group summary information
dom0image: ***** FILE SYSTEM WAS MODIFIED *****
dom0image: 8676/32768 files (0.1% non-contiguous), 90308/131072 blocks
root@dom0image:~# resize2fs /dev/mapper/vg0-testvm01p2
resize2fs 1.44.2 (14-May-2018)
Resizing the filesystem on /dev/mapper/vg0-testvm01p2 to 229376 (4k) blocks.
The filesystem on /dev/mapper/vg0-testvm01p2 is now 229376 (4k) blocks long.
```

7) создать точку монтирования и примонтировать образ ВМ:

```
root@dom0image:~# mkdir -p /mnt
root@dom0image:~# mount /dev/mapper/vg0-testvm01p2 /mnt
```

8) создать файл /etc/fstab в образе ВМ из имеющегося в /etc/fstab, исключив монтируемые с UUID разделы:

```
root@dom0image:~# cat /etc/fstab | grep -v "auuid" >/mnt/etc/fstab
root@dom0image:~# cat /mnt/etc/fstab
# <file system> <mount pt> <type> <options> <dump> <pass>
/dev/root / ext2 rw,noauto 0 1 proc /proc proc defaults 0 0 sysfs /sys sysfs defaults 0 0
devpts /dev/pts devpts defaults,gid=5,mode=620,ptmxmode=0666 0 0
tmpfs /run tmpfs mode=0755,nosuid,nodev,rootcontext=system_u:object_r:var_run_t:s0
   0     0
tmpfs /tmp tmpfs
mode=1777,defaults,noexec,nosuid,rootcontext=system_u:object_r:tmp_t:s0 0 0
```

9) создать файл /etc/inittab в образе ВМ:

```
root@dom0image:~# cat > /mnt/etc/inittab id:3:initdefault:
 si0::sysinit:/bin/mount -t proc proc /proc si1::sysinit:/bin/mount -o remount,rw / si2::sysinit:/bin/
 mkdir -p /dev/pts si3::sysinit:/bin/mount -a -O no_netdev si4::sysinit:/bin/hostname -F /etc/
 hostname rcS:12345:wait:/etc/init.d/rcS
sole::respawn:/sbin/getty -L console 0 vt100 # GENERIC_SERIAL
shd0:06:wait:/etc/init.d/rcK
shd1:06:wait:/sbin/swapoff -a
shd2:06:wait:/bin/umount -a -r
hlt0:0:wait:/sbin/halt -dhp
reb0:6:wait:/sbin/reboot
```

Завершив ввод текста нажатием комбинации клавиш «Ctrl-D»;

10) запретить запуск ряда стартовых сценариев в образе ВМ:

```
root@dom0image:~# mkdir -p /mnt/etc/init.d/disabled root@dom0image:~# mv -v /mnt/etc/init.d/
{S07*,S10auditd,S30*,S4*,S50*,S51*,S6*,S7*} /mnt/etc/init.d/disabled '/mnt/etc/init.d/
S07selinux' -> '/mnt/etc/init.d/disabled/S07selinux' '/mnt/etc/init.d/S10auditd' -> '/mnt/etc/init.d/
disabled/S10auditd' '/mnt/etc/init.d/S30rpcbind' -> '/mnt/etc/init.d/disabled/S30rpcbind' '/mnt/etc/
init.d/S40network' -> '/mnt/etc/init.d/disabled/S40network' '/mnt/etc/init.d/S45nslcd' -> '/mnt/etc/
init.d/disabled/S45nslcd' '/mnt/etc/init.d/S49ntp' -> '/mnt/etc/init.d/disabled/S49ntp' '/mnt/etc/init.d/
S50iscsid' -> '/mnt/etc/init.d/disabled/S50iscsid' '/mnt/etc/init.d/S50nfs' -> '/mnt/etc/init.d/disabled/
S50nfs' '/mnt/etc/init.d/S50sshd' -> '/mnt/etc/init.d/disabled/S50sshd' '/mnt/etc/init.d/S51open-iscsi'
-> '/mnt/etc/init.d/disabled/S51open-iscsi' '/mnt/etc/init.d/S60xen-watchdog' -> '/mnt/etc/init.d/
disabled/S60xen-watchdog' '/mnt/etc/init.d/S60xencommons' -> '/mnt/etc/init.d/disabled/
S60xencommons' '/mnt/etc/init.d/S61xendriverdomain' -> '/mnt/etc/init.d/disabled/
S61xendriverdomain'
'/mnt/etc/init.d/S65xendomains' -> '/mnt/etc/init.d/disabled/S65xendomains' '/mnt/etc/init.d/
S70libvirtd' -> '/mnt/etc/init.d/disabled/S70libvirtd' '/mnt/etc/init.d/S71virtlogd' -> '/mnt/etc/init.d/
disabled/S71virtlogd' '/mnt/etc/init.d/S72libvirt-guests' -> '/mnt/etc/init.d/disabled/S72libvirt-
guests'
```

11) задать имя хоста:

```
root@dom0image:~# echo "testvm01" >/mnt/etc/hostname
root@dom0image:~# cat /mnt/etc/hostname
testvm01
root@dom0image:~# sed -i 's:dom0image:testvm01:' /mnt/etc/hosts
root@dom0image:~# cat /mnt/etc/hosts
127.0. 0.1      localhost
127.0. 1.1      testvm01
```

12) размонтировать /mnt:

```
root@dom0image:~# umount /mnt
```

13) создать конфигурационный файл для ВМ следующего содержания:

```
#builder='hvm' memory = 1024 vcpus = 1 name = "testvm01"
kernel = "/boot/tk2019kernel"
cmdline = "debug earlyprintk=xen console=hvc0 rootfstype=ext4 selinux=0 root=/dev/ xvda2"
disk = [ 'phy:/dev/vg0/testvm01,xvda,w']
vga='none'
sdl=0
vnc=0
```

и сохранить его как /home/testvm01.cfg.

14) запустить виртуальную машину testvm01 в паравиртуальном режиме (строка «builder="hvm"» закомментирована, используется прямая загрузка ядра /boot/tk2019kernel):

```
root@dom0image:~# xl create -c /home/testvm01.cfg Parsing config from /home/testvm01.cfg
(early) [ 0.000000] Linux version 4.9.71 (buildroot@buildroot) (gcc version 4.9.4 (Buildroot
1.00-gea4507b) ) #1 SMP Fri Jun 22 11:10:50 MSK 2018 (early) [ 0.000000] Command line:
debug earlyprintk=xen console=hvc0 rootfstype=ext4 selinux=0 root=/dev/xvda2
0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point
```

0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers' 0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers' 0.000000] x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]: 256 0.000000] x86/fpu: Enabled xstate features 0x7, context size is 832 " bytes, using 'standard' format.
(early) [0.000000] x86/fpu: Using
'eager' FPU context switches.
netconsole: network logging started hctosys: unable to open rtc device (rtc0)
EXT4-fs (xvda2): mounted filesystem with ordered

data mode. Opts:
1.192211] VFS: Mounted root (ext4 filesystem) on device 202:2.
1.192444] devtmpfs: mounted
1.192984] Freeing unused kernel memory: 1328K 1.192990] Write protecting the kernel read-only data: 12288k 1.198936] Freeing unused kernel memory: 1920K 1.199832] Freeing unused kernel memory: 1992K version 2.88 booting 1.225484] EXT4-fs (xvda2): re-mounted. Opts: (null)
Entering runlevel: 3
Loading static modules:
Loading loop ...
Loading tun ...
Loading usb_common ...
Loading usbcore .
Loading ehci_pci Loading ehci_hcd Loading xhci_pci Loading xhci_hcd Loading usb_storage ...
Loading md_mod ...
Loading raid0 ...
Loading raid1 ... done Populating /dev using udev:
Add subsystems...
Add devices...
Settling udev...
Add raid volumes, if any:
mdadm: No arrays found in config file or automatically mdadm: No arrays found in config file or automatically done
Init LVM volumes, if any: done
Setting up UTF-8 cyrillic console: OK Mounting local filesystems...done.
Activating swapfile swap...done.
Cleaning up temporary files
Starting logging: OK Starting rsyslog daemon: OK Starting imbalance: OK
Initializing random number generator... done.
Cleaning up temporary files
Starting cron ... done.
Starting local settings script ... done.
Hypervisor Dom0 image testvm01 login:

15) войти в виртуальную машину с правами Администратора (root) - пароль идентичный:

Hypervisor Dom0 image
testvm01 login: root
Password:
root@testvm01:~#

16) создать запись в /etc/fstab для первого раздела диска ВМ:

```
root@testvm01:~# echo "/dev/xvda1 /boot ext4 defaults,noatime,nodiratime,discard 0 1" >>/etc/
fstab
root@testvm01:~# cat /etc/fstab
# <file system> <mount pt> <type> <options> <dump> <pass>
/dev/root / ext2 rw,noauto 0 1 proc /proc proc defaults 0 0 sysfs /sys sysfs defaults 0 0
devpts /dev/pts devpts defaults,gid=5,mode=620,ptmxmode=0666 0 0
tmpfs /run tmpfs mode=0755,nosuid,nodev,rootcontext=system_u:object_r:var_run_t:s0
   0      0
tmpfs /tmp tmpfs
mode=1777,defaults,noexec,nosuid,rootcontext=system_u:object_r:tmp_t:s0 0 0 /dev/xvda1 /boot
ext4 defaults,noatime,nodiratime,discard 0 1
```

17) примонтировать /boot и проверить:

```
root@testvm01:~# mount /boot
root@testvm01:~# df -h | grep /boot
/dev/xvda1      119M 1,6M 109M 2% /boot
```

18) установить загрузчик GRUB на первый раздел дискового образа

```
root@testvm01:~# grub-install /dev/xvda
Выполняется установка для платформы i386-pc.
Установка завершена. Ошибок нет.
root@testvm01:~#
```

19) создать сценарий загрузки GRUB2 поместить его в файл /boot/grub/ grub.cfg при помощи команды:

```
root@testvm01:~# cat >/boot/grub/grub.cfg <<EOF
> insmod gzio
> insmod xzio
> insmod lzopio
> insmod part_gpt
> insmod part_msdos
> insmod ext2
> insmod search
>
> set default=0
> set timeout=0
>
> menuentry "VM" {
> linux /tk2019kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4 selinux=0 nomodeset
> }
> EOF
root@testvm01:~#
```

В данном случае после первого EOF вводится содержимое файла (система предваряет ввод каждой строки символом «>», после ввода второго EOF и нажатия «Enter» файл создастся с введённым содержимым.

20) проверить содержимое файла /boot/grub/grub.cfg:

```
root@testvm01:~# cat /boot/grub/grub.cfg
insmod gzio
insmod xzio
```

```
insmod lzopio
insmod part_gpt
insmod part_msdos
insmod ext2
insmod search

set default=0
set timeout=0

menuentry "VM" {
linux /tk2019kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4 selinux=0 nomodeset }
root@testvm01:~#
```

21) теперь виртуальная машина готова к запуску в режиме HVM. Необходимо сначала выключить ВМ командой:

```
root@testvm01:~# shutdown -h now
Broadcast message from root@testvm01 (console) (Fri Jun 22 18:41:41 2018):
The system is going down for system halt NOW!
INIT: Switching to runlevel: 0
Local shutdown actions... done.
Stopping cron ...done.
Saving random seed... done.
Stopping irqbalance: OK
Stopping rsyslog daemon: OK
Stopping logging: OK
Unmounting temporary filesystems...done.
Deactivating swap...done.
Unmounting local filesystems...done.
[ 1166.526084] reboot: System halted
root@dom0image:~#
```

Из вывода команды выше видно, что управление вернулось в Гипервизор.

22) поменять режим работы виртуальной машины на HVM, проверить результат:

```
root@dom0image:~# sed -i 's:#builder:builder:' /home/testvm01.cfg
root@dom0image:~# sed -i 's:kernel:#kernel:' /home/testvm01.cfg
root@dom0image:~# sed -i 's:cmdline:#cmdline:' /home/testvm01.cfg
root@dom0image:~# cat /home/testvm01.cfg
builder='hvm'
memory = 1024
vcpus = 1
name = "testvm01"
#kernel = "/boot/tk2019kernel"
#cmdline = "debug earlyprintk=xen console=hvc0 rootfstype=ext4 selinux=0 root=/dev/ xvda2"
disk = [ 'phy:/dev/vg0/testvm01,xvda,w']
vga='none'
sdl=0
vnc=0
```

Из вывода команд выше видно, что теперь установлен режим работы HVM (builder='hvm') и закомментированы строки прямой загрузки ядра и его аргументов.

23) скопировать ядро dom0image в раздел загрузки дискового образа ВМ:

```
root@dom0image:~# mount /dev/mapper/vg0-testvm01p1 /mnt
root@dom0image:~# cp -v /boot/tk2019kernel /mnt '/boot/tk2019kernel' -> '/mnt/tk2019kernel'
root@dom0image:~# umount /mnt
```

24) запустить виртуальную машину:

```
root@dom0image:~# xl create -c /home/testvm01.cfg
Parsing config from /home/testvm01.cfg
[ 0.000000] Linux version 4.9.71 (buildroot@buildroot) (gcc version 4.9.4 (Buildroot 1.00-gea4507b) ) #1 SMP Fri Jun 22 11:10:50 MSK 2018
[ 0.000000] Command line: BOOT_IMAGE=/tk2019kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4 selinux=0 nomodeset [ 0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers'
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[ 0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
[ 0.000000] x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]: 256 [ 0.000000] x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, 'standard' format.
x86/fpu: Using 'eager' FPU context switches. e820: BIOS-provided physical RAM map:
BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved BIOS-e820: [mem 0x0000000000100000-0x000000003fffefff] usable BIOS-e820: [mem 0x000000003ffff000-0x000000003fffffff] reserved BIOS-e820: [mem 0x00000000fc000000-0x00000000ffffffff] reserved NX (Execute Disable) protection: active
SMBIOS 2.4 present.
[ 0.000000] Hypervisor detected: Xen Xen version 4.8.
[ 0.000000] Netfront and the Xen platform PCI driver have been compiled for this kernel: unplug emulated NICs.
[ 0.000000] Blkfront and the Xen platform PCI driver have been compiled for this kernel: unplug emulated disks.
[ 0.000000] You might have to change the root device
[ 0.000000] from /dev/hd[a-d] to /dev/xvd[a-d]
[ 0.000000] in your root= kernel command line option
[ 0.000000] e820: last_pfn = 0x3ffff max_arch_pfn = 0x400000000
[ 0.000000] x86/PAT: Configuration [0-7]: WB WC UC- UC WB WC UC- WT
[ 0.000000] found SMP MP-table at [mem 0x000f25e0-0x000f25ef] mapped at [ffff9b01000f25e0]
[ 0.348332] random: fast init done
[ 0.356782] blkfront: xvda: flush diskcache: enabled; persistent grants: enabled; indirect descriptors: enabled;
xvda: xvda1 xvda2 console [netcon0] enabled netconsole: network logging started
rtc_cmos 00:02: setting system clock to 2018-06-22 15:55:29 UTC EXT4-fs (xvda2): mounted filesystem with ordered data mode. Opts:
[ 0.450257] VFS: Mounted root (ext4 filesystem) readonly on device 202:2. 0.450388] devtmpfs: mounted
[ 0.450903] Freeing unused kernel memory: 1328K 0.529855] Write protecting the kernel read-only data: 12288k 0.530201] Freeing unused kernel memory: 1920K 0.533383] Freeing unused kernel memory: 1992K INIT: version 2.88 booting
[ 0.551241] EXT4-fs (xvda2): re-mounted. Opts: (null)
[ 0.561059] EXT4-fs (xvda1): mounted filesystem with ordered data mode. Opts: discard INIT: Entering runlevel: 3

Loading static modules:
```

```
Loading loop ...
Loading tun ...
Loading usb_common ...
Loading usbcore .
Loading ehci_pci Loading ehci_hcd Loading xhci_pci Loading xhci_hcd Loading usb_storage ...
Loading md_mod ...
Loading raid0 ...
Loading raid1 ... done Populating /dev using udev:
Add subsystems...
Add devices...
Settling udev...
Add raid volumes, if any:
mdadm: No arrays found in config file or automatically mdadm: No arrays found in config file or
automatically done
Init LVM volumes, if any: done
Setting up UTF-8 cyrillic console: OK
Mounting local filesystems...done.
Activating swapfile swap...done.
Cleaning up temporary files
Starting logging: OK Starting rsyslog daemon: OK
Starting irqbalance: OK
Initializing random number generator... done.
Cleaning up temporary files....
Starting cron ... done.
Starting local settings script ... done.

Hypervisor Dom0 image
testvm01 login:
```

Из вывода команды видно, что ядро загружено как на обычной ЭВМ, но при этом использует паравиртуальные драйверы, поскольку обнаружен Гипервизор.

25) зайти с учётной записью Администратора (root):

```
Hypervisor Dom0 image
testvm01 login: root
Password:
root@testvm01:~#
```

26) проверить, что виртуальная машина загружена в режиме HVM и находит эмулируемые устройства - чипсет(i440FX), дисковый (IDE) контроллер, и т.д.:

```
root@testvm01:~# lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
00:02.0 Unassigned class [ff80]: XenSource, Inc. Xen Platform Device (rev 01)
```

27) выключить ВМ:

```
root@testvm01:~# shutdown -h now
Broadcast message from root@testvm01 (console) (Fri Jun 22 19:04:08 2018):
The system is going down for system halt NOW!
INIT: Switching to runlevel: 0
```

```
Local shutdown actions... done.
Stopping cron ...done.
Saving random seed... done.
Stopping irqbalance: OK
Stopping rsyslog daemon: OK
Stopping logging: OK
Unmounting temporary filesystems...done.
Deactivating swap...done.
Unmounting local filesystems...done.
[ 521.966963] reboot: Power down
```

С помощью команд, приведённых выше, подготовлена тестовая виртуальная машина testvm01.

6.7. Клонирование виртуальной машины со сменой имени

Для клонирования виртуальной машины testvm01 и смены имени «клона» на testvm02 необходимо выполнить следующие действия:

1) сделать копию тома LVM тестовой машины testvm01, назвав его testvm02:

```
root@dom0image:~# lvcreate -n testvm02 -l 256 vg0
Logical volume "testvm02" created.
root@dom0image:~# dd if=/dev/vg0/testvm01 of=/dev/vg0/testvm02 bs=1M oflag=direct 1024+0
records in
1024+0 records out
1073741824 bytes (1,1 GB, 1,0 GiB) copied, 1,26638 s, 848 MB/s
```

Первая команда создаёт том testvm02 идентичного testvm01 размера, вторая команда делает копию testvm01 в testvm02.

2) просканировать разделы на блочном устройстве /dev/vg0/ testvm02 - дисковом образе ВМ testvm02, и убедиться, что они обнаружены:

```
root@dom0image:~# parted /dev/vg0/testvm02 p
Model: Linux device-mapper (linear) (dm)
Disk /dev/dm-4: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number Start End Size Type File system Flags
  0     1049kB 134MB 133MB primary ext4
  1     134MB 1074MB 940MB primary ext4
root@dom0image:~# partprobe /dev/mapper/vg0-testvm02
root@dom0image:~# ls /dev/mapper/vg0-testvm02*
/dev/mapper/vg0-testvm02 /dev/mapper/vg0-testvm02p1 /dev/mapper/vg0-testvm02p2
```

3) примонтировать корневой раздел с образа диска testvm02 и изменить имя хоста:

```
root@dom0image:~# mount /dev/mapper/vg0-testvm02p2 /mnt
root@dom0image:~# echo "testvm02" >/mnt/etc/hostname
```

```
root@dom0image:~# cat /mnt/etc/hostname testvm02
root@dom0image:~# sed -i 's:testvm01:testvm02:' /mnt/etc/hosts root@dom0image:~# cat /mnt/
etc/hosts
127.0. 0.1      localhost
127.0. 1.1      testvm02
```

4) размонтировать корневой раздел с образа диска testvm02:

```
root@dom0image:~# umount /mnt
```

5) скопировать конфигурационный файл ВМ testvm01.cfg в testvm02.cfg с изменением имени ВМ и пути к образу диска, проверить корректность изменения:

```
root@dom0image:~# cat /home/testvm01.cfg | sed 's:testvm01:testvm02:' >/home/ testvm02.cfg
root@dom0image:~# cat /home/testvm02.cfg
builder='hvm'
memory = 1024
vcpus = 1
name = "testvm02"
# kernel = "/boot/tk2019kernel"
# cmdline = "debug earlyprintk=xen console=hvc0 rootfstype=ext4 selinux=0 root=/dev/ xvda2"
disk = [ 'phy:/dev/vg0/testvm02,xvda,w']
vga='none'
sdl=0
vnc=0
```

6) запустить виртуальную машину testvm02:

```
root@dom0image:~# xl create -c /home/testvm02.cfg Parsing config from /home/testvm02.cfg
[ 0.000000] Linux version 4.9.71 (buildroot@buildroot) (gcc version 4.9.4 (Buildroot 1.00-
gea4507b) ) #1 SMP Fri Jun 22 11:10:50 MSK 2018

Mounting local filesystems...done.
Activating swapfile swap...done.
Cleaning up temporary files
Starting logging: OK Starting rsyslog daemon: OK Starting irqbalance: OK
Initializing random number generator... done. Cleaning up temporary files....
Starting cron ... done.
Starting local settings script ... done.
Hypervisor Dom0 image
testvm02 login:
```

7) войти в тестовую машину testvm02 с учётной записью Администратора, убедиться в том, что она запущена в HVM режиме (видны эмулируемые устройства), а затем выключить:

```
Hypervisor Dom0 image testvm02
login: root Password:
root@testvm02:~# lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
```

```
00:02.0 Unassigned class [ff80]: XenSource, Inc. Xen Platform Device (rev 01) root@testvm02:~#
shutdown -h now
Broadcast message from root@testvm02 (console) (Fri Jun 22 19:29:37 2018):
The system is going down for system halt NOW!
INIT: Switching to runlevel: 0
root@testvm02:~# Local shutdown actions... done.
Stopping cron ...done.
Saving random seed... done.
Stopping irqbalance: OK
Stopping rsyslog daemon: OK
Stopping logging: OK
Unmounting temporary filesystems...done.
Deactivating swap...done.
Unmounting local filesystems...done.
[ 160.582474] reboot: Power down
root@dom0image:~#
```

Из вывода команд выше видно, что после выключения ВМ управление вернулось к консоли менеджера виртуальных машин.

Описанный выше способ — клонирование тома ВМ и замена параметров конфигурации — может быть использован для создания дополнительных ВМ на основе образца по мере необходимости.

6.8. Настройка менеджера томов LVM

По умолчанию программный модуль «Синергия-ТК» устанавливается таким образом, что он занимает всё свободное пространство на целевом носителе. Это позволяет локально хранить инсталляционные образы дисков ПО, образы дисков с данными, но намного чаще требуется использование блочных устройств для образов дисков виртуальных машин (как минимум, по соображениям производительности ВМ). Порядок настройки менеджера томов LVM для проведения тестирования приведён ниже. В примере используется блочное устройство /dev/ nvme0n1 — твердотельный SSD NVMe накопитель. На другом оборудовании блочное устройство, использованное при установке программного модуля «Синергия-ТК», может отличаться. Приведённая ниже процедура предполагает размер накопителя, на который произведена инсталляция, не менее 240 Гб.

1) войти в систему под учётной записью администратора (root):

```
Hypervisor Dom0 image tk2019image
login: root
Password:
root@tk2019image:~#
```

2) отключить «форсированный» режим SELinux:

root@tk2019image:~# setenforce 0

3) размонтировать /home и убедиться, что операция выполнена:

root@tk2019image:~# umount /home
root@tk2019image:~# df Filesystem 1K-blocks Used Available Use% Mounted on
/dev/root 499656 335296 148636 70% /
devtmpfs 1639152 0 1639152 0% /dev
tmpfs 1670656 380 1670276 1% /run
tmpfs 1670656 4 1670652 1% /tmp
/dev/nvme0n1p2 999576 73276 858308 8% /boot
/dev/nvme0n1p3 8191416 51820 7703784 1% /var
tmpfs 5120 0 5120 0% /run/lock
tmpfs 668260 0 668260 0% /run/shm

4) проверить разбиение диска, и поменять тип раздела 4 на lvm:

root@tk2019image:~# parted /dev/nvme0n1 u MiB p Model: Unknown (unknown)
Disk /dev/nvme0n1: 488386MiB
Sector size (logical/physical): 512B/512B
Partition Table: gpt

5) преобразовать раздел 4 в физический том LVM (на другом оборудовании UUID и размеры могут отличаться):

root@tk2019image:~# wipefs /dev/nvme0n1p4
DEVICE OFFSET TYPE UUID LABEL
nvme0n1p4 0x438 ext4 356a4fc3-f071-44a4-97bf-39997fd9158c dom0home
root@tk2019image:~# wipefs -a /dev/nvme0n1p4
/dev/nvme0n1p4: 2 bytes were erased at offset 0x00000438 (ext4): 53 ef root@tk2019image:~# pvcreate /dev/nvme0n1p4
Physical volume "/dev/nvme0n1p4" successfully created.
root@tk2019image:~# pvs
PV VG Fmt Attr PSize PFree
lvm2 --- 467,94g 467,94g

6) создать группу томов vg0 а в ней том home размером 64GB:

root
root@tk2019image:~# vgcreate vg0 /dev/nvme0n1p4
Volume group "vg0" successfully created
root@tk2019image:~# lvcreate -n home -l 16384 vg0
Logical volume "home" created.
root@tk2019image:~# lvs
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
home vg0 -wi-a 64,00g

7) создать в томе home файловую систему ext4, определить UUID, обновить /etc/fstab и примонтировать /home:

root@tk2019image:~# mkfs.ext4 -L dom0home -E nodiscard /dev/mapper/vg0-home mke2fs 1.44.2 (14-May-2018)

```
Creating filesystem with 16777216 4k blocks and 4194304 inodes Filesystem UUID: adf27af7-
076d-4ed3-b96e-d8f3fd8a7222 Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
4096000, 7962624, 11239424
Allocating group tables: done
Writing inode tables: done
Creating journal (131072 blocks): done
Writing superblocks and filesystem accounting information: done
root@tk2019image:~# cat /etc/fstab | grep -v "/home" >/etc/fstab.new root@tk2019image:~#
printf "UUID=%s\t/home
\text4\tdefaults,noexec,noatime,nodiratime,discard\t0 2\n" adf27af7-076d-4ed3-b96e-
d8f3fd8a7222 >>/etc/fstab.new root@tk2019image:~# cat /etc/fstab.new
# <file system> <mount pt> <type> <options> <dump> <pass>
/dev/root / ext2 rw,noauto 0 1 proc /proc proc defaults 0 0 sysfs /sys sysfs defaults 0 0
devpts /dev/pts devpts defaults,gid=5,mode=620,ptmxmode=0666 0 0
tmpfs /run tmpfs mode=0755,nosuid,nodev,rootcontext=system_u:object_r:var_run_t:s0
   0      0
tmpfs /tmp tmpfs
mode=1777,defaults,noexec,nosuid,rootcontext=system_u:object_r:tmp_t:s0 0 0
UUID=1593c050-f11f-4f54-97dd-1a1e90963e24 /boot ext4 defaults,noatime,nodiratime 0 2
UUID=e96b3026-5884-4a26-891e-9b087b18f763 /var ext4 defaults,noatime,nodiratime 0 2
UUID=adf27af7-076d-4ed3-b96e-d8f3fd8a7222 /home ext4
defaults,noexec,noatime,nodiratime,discard 0 2 root@tk2019image:~# cat /etc/fstab.new >/etc/
fstab root@tk2019image:~# mount /home root@tk2019image:~# df -h Filesystem /dev/root
devtmpfs tmpfs tmpfs
/dev/nvme0n1p2 /dev/nvme0n1p3 tmpfs tmpfs
/dev/mapper/vg0-home 63G
```

8) восстановить контексты безопасности, сохранить настройки системы и перезагрузить тестовую ЭВМ:

```
root@tk2019image:~# restorecon -R /
root@tk2019image:~# ./save_dom0img.sh
Найден образ tk2019image по пути /boot/tk2019image...
Распаковка образа tk2019image:
/boot/tk2019image (1/1)
100 %  56,4 MiB / 512,0 MiB = 0,110 129 MiB/s    0:03
tk2019image примонтирован в /tmp/tmp.MDx848SyAP.
Сохранение изменений...
Перемаркировка ФС tk2019image мандатными метками в соответствии с конфигурацией...
Размонтирование служебных ФС и образа...
Сжатие образа tk2019image:
/tmp/tk2019image (1/1)
100 %  56,9 MiB / 512,0 MiB = 0,111 4,1 MiB/s     2:06
Сохраняем контрольные суммы: /boot/tk2019image /boot/tk2019kernl /boot/ hypervisor.gz /
boot/policy.mls готово.
Изменения в настройках системы сохранены в /boot/tk2019image...
root@tk2019image:~# shutdown -r now
Broadcast message from root@tk2019image (pts/1) (Thu Jun 13 01:54:30 2018): The system is
going down for reboot NOW!
```

9) после перезагрузки, зайти в систему с учётной записью Администратора (root) и проверить корректность настроек после перезагрузки:

```
root@tk2019image# lvs
467,94 GiB 4,00 MiB 119792
16384 / 64,00 GiB 103408 / 403,94 GiB
TMYBiz-yFxk-mRAU-5wbO-Vp3m-heKe-ImJnnq
```

Как видно из вывода команды выше, размер /home стал 63GiB (64GiB за вычетом служебной информации LVM), в списке логических томов присутствует том «home», с остатком свободного места в группе томов 403.94 GiB. Этого свободного пространства уже достаточно и для хранения ISO образов с ПО, и для создания блочных устройств, которые потребуются для запуска ВМ.

6.9. Сценарии настройки в программном модуле «Синергия-ТК»

Ниже приводится описание примеров настроек, присутствующих в программном модуле «Технология «тонкого клиента» «Синергия-ТК».

6.9.1. Настройка сетевого моста

Практики безопасного использования систем виртуализации предполагают разделение управляющих интерфейсов и сетевых интерфейсов, используемых виртуальными машинами. Для работы в АСЗИ с обработкой информации с различными уровнями конфиденциальности, необходимо использовать несколько физических интерфейсов, подключаемых, соответственно, в различные физические сегменты сети. Таким образом, в рекомендуемой конфигурации сервер должен иметь как минимум 2 сетевых интерфейса — один для управления программным модулем «Синергия-ТК» и доступа Администратора, и второй (или более) — для подключения виртуальных машин.

По умолчанию в образе tk2019image включается только первый сетевой интерфейс (управляющий), с получением адреса по DHCP.

Для настройки сетевого моста рекомендуется использовать сценарий /etc/rc.local, фрагмент сценария, выполняющего настройку моста xenbr0, включающего интерфейс ethl, приведён ниже:

```
### В примере ниже автоматически создаётся сетевой мост для виртуальных машин.
if ! brctl show ${bridge} 2>/dev/null 1>/dev/null ; then
```

```
    printf "^Создаем бридж ${bridge}…"
    brctl addbr ${bridge}
    printf " done.\n"
  else
    printf "\nБридж ${bridge} уже создан:^"
    brctl show ${bridge}
  fi

  ifconfig ${bridge} up 2>/dev/null

  if ifconfig ${bridge_iface} 2>/dev/null 1>/dev/null ; then
    if ! brctl show ${bridge} 2>/dev/null | grep ${bridge_iface} ; then
      printf "Добавляем интерфейс ${bridge_iface} в бридж ${bridge}…"
      brctl addif ${bridge} ${bridge_iface}
      printf " готово.\n"
    else
      printf "Интерфейс ${bridge_iface} уже в бридже ${bridge}.\n"
      brctl show ${bridge}
    fi
    ifconfig ${bridge_iface} up 2>/dev/null 1>/dev/null
  fi
```

Данный сценарий также исключает повторное добавление интерфейса и создание моста при повторном запуске сценария.

6.9.2. Настройка сетевых блочных устройств

В сценарии /etc/rc.local также приведён пример настройки сетевых устройств, использующих протокол iSCSI — как по стандартному Ethernet, так и через Infiniband — через IPoIB.

Для использования данного фрагмента сценария необходимо определить iSCSI target в его начале, и IP адреса сервера, доступного через IB, либо сервера, доступного через EThernet.

В примере используются настройки, валидные для лаборатории Разработчика - сервер доступен как через IPoIB по адресу 10.0.1.1, так и по Ethernet по адресу 192.168.1.1.

```
############## Параметры настройки ####################
#target=iqn.2017-09.net.rus-soft.farmstorage:farmslave01
target=iqn.2018-05.info.wakizashi.spb.nas:hypervisor.home
# Имя моста для гипервизора
bridge=xenbr0
# Имя интерфейса для моста
bridge_iface=eth1
# Имя сетевого интерфейса IP over IB
ipoib_iface=ib0
```

```
# IP сервера через IB
server_ib="10.0.1.1"
# IP сервера через Ethernet
server_ip="192.168.1.1"
...

##### Ожидаем интерфейс IPoIB:
if ifconfig ${ipoib_iface} 2>/dev/null 1>/dev/null ; then
  printf "Включаем интерфейс ${ipob_iface}…"
  ifup ${ipoib_iface} 2>/dev/null
  declare -i counter counter=0
  while [ $counter -le 10 ] ; do
    ifconfig ${ipoib_iface} 2>/dev/null | grep -e "UP,.*RUNNING" 1>/dev/null && break
    let counter++
    printf "."
    sleep 1
  done
  printf " готово.\n"
else
  server_ib=""
fi
############ Монтирование сетевых ФС:
# Создаём каталог /home: mkdir -p /home

# Далее идёт пример настройки сетевых ресурсов по IP over Infiniband.
# Если сервер доступен - используем его адрес, если нет - обычный IPv4:

if test -e "$server_ib"; then # Если на предыдущих шагах был найден интерфейс:
  # Выдержим паузу до появления коннекта - IB поднимается долго
  printf "Проверяем доступность сервера через IPoIB... "
  declare -i counter counter=0
  while [ $counter -le 20 ] ; do
    ping -W 2 -c1 $server_ib 2>/dev/null 1>/dev/null && break
    let counter++
    printf "."
    sleep 1
  done
  printf " done.\n"
  if ping -W 2 -c1 $server_ib ; then
    # IPoIB приоритетнее:
    server_ip=$server_ib
  fi
fi

# Если сервер доступен:
if ping -W 2 -c1 $server_ip ; then
  # iSCSI target - специфичен для хоста.
  if test -b /dev/disk/by-path/ip-${server_ip}:3260-iscsi-${target}-lun-0 ; then
    device='readlink /dev/disk/by-path/ip-${server_ip}:3260-iscsi-${target}-lun-0' &&
    mount /dev/'basename $device' /home
  else
    iscsiadm -m discovery -p ${server_ip} -t st 2>/dev/null | grep $target &&
```

```
  iscsiadm -m node -p ${server_ip} -T $target -l && udevadm settle &&
  device='readlink /dev/disk/by-path/ip-${server_ip}:3260-iscsi-${target}-lun-0' &&
  mount /dev/'basename $device' /home
 fi
 # Подмонтируем серевые устройства:
 mount -a -O _netdev
fi
```

В данном фрагменте происходит ожидание линка IPoIB на интерфейсе ib0 (при его наличии), проверяется доступность сервера по IBoIP, и если он не доступен в течении заданного времени ожидания, используется адрес Ethernet.

После определения IP адреса сервера происходит поиск соответствующего target-a, и подключение к нему.

В случае использования устройств iSCSI как томов виртуальных машин в АСЗИ, следует назначить подключённому устройству (в сценарии — device) соответствующую мандатную метку.

6.9.3. Автоматическое обновление ключей SSH

Для обеспечения идентичности настроек различных серверов виртуальных машин, приводится пример централизованного обновления ключей доступа SSH (также в сценарии /etc/rc.local:

```
# Обновим ключи SSH для доступа к хосту:
if test -f /home/images/keys/authorized_keys ; then
  mkdir -p /root/.ssh
  cat /home/images/keys/authorized_keys >/root/.ssh/authorized_keys
fi
```

6.9.4. Автоматический запуск виртуальных машин

В том же сценарии (/etc/rc.local) производится также и автоматический запуск виртуальных машин:

```
# Запускаем виртуальные машины, конфигурации которых находятся в /etc/xen/vms:
for i in /etc/xen/vms/vm??* ;do
  # Игнорируем сломанные симлинки (если есть).
  [ ! -f "$i" ] && continue
  case "$i" in
    *.cfg)
    xl create $i ;;
    *)
    echo "Неопознанный файл $i в директории автозапуска ВМ!"
    ;;
  esac
```

done

6.10. Установка ОС в виртуальную машину

Ниже приводится краткая инструкция по установке ОС в виртуальную машину, включая подготовку к работе с PCI Passthrough и паравиртуальными драйверами.

6.10.1. Подготовка виртуального диска и инсталляционного носителя

Для установки системы необходимо определить требуемый объем дискового пространства для целевой ОС. Например, в MS Windows, с учётом того, что не требуется использование режима гибернации, объёма 64 GiB для системного диска более чем достаточно. ОС семейства UNIX/Linux требуют значительно меньше дискового пространства, для типовой инсталляции достаточно 16 GiB системного раздела.

Для создания виртуального диска рекомендуется использовать подсистему LVM (в случае локального хранения данных), либо сетевые монтируемые тома iSCSI, FibreChannel, SRP/iWARP — при наличии соответственно настроенной инфраструктуры хранения данных.

Ниже рассматривается вариант с созданием локального устройства для виртуальных дисков 2-х ВМ — с ОС Windows и ОС Linux (Ubuntu).

1) Для создания группы томов LVM необходимо выполнить шаги, предусмотренные в п. 4.8;

2) Для создания тома, на который будет производиться инсталляция целевой ОС, необходимо выполнить команду lvcreate -n <имя_тома> -l <размер> <имя_группы томов>. В случае выполнения п. 1, имя группы томов - vgO, размер указывается в блоках 4K, для 64GiB это будет 16384, для 16GiB - 4096 и зададим имена томов ВМ - vm01 и vmQ2:

```
root@tk2019image:~# lvcreate -n vm01 -l 16384 vg0
 Logical volume "vm01" created.
root@tk2019image:~# lvcreate -n vm01 -l 16384 vg0
 Logical volume "vm02" created.
root@tk2019image:~# lvscan
 ACTIVE        '/dev/vg0/home' [64,00 GiB] inherit
 ACTIVE        '/dev/vg0/vm01' [64,00 GiB] inherit
```

```
 ACTIVE        '/dev/vg0/vm02' [16,00 GiB] inherit
```

3) Для установки ОС необходимо скопировать образ ISO инсталляционного диска в локальную систему. В примере ниже используется инсталляционный диск демонстрационной версии Windows 8.1 Enterprise (с тестовым периодом 90 дней, свободно доступный для загрузки на сайте Microsoft) и Ubuntu 16.04 MATE.

Образ диска разместим в каталоге /home/ISO:

```
root@tk2019image:~# ls -la /home/ISO total 3868640
drwxr-xr-x. 2 root root          4096 июн 28 22:07 .
drwxr-xr-x. 9 root root          4096 июн 28 22:07 ..
-r--r--r--. 1 root root 3961473024 ноя 26 2014 WIN8.1-Ent-EVAL.ISO
-r--r--r--. 1 root root 1728053248 ноя 15 07:35 ubuntu-mate-16.04.3-desktop- amd64.iso
```

6.10.2. Настройка ВМ с ОС Windows

Для настройки ВМ необходимо выполнить следующее:

1) перед запуском ВМ необходимо убедиться, что создан сетевой мост xenbr0 (см. п. 4.9.1):

```
root@tk2019image:~# brctl show
bridge name bridge id STP enabled interfaces
xenbr0 8000.0cc47a98b3dd no eth1
```

2) создадим конфигурационный файл домена /home/vm01-win81ent.cfg со следующим содержимым:

```
builder = "hvm"
memory = 16384
name = "vm01-win81ent"
vcpus = 8
mmio_hole = 3072
machine = 'q35'
vif = [
# Только виртуальный адаптер. Требует установленных PV драйверов в госте.
# 'bridge=xenbr0, mac=00:16:3e:38:3c:01, type=vif, vifname=vif-vm01'
# И виртуальный, и эмулируемый. При инсталляции гостя без PVHVM драйверов
# использовать эту строку.
'bridge=xenbr0, mac=00:16:3e:38:3c:01'
]
disk = [
'raw:/dev/vg0/vm01,hda,w',
'file:/home/ISO/WIN8.1-Ent-EVAL.ISO,hdc:cdrom,r',
]
# Первым грузим CDROM - инсталляция boot='cd'
usb = 1
usbdevice = [ 'tablet' ]
# VNC options vnc = 1
```

```
# Первый экран - соответствует номеру ВМ.
vncdisplay = "1"
# Доступен для внешних клиентов VNC.
vnclisten = "0.0.0.0"
# Triggers
on_poweroff = 'destroy'
on_reboot = 'restart'
on_crash = 'destroy'
vga = 'stdvga'
serial=['stdio']
keymap="en-us"
localtime = 1
# Для виндовс - нужно включить:
viridian = [ "all" ]
hap = 1
acpi = 1
acpi_s3 = 0
acpi_s4 = 0
apic = 1
hpet = 1
pae = 1
nx = 1
pci_power_mgmt = 1
pci_msitranslate = 1
pci_passthrough = 1
pci_permissive = 1
xen_platform_pci = 1
xen_extended_power_mgmt = 1
ioports=["3b0-3df"]
pci_seize = 1 pci_power_mgmt = 1
# PCI passthrough - nVidia M2000 gfx_passthru = 0
#pci = [
# '02:00.0', '02:00.1',
# '00:1a.0',
# '00:1d.0',
# '00:14.0',
# '07:00.0',
#]
# Для passthrough USB контроллера:
#rdm = "strategy=host,policy=relaxed"
```

В приведённом выше конфигурационном файле закомментированы строки, которые понадобятся далее, для организации прямого доступа к GPU.

6.10.3. Запуск ВМ и установка ОС

1) Запустить виртуальную машину.

2) После запуска виртуальной машины, примерно через 3-5 секунд необходимо подключиться к ней по протоколу VNC, используя IP адрес

установленного СВТ (в примере ниже подключение производится из АРМ с Linux, адрес ЭВМ с установленным программным модулем «Синергия-ТК» — 192.168.0.184): $ vncviewer 192.168.0.184:1;

После некоторого ожидания, на экране появится следующее изображение (рис. 3).

С другой версией ОС изображение будет другим.

3) Следуя инструкциям инсталлятора ОС, установить систему. В процессе установки гостевая ОС может несколько раз перезагружаться, и соединение с VNC будет обрываться. В этом случае нужно подключиться заново при помощи команды vncviewer, указанной выше.

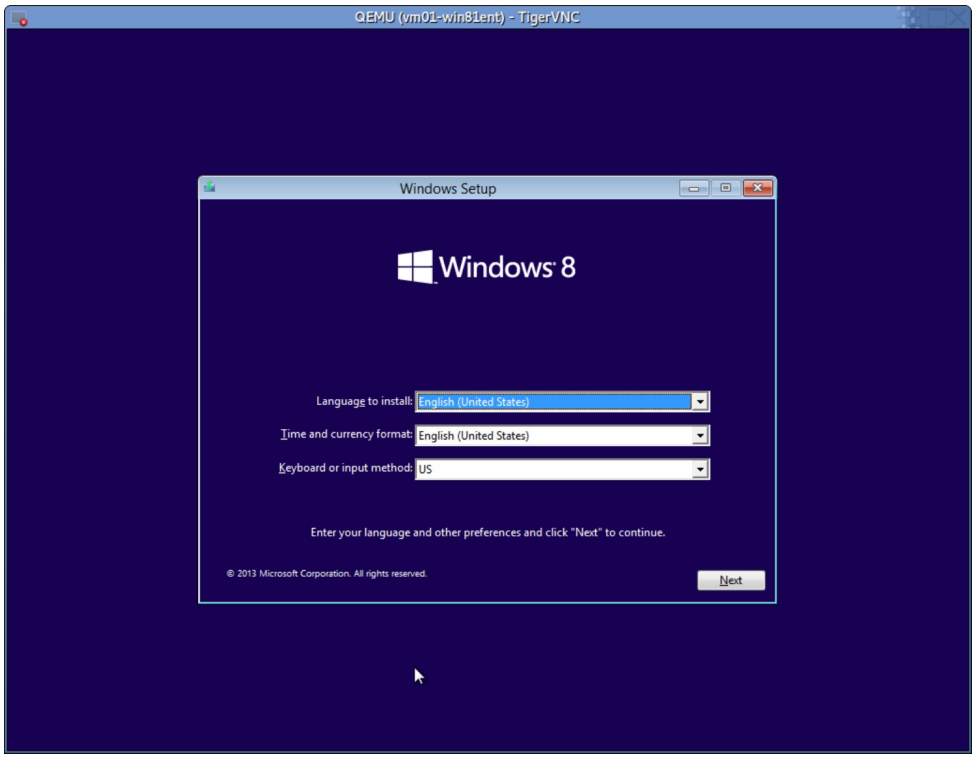

Рисунок 3

4) После установки системы, при входе через VNC, получаем следующее изображение (рис. 4).

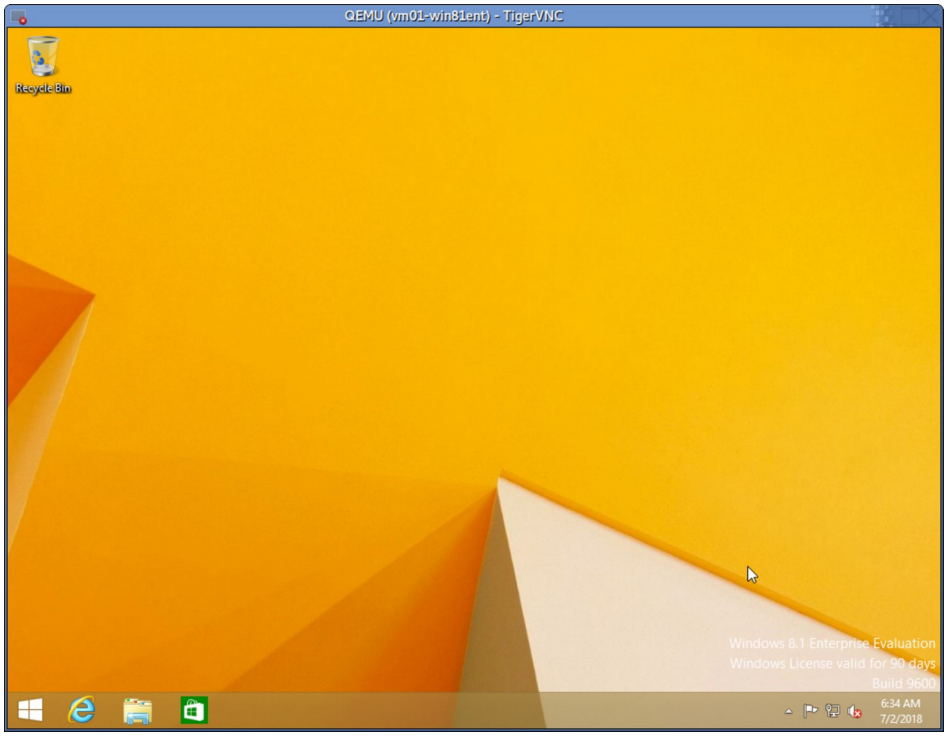

Рисунок 4

6.10.4. Установка виртуальной машины с операционной системой Linux

Для установки ВМ с ОС Ubuntu Linux необходимо выполнить следующие действия:

1) создать конфигурационный файл ВМ /home/vm02-ubuntu.cfg со следующим содержимым:

```
builder = "hvm"
memory = 16384
name = "vm02-ubuntu"
vcpus = 8
mmio_hole = 3072
machine = 'q35'
vif = [
# Только виртуальный адаптер. Требует установленных PV драйверов в госте.
'bridge=xenbr0, mac=00:16:3e:38:3c:01, type=vif, vifname=vif-vm01'
]
disk = [
'raw:/dev/vg0/vm02,hda,w',
'file:/home/lSO/ubuntu-mate-16.04.3-desktop-amd64.iso,hdc:cdrom,r',
]
# Первым грузим CDROM - инсталляция boot='cd'
usb = 1
usbdevice = [ 'tablet' ]
# VNC options vnc = 1
# Первый экран - соответствует номеру ВМ.
vncdisplay = "2"
# Доступен для внешних клиентов VNC.
```

```
vnclisten = "0.0.0.0"
# Triggers
on_poweroff = 'destroy'
on_reboot = 'restart'
on_crash = 'destroy'
vga = 'stdvga'
serial=['stdio']
keymap="en-us"
localtime = 1
hap = 1
acpi = 1
acpi_s3 = 0
acpi_s4 = 0
apic = 1
hpet = 1
pae = 1
nx = 1
pci_power_mgmt = 1
pci_msitranslate = 1
pci_passthrough = 1
pci_permissive = 1
xen_platform_pci = 1
xen_extended_power_mgmt = 1
ioports=["3b0-3df"]
pci_seize = 1
pci_power_mgmt = 1
# PCI passthrough - nVidia M2000 gfx_passthru = 0
#pci = [
# '03:00.0', '03:00.1',
# '00:1a.0',
# '00:1d.0',
# '00:14.0',
# '07:00.0',
#]
# Для passthrough USB контроллера:
#rdm = "strategy=host,policy=relaxed"
```

2) запустить виртуальную машину:

```
root@tk2019image:~# xl create /home/vm02-ubuntu.cfg
Parsing config from /home/vm02-ubuntu.cfg
```

3) после запуска виртуальной машины, примерно через 3-5 секунд необходимо подключиться к ней по протоколу VNC, используя IP адрес установленного СВТ (в примере ниже подключение производится из АРМ с Linux, адрес ЭВМ с установленным программным модулем «Синергия-ТК» — 192.168.0.184):

```
$ vncviewer 192.168.0.184:2;
```

4) после загрузки ВМ с инсталляционного носителя, при входе через VNC, получаем следующее изображение (рис. 5);

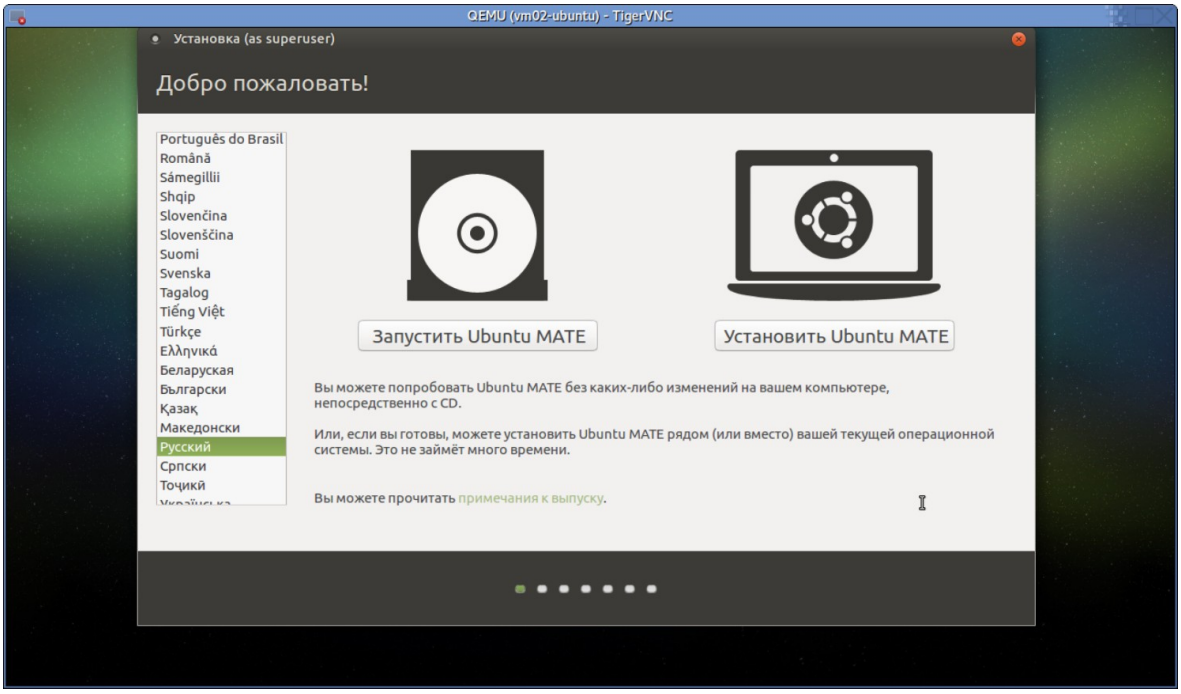

Рисунок 5

5) следуя инструкциям инсталлятора, установить ОС;

6) после перезагрузки гостевой ОС, подключения через VNC и входа в систему, можно увидеть следующее изображение (рис. 6).

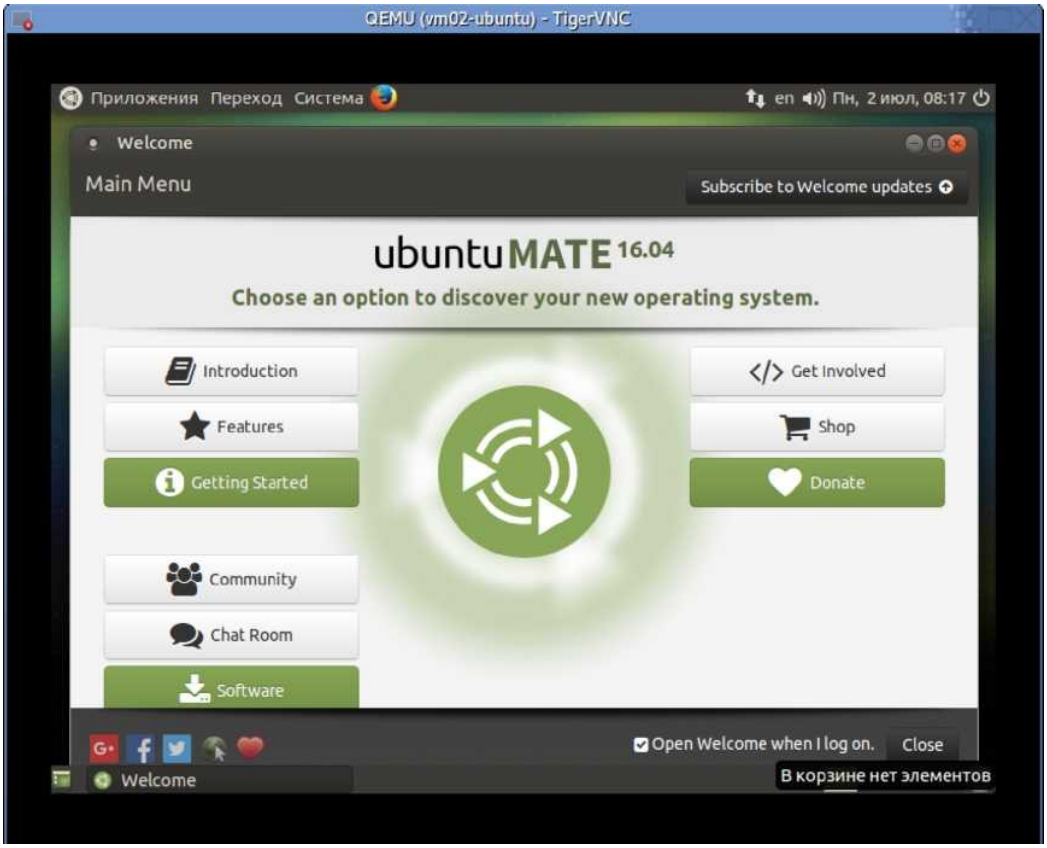

Рисунок 6

Современные ОС семейства Linux не требуют дополнительной специальной настройки для работы с программным модулем «Синергия-ТК».

6.10.5. Установка паравиртуальных драйверов устройств

Для наиболее эффективной работы гостевой системы рекомендуется установить паравиртуальные драйверы устройств ввода-вывода, предоставляемые программным модулем «Синергия-ТК». Паравиртуальные драйверы позволяют обращаться к устройствам напрямую, используя гипервызовы, и минуя эмуляцию, реализуемую при помощи QEMU. Программный модуль «Синергия-ТК» обеспечивает совместимость с XEN, соответственно, подходят драйверы для XEN.

В большинстве ОС семейства UNIX/Linux паравиртуальные драйверы уже включены, но для Windows их нужно загружать с внешнего ресурса. Для ОС Windows паравиртуальные драйверы можно скачать по адресу https://www.xenproject.org/downloads/windows-pv-drivers/winpv-drivers-8/winpv-drivers-821.html

Для установки паравиртуальных драйверов в ОС Windows выполним следующие действия:

1) на АРМ администратора скачать необходимые драйверы и создать ISO образ с ними;

2) скопировать ISO образ с драйверами на ЭВМ с установленным программным модулем «Синергия-ТК»;

3) в конфигурационном файле /home/vm01-win81ent.cfg;

4) поменять путь к устройству hdc с инсталляционного диска ОС на ISO с драйверами. Секция дисков может выглядеть следующим образом:

```
disk = [
'raw:/dev/vg0/vm01,hda,w',
'file:/home/ISO/HypervisorDRV.iso,hdc:cdrom,r',
# 'file:/home/ISO/WIN8.1-Ent-EVAL.ISO,hdc:cdrom,r',
]
```

где /home/ISO/HypervisorDRV.iso - созданый образ ISO с драйверами;

5) выключить ВМ средствами гостевой ОС (в данный момент ОС еще не может принимать команды от гипервизора, выключение доступно только через ACPI);

6) запустить ВМ, убедиться, что «CDROM» диск с драйверами подключён;

7) в зависимости от ОС, установить паравиртуальные драйверы.

6.11. Примеры настройки драйвер-доменов

Ниже приводится описание примеров настроек, присутствующих в программном модуле «Технология «тонкого клиента» «Синергия-ТК».

6.11.1. Настройка образа драйвер-домена

Для настройки дискового драйвер-домена необходимо создать виртуальную машину, как описано в Раздел 5.6, «Создание виртуальной машины на базе образа dom0», с учётом указания другого имени ВМ.

Например, drvdom, выполнив стандартные этапы создания домена на основе tk2019image:

1) вход в систему под учётной записью Администратора (root):

```
Hypervisor Dom0 image tk2019image login: root Password:
root@tk2019image:~#
```

2) отключение режима SELinux «enforcing»:

```
root@tk2019image:~# setenforce 0
```

3) создание тома LVM для драйвер-домена:

```
root@tk2019image:~# lvcreate -n drvdom -l 256 vg0 Logical volume "drvdom" created.
root@tk2019image:~# lvscan | grep drvdom
 ACTIVE            '/dev/vg0/drvdom' [1,00 GiB] inherit
```

4) разбиение диска на разделы:

```
root@tk2019image:~# parted -s /dev/mapper/vg0-drvdom mklabel msdos root@tk2019image:~#
parted -s /dev/mapper/vg0-drvdom mkpart p ext2 1MiB 128MiB root@tk2019image:~# parted -s /
dev/mapper/vg0-drvdom mkpart p ext2 128MiB 100% root@tk2019image:~# parted -s /dev/
mapper/vg0-drvdom u MiB p Model: Linux device-mapper (linear) (dm)
Disk /dev/dm-3: 1074MB
Sector size (logical/physical): 512B/512B Partition Table: msdos Disk Flags:
Number Start End Size Type File system Flags
   0      1049kB 134MB 133MB primary
   1      134MB 1074MB 940MB primary
```

5) создание файловой системы для загрузчика в первом разделе:

root@tk2019image:~# mkfs.ext4 -L boot /dev/mapper/vg0-drvdom1 mke2fs 1.44.2 (14-May-2018)
Discarding device blocks: done
Creating filesystem with 130048 1k blocks and 32512 inodes Filesystem UUID: f2e92020-3bba-4030-9685-ffe34e81c139 Superblock backups stored on blocks:
8193, 24577, 40961, 57345, 73729
Allocating group tables: done Writing inode tables: done Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

6) скопировать образ dom0 (/boot/tk2019image) во второй раздел и изменить размер ФС:

root@tk2019image:~# lzcat /boot/tk2019image | dd of=/dev/mapper/vg0-drvdom2 oflag=direct bs=1M 512+0 records in 512+0 records out
536870912 bytes (537 MB, 512 MiB) copied, 0,430438 s, 1,2 GB/s
root@tk2019image:~# e2fsck -f /dev/mapper/vg0-drvdom2 e2fsck 1.44.2 (14-May-2018)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 3A: Optimizing directories
Pass 4: Checking reference counts
Pass 5: Checking group summary information
tk2019image: ***** FILE SYSTEM WAS MODIFIED *****
tk2019image: 8676/32768 files (0.1% non-contiguous), 90308/131072 blocks
root@tk2019image:~# resize2fs /dev/mapper/vg0-drvdom2 resize2fs 1.44.2 (14-May-2018)
Resizing the filesystem on /dev/mapper/vg0-testvm01p2 to 229376 (4k) blocks.
The filesystem on /dev/mapper/vg0-testvm01p2 is now 229376 (4k) blocks long.

7) создание точки монтирования и монтирование раздела драйвер- домена:

root@tk2019image:~# mkdir -p /mnt/drvdom
root@tk2019image:~# mount /dev/mapper/vg0-drvdom2 /mnt/drvdom

8) создание файла /etc/fstab:

root@tk2019image:~# cat >/mnt/drvdom/etc/fstab
<file system> <mount pt> <type> <options> <dump> <pass> /dev/root / ext2 rw,discard,noauto 0 1 proc /proc proc defaults 0 0 sysfs /sys sysfs defaults 0 0
devpts /dev/pts devpts defaults,gid=5,mode=620,ptmxmode=0666 0 0 tmpfs /run tmpfs mode=0755,nosuid,nodev 0 0 tmpfs /tmp tmpfs mode=1777,defaults,noexec,nosuid 0 0 /dev/xvda1 /boot ext4 defaults,noatime,nodiratime,discard 0 1

9) создание файла /etc/inittab:

root@tk2019image:~# cat > /mnt/drvdom/etc/inittab id:3:initdefault:
si0::sysinit:/bin/mount -t proc proc /proc si1::sysinit:/bin/mount -o remount,rw / si2::sysinit:/bin/mkdir -p /dev/pts si3::sysinit:/bin/mount -a -O no_netdev si4::sysinit:/bin/hostname -F /etc/hostname rcS:12345:wait:/etc/init.d/rcS
sole::respawn:/sbin/getty -L console 0 vt100 # GENERIC_SERIAL
shd0:06:wait:/etc/init.d/rcK
shd1:06:wait:/sbin/swapoff -a
shd2:06:wait:/bin/umount -a -r
hlt0:0:wait:/sbin/halt -dhp
reb0:6:wait:/sbin/reboot

10) запрет запуска ряда стартовых сценариев:

```
root@tk2019image:~# mkdir -p /mnt/etc/init.d/disabled root@tk2019image:~# mv -v /mnt/
drvdom/etc/init.d/
{S07*,S10auditd,S30*,S4*,S50*,S51*,S7*} /mnt/drvdom/etc/init.d/disabled '/mnt/etc/init.d/
S07selinux' -> '/mnt/etc/init.d/disabled/S07selinux' '/mnt/etc/init.d/S10auditd' -> '/mnt/etc/init.d/
disabled/S10auditd'
'/mnt/etc/init.d/S30rpcbind' -> '/mnt/etc/init.d/disabled/S30rpcbind' '/mnt/etc/init.d/S40network' ->
'/mnt/etc/init.d/disabled/S40network' '/mnt/etc/init.d/S45nslcd' -> '/mnt/etc/init.d/disabled/
S45nslcd' '/mnt/etc/init.d/S49ntp' -> '/mnt/etc/init.d/disabled/S49ntp' '/mnt/etc/init.d/S50iscsid' -> '/
mnt/etc/init.d/disabled/S50iscsid'
'/mnt/etc/init.d/S50nfs' -> '/mnt/etc/init.d/disabled/S50nfs' '/mnt/etc/init.d/S50sshd' -> '/mnt/etc/
init.d/disabled/S50sshd' '/mnt/etc/init.d/S51open-iscsi' -> '/mnt/etc/init.d/disabled/S51open-iscsi' '/
mnt/etc/init.d/S70libvirtd' -> '/mnt/etc/init.d/disabled/S70libvirtd' '/mnt/etc/init.d/S71virtlogd' -> '/
mnt/etc/init.d/disabled/S71virtlogd' '/mnt/etc/init.d/S72libvirt-guests' -> '/mnt/etc/init.d/disabled/
S72libvirt-guests'
```

В отличие от примера в Раздел 4.6, «Создание виртуальной машины на базе образа dom0», сценарии xen остаются — они необходимы для работы драйвер-домена.

11) задание имени хоста:

```
root@tk2019image:~# echo "drvdom" >/mnt/drvdom/etc/hostname root@tk2019image:~# cat /
mnt/drvdom/etc/hosts
 127.0. 0.1 drvdom
```

12) размонтировать /mnt/drvdom:

```
root@tk2019image:~# umount /mnt/drvdom
```

13) создать конфигурационный файл для драйвер-домена:

```
root@tk2019image:~# cat > /home/drvdom.cfg
#builder='hvm'
memory = 1024
vcpus = 1
name = "drvdom"
kernel = "/boot/tk2019kernel"
cmdline = "debug earlyprintk=xen console=hvc0 rootfstype=ext4 selinux=0 root=/dev/ xvda2"
disk = [ 'phy:/dev/vg0/drvdom,xvda,w']
driverdomain = 1
vga='none'
sdl=0
vnc=0
usb=0
localtime = 1
hap = 1
acpi = 1
acpi_s3 = 0
acpi_s4 = 0
apic = 1
hpet = 1
pae = 1
nx = 1
```

```
pci_power_mgmt = 1
pci_msitranslate = 1
pci_passthrough = 1
pci_permissive = 1
xen_platform_pci = 1
xen_extended_power_mgmt = 1
```

14) запустить виртуальную машину testvm01 в паравиртуальном режиме (строка «builder="hvm"» закомментирована, используется прямая загрузка ядра /boot/tk2019kernel):

```
root@tk2019image:~# xl create -c /home/drvdom.cfg
Hypervisor Dom0 image drvdom login: root Password:
root@sasdom:~# blkid
/dev/xvda1: LABEL="boot" UUID="f2e92020-3bba-4030-9685-ffe34e81c139" TYPE="ext4"
PARTUUID="ffaca1bf-01"
/dev/xvda2: LABEL="tk2019image" UUID="053c9cec-afdf-473a-b559-146fe30870da"
TYPE="ext4" PARTUUID="ffaca1bf-02"
```

15) запретить запуск чего либо в rc.local:

```
root@drvdom:~# printf '#!/bin/bash\nexit 0' >/etc/rc.local
```

16) добавить строку для /boot:

```
root@drvdom:~# echo "/dev/xvda1 /boot ext4 defaults,noatime,nodiratime,discard 0 1" >>/etc/
fstab
```

17) примонтировать /boot и установить GRUB:

```
root@drvdom:~# mount /boot
root@drvdom:~# grub-install /dev/xvda
Выполняется установка для платформы i386-pc.
Установка завершена. Ошибок нет.
root@drvdom:~# cat >/boot/grub/grub.cfg
insmod gzio
insmod xzio
insmod lzopio
insmod part_gpt
insmod part_msdos
insmod ext2
insmod search
set default=0 set timeout=0
menuentry "drvdom" { linux /tk2019kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4
selinux=0 nomodeset
}
```

18) выключим ВМ:

```
root@drvdom:~# shutdown -h now
Broadcast message from root@drvdom (console) (Wed Jul 4 12:10:43 2018):
The system is going down for system halt NOW!
INIT: Switching to runlevel: 0 Local shutdown actions... done.
Stopping cron ...done.
Saving random seed... done.
Stopping irqbalance: OK Stopping rsyslog daemon: OK
Stopping logging: OK
```

```
Unmounting temporary filesystems...done. Deactivating swap...done.
Unmounting local filesystems...done.
[ 321.124286] reboot: System halted
```

19) скопируем ядро dom0:

```
root@tk2019image:~# mount /dev/mapper/vg0-drvdom1 /mnt/drvdom/ root@tk2019image:~# cp
-va /boot/tk2019kernel /mnt/drvdom/
'/boot/tk2019kernel' -> '/mnt/drvdom/tk2019kernel'
root@ws00:~# umount /mnt/drvdom
```

20) закомментировать строки kernel = и cmdline =, раскомментировать строку #builder='hvm' в файле конфигурации ВМ, получив следующее его содержимое:

```
builder='hvm'
memory = 1024
vcpus = 1
name = "drvdom"
#kernel = "/boot/tk2019kernel"
#cmdline = "debug earlyprintk=xen console=hvc0 rootfstype=ext4 selinux=0 root=/dev/ xvda2"
disk = [ 'phy:/dev/vg0/drvdom,xvda,w']
driverdomain = 1
vga='none'
sdl=0
vnc=0
usb=0
localtime = 1
hap = 1
acpi = 1
acpi_s3 = 0
acpi_s4 = 0
apic = 1
hpet = 1
pae = 1
nx = 1
pci_power_mgmt = 1
pci_msitranslate = 1
pci_passthrough = 1
pci_permissive = 1
xen_platform_pci = 1
xen_extended_power_mgmt = 1
```

Теперь ВМ готова к запуску в PVH/HVM режимах.

6.11.2. Настройка дискового драйвер-домена

В данном варианте настраивается драйвер-домен для SAS HBA контроллера (англ. Serial Attached SCSI Host Bus Adapter), соответственно имя домена будет «sasdom». Для настройки дискового драйвер-домена следует выполнить следующее:

1) создать блочное устройство для диска драйвер-домена:

```
root@ws00:~# lvcreate -n sasdom -L 1GiB vg0
 Logical volume "sasdom" created.
```

2) скопировать содержимое созданного ранее шаблона драйвер-домена:

```
root@ws00:~# dd if=/dev/mapper/vg0-sasdom of=/dev/mapper/vg0-usbdom bs=1M
```

3) примонтировать блочное устройство дискового домена:

```
root@ws00:~# partprobe /dev/mapper/vg0-sasdom
root@ws00:~# mkdir /mnt/sasdom
root@ws00:~# mount /dev/mapper/vg0-sasdom1 /mnt/sasdom
```

4) задать имя хоста:

```
boot@ws00:~# echo "sasdom" >/mnt/sasdom/etc/hostname
boot@ws00:~# echo "127.0.0.1 sasdom localhost" >/mnt/sasdom/etc/hosts
```

5) размонтировать том, скопировать конфигурационный файл ВМ-шаблона, и добавить в него нужный дисковый контроллер:

```
boot@ws00:~# umount /mnt/sasdom
boot@ws00:~# cp /home/drvdom.cfg /home/sasdom.cfg
root@ws00:~# lspci | grep SAS
0b:00.0 Serial Attached SCSI controller: LSI Logic / Symbios Logic SAS2308 PCI- Express
Fusion-MPT SAS-2 (rev 05) root@ws00:~# xl pci-assignable-list | grep 0b:00 0000:0b:00.0
root@ws00:~# cat >>/home/sasdom.cfg pci = [
'0b:00.0'
]
```

6) запустить дисковый драйвер-домен и проверить наличие дисков, подключённых к контроллеру (в данном случае это два 1TB диска):

```
root@ws00:~# xl create -c /home/sasdom.cfg Parsing config from /home/sasdom.cfg
[ 0.000000] Linux version 4.9.88 (buildroot@buildroot) (gcc version 7.3.0 (Buildroot 1.01) ) #1
SMP Thu Jun 28 08:00:00 MSK 2018
[ 0.000000] Command line: BOOT_IMAGE=/tk2019kernel console=hvc0 root=/dev/xvda2
rootfstype=ext4 selinux=0 nomodeset
INIT: version 2.88 booting
[ 0.806415] EXT4-fs (xvda2): re-mounted. Opts: (null)
[ 0.816772] EXT4-fs (xvda1): mounted filesystem with ordered data mode. Opts: discard INIT:
Entering runlevel: 3

Loading static modules:
 Loading loop ...
 Loading tun ...
 Loading usb_common ...
 Loading usbcore ...
 Loading ehci_pci ...
 Loading ehci_hcd ...
 Loading xhci_pci ...
 Loading xhci_hcd ...
 Loading usb_storage ...
 Loading md_mod ...
```

```
Loading raid0 ...
Loading raid1 ... done Populating /dev using udev:
Add subsystems...
Add devices...
Settling udev...
Add raid volumes, if any:
mdadm: No arrays found in config file or automatically
mdadm: No arrays found in config file or automatically
done

Init LVM volumes, if any: done
Setting up UTF-8 cyrillic console: OK
Mounting local filesystems...done.
Activating swapfile swap...done.
Cleaning up temporary files
Starting logging: OK
Starting rsyslog daemon: OK
Starting irqbalance: OK
Initializing random number generator... done.
Cleaning up temporary files
Starting xl devd...
Starting cron ... done.
Starting local settings script ... done.
Hypervisor Dom0 image sasdom login: root
```

7) настроить RAID и LVM, а затем выключить ВМ:

```
root@sasdom:~# parted -s /dev/sda mklabel gpt
root@sasdom:~# parted -s /dev/sdb mklabel gpt
root@sasdom:~# parted -s /dev/sda mkpart RAID0A 1MiB 100%
root@sasdom:~# parted -s /dev/sdb mkpart RAID0B 1MiB 100%
root@sasdom:~# parted -s /dev/sda set 1 raid on
root@sasdom:~# parted -s /dev/sdb set 1 raid on
root@sasdom:~# mdadm -C /dev/md0 -n 2 -l stripe /dev/sda1 /dev/sdb1
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
root@sasdom:~# ./mkconf.sh >/etc/mdadm/mdadm.conf
root@sasdom:~# pvcreate /dev/md0
 Physical volume "/dev/md0" successfully created.
root@sasdom:~# sed -i 's@/dev/md/0@/dev/md0@g' /etc/mdadm/mdadm.conf
root@sasdom:~# vgcreate vg.sas /dev/md0
 Volume group "vg.sas" successfully created
root@sasdom:~# lvcreate -n lvol00 -l 16384 vg.sas
 Logical volume "lvol00" created.
root@sasdom:~# lvscan
 ACTIVE        '/dev/vg.sas/lvol00' [64,00 GiB] inherit
root@sasdom:~# poweroff
WARNING: could not determine runlevel - doing soft poweroff
(it's better to use shutdown instead of poweroff from the command line)

Broadcast message from root@sasdom (console) (Thu Jul 2 12:41:05 2018):

The system is going down for system halt NOW!
INIT: Switching to runlevel: 0
```

root@sasdom:~# Local shutdown actions... done.
Stopping cron ...done.
Stopping xl devd...
Saving random seed... done.
Stopping irqbalance: OK
Stopping rsyslog daemon: OK
Stopping logging: OK
Unmounting temporary filesystems...done.
Deactivating swap...done.
Unmounting local filesystems...done.
Waiting for MD arrays to become idle...done.
[328.571794] reboot: Power down

8) снова запустить ВМ и убедиться в наличии созданного тома LVM:

root@ws00:~# xl crea -c /home/sasdom.cfg Parsing config from /home/sasdom.cfg
[0.000000] Linux version 4.9.88 (buildroot@buildroot) (gcc version 7.3.0 (Buildroot 1.01)) #1 SMP Thu Jun 28 08:00:00 MSK 2018 [0.000000] Command line: BOOT_IMAGE=/tk2019kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4 selinux=0 nomodeset
INIT: version 2.88 booting
[0.695564] EXT4-fs (xvda2): re-mounted. Opts: (null)
[0.706214] EXT4-fs (xvda1): mounted filesystem with ordered data mode. Opts: discard INIT: Entering runlevel: 3
Loading static modules: Loading loop ...
Loading tun ...
Loading usb_common ...
Loading usbcore ...
Loading ehci_pci ...
Loading ehci_hcd ...
Доading xhci_pci ...
Loading xhci_hcd ...
Loading usb_storage ...
Loading md_mod ...
Loading raid0 ...
Loading raid1 ... done
Populating /dev using udev:
Add subsystems...
Add devices...
Settling udev...
Add raid volumes, if any:
done
Init LVM volumes, if any:
  0 logical volume(s) in volume group "vg.sas" now active done
Setting up UTF-8 cyrillic console: OK
Mounting local filesystems...done.
Activating swapfile swap...done.
Cleaning up temporary files....
Starting logging: OK Starting rsyslog daemon: OK
Starting irqbalance: OK
Initializing random number generator... done.
Cleaning up temporary files
Starting xl devd...
Starting cron ... done.
Starting local settings script ... done.

```
Hypervisor Dom0 image sasdom login: root Password:
root@sasdom:~# lvscan
ACTIVE          '/dev/vg.sas/lvol00' [64,00 GiB] inherit
root@sasdom:~#
```

9) завершить сеанс работы в консоли (комбинация клавиш «Ctrl+]») и проверить доступность блочного устройства для остальных доменов (проверить блочные устройства Dom0, затем подключить том из sasdom, проверить его доступность, и отключить):

```
root@ws00:~# xl block-attach 0
'format=raw,backendtype=phy,backend=sasdom,vdev=xvda,target=/dev/vg.sas/lvol00'
root@ws00:~# lsblk

root@ws00:~# xl block-list 0
Vdev BE handle state evt-ch ring-ref 51712 6 0       4 78 8
root@ws00:~# xl block-detach 0
51712 root@ws00:~# xl block-list 0
Vdev BE handle state evt-ch ring-ref BE-path
root@ws00:~# lsblk
```

Таким образом, на основе шаблона был создан дисковый драйвер-домен (или «сторадж-домен», из которого можно экспортировать блочные устройства в остальные домены, включая Dom0.

Аналогичным образом настраиваются сетевые и USB домены, а также любые их комбинации (например, сторадж-домен SAN, экспортирующий блочные устройства, доступные в выделенном для SAN сегменте Ethernet).

7. БЕЗОПАСНАЯ НАСТРОЙКА ТИПОВЫХ КОНФИГУРАЦИЙ

Разделы ниже приводят описание процедуры настройки типовых конфигураций, соответствующих этим вариантам, но не ограничивающих использование гипервизора только приведенными конфигурациями.

7.1. Типовая конфигурация 1 - автоматический сервер ВМ

Данная конфигурация обеспечивает вариант применения гипервизора, установленного на СВТ, в качестве сервера виртуальных машин, работающего в автоматическом режиме, для обеспечения функционирования информационных систем предприятия.

Для безопасной настройки автоматического сервера ВМ необходимо выполнить следующие действия:

- подготовить менеджер томов для создания образов виртуальных машин в соответствии с подразделом 6.8;
- создать сетевые мосты для требуемых сегментов сетей с различными уровнями конфиденциальности согласно подразделу 6.3 и п. 6.9.1;
- произвести установку одной или нескольких гостевых ОС в соответствии с подразделом 6.10;
- согласно п. 6.9.4, настроить автоматический запуск виртуальных машин.

При корректной настройке системы, виртуальные машины будут автоматически запущены при старте СВТ.

7.2. Типовая конфигурация 2 - АРМ инженера с прямым доступом к оборудованию (GPU)

Данная конфигурация обеспечивает вариант применения гипервизора, установленного на СВТ, в качестве инженерного АРМ, работающего с одним или несколькими пользователями в интерактивном режиме с запуском виртуальных машин, использующих возможности прямого доступа к специальному оборудованию СВТ — например, видеокартам, применяемым для расчетов и работы с графическим ПО (САПР, моделирование физических процессов, и т. д.).

Для безопасной настройки АРМ инженера необходимо выполнить следующие действия:

- подготовить менеджер томов для создания образов виртуальных машин в соответствии с подразделом 6.8;
- создать сетевые мосты для требуемого сегмента сетей с соответствующими уровнями конфиденциальности, согласно подразделу 6.3 и п. 6.9.1;
- произвести установку гостевой ОС в соответствии с подразделом 6.10;
- определить устройство (GPU), для которого требуется прямой доступ из ВМ, и добавить соответствующие параметры в конфигурационный файл ВМ, согласно подразделу 6.4, например:

ioports=["3b0-3df"] pci_seize = 1 pci_power_mgmt = 1 # PCI passthrough - nVidia M2000 gfx_passthru = 0 pci = ["03:00.0’, "03:00.1’, # "00:1a.0’, # "00:1d.0’, # "00:14.0’, # "07:00.0’,] # Для passthrough USB контроллера: #rdm = "strategy=host,policy=relaxed"

- произвести установку необходимых драйверов устройств в виртуальной машине согласно описанию ОС виртуальной машины и оборудования, к которому осуществляется прямой доступ;
- согласно п. 6.9.4 настроить автоматический запуск виртуальных машин.

При корректной настройке системы виртуальные машины с прямым доступом к оборудованию будут автоматически запущены при старте СВТ.

7.3. Типовая конфигурация 3 - интерактивное АРМ разработчика

Данная конфигурация обеспечивает следующий вариант применения гипервизора: интерактивное АРМ разработчика, работающего с несколькими виртуальными машинами в фоновом режиме, с возможностью переноса данных между ВМ в виде дисковых образов на отчуждаемых носителях.

В случае с АРМ разработчика у пользователя имеется доступ к средствам разработки и отладки ПО и оборудования, соответственно, должны быть приняты дополнительные меры для обеспечения безопасности при работе в АСЗИ.

В большинстве случаев рекомендуется использование такого АРМ как однопользовательской изолированной системы.

Для безопасной настройки АРМ разработчика необходимо выполнить следующие действия:

- подготовить менеджер томов для создания образов виртуальных машин в соответствии с подразделом 6.8;
- создать сетевые мосты для требуемых сегментов сетей с различными уровнями конфиденциальности согласно подразделу 6.3 и п. 6.9.1;
- произвести установку одной или нескольких гостевых ОС в соответствии с подразделом 6.10;
- завести соответствующую учетную запись для пользователя — разработчика;
- проинициализировать отчуждаемые устройства, настроить соответствующие записи в /etc/fstab для сопоставления пользователя с устройством.

8. НАСТРОЙКА ОРГАНИЗАЦИИ ДОСТУПА К ВИРТУАЛЬНЫМ МАШИНАМ С ПОМОЩЬЮ КОМПЛЕКСА СРЕДСТВ УПРАВЛЕНИЯ

Настройка производится в соответствии с п. 2.4.8 Руководства пользователя по эксплуатации 07623615.00439-10 90 01.

ПЕРЕЧЕНЬ ТЕРМИНОВ

Виртуализация	—	предоставление набора вычислительных ресурсов или их логического объединения, абстрагированное от аппаратной реализации, и обеспечивающее при этом логическую изоляцию друг от друга вычислительных процессов, выполняемых на одном физическом ресурсе
Виртуальная машина	—	программная система, эмулирующая аппаратное обеспечение целевой платформы и исполняющая программы для целевой платформы на хост-платформе
Гипервизор	—	менеджер виртуальных машин, обеспечивающий одновременное, параллельное выполнение нескольких виртуальных машин на одном и том же компьютере (хосте)
Дистрибутив	—	форма распространения программного обеспечения
Домен	—	запущенная копия виртуальной машины
Домен ввода-вывода	—	служебный домен (виртуальная машина), в котором запущена специализированная ОС, обеспечивающая работу с оборудованием хост-системы, либо внешним оборудованием
Тонкий клиент	—	компьютер или программа-клиент в сетях с клиент-серверной или терминальной архитектурой, который переносит все или большую часть задач по обработке информации на сервер
Хост	—	устройство, предоставляющее сервисы формата «клиент-сервер» в режиме сервера по каким-либо интерфейсам и уникально определённое на этих интерфейсах
Dom0	—	первый запускаемый домен (виртуальная машина), который автоматически создаётся и загружается сразу после загрузки и инициализации гипервизора. Этот домен имеет особые права на управление гипервизором. По умолчанию всё аппаратное обеспечение хост-системы доступно из

		Dom0
Ethernet	—	технология организации локальных сетей
Intel VT	—	технология виртуализации Intel на платформе x86, ранее известная под кодовым названием «Vanderpool»
Intel VT-d	—	технология Intel для виртуализации IOMMU
Linux	—	ядро операционной системы, может использоваться как общее наименование операционных систем, основанных на данном ядре

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

ACPI	—	advanced configuration and power interface
BIOS	—	basic input/ouptut system
EPT	—	extended page table
EPTE	—	технология EPT с технологией PTE
GPU	—	graphic processing unit
MMU	—	memory management unit
PCI Express	—	peripheral component interconnect express
PTE	—	page table entry
RFC	—	request for comment
RVI	—	rapid virtualization indexing
SPICE	—	simple protocol for independent computing environment
UEFI	—	unified extensible firmware interface
АРМ	—	автоматизированное рабочее место
АСЗИ	—	автоматизированная система в защищённом исполнении
вАРМ	—	виртуальное автоматизированное рабочее место
ВМ	—	виртуальная машина
ГГц	—	гигагерц
КСЗ	—	комплекс средств защиты
ПО	—	программное обеспечение
СВТ	—	средства вычислительной техники
ОС	—	операционная система
ЦП	—	центральный процессор
ЭВМ	—	электронная вычислительная машина

Лист регистрации изменений

Номера листов (страниц)					*Всего листов (страниц) в докум.*	*№ документа*	*Входящий № сопроводительного докум. и дата*	*Подп.*	*Дата*
Изм.	*измененных*	*замененных*	*новых*	*аннулированных*					