

ПЛАТФОРМА ВИРТУАЛИЗАЦИИ «V-СОФТ»

Версия 1.5.1

Руководство администратора

92386160.02001-01 90 01

Листов 196

АННОТАЦИЯ

Документ содержит сведения, необходимые для установки и настройки платформы виртуализации «V-Софт».

Документ разработан в соответствии с ГОСТ 19.103-77 «Единая система программной документации. Обозначение программ и программных документов», ГОСТ 19.104-78 «Единая система программной документации. Основные надписи», ГОСТ 19.105-78 «Единая система программной документации. Общие требования к программным документам», ГОСТ 19.106-78 «Единая система программной документации. Требования к программным документам, выполненным печатным способом».

ОГЛАВЛЕНИЕ

1. Назначение программы.....	6
2. Условия выполнения программы.....	7
2.1. Требования к аппаратному обеспечению.....	7
2.2. Требования к программному обеспечению	7
2.3. Требования к персоналу.....	7
3. Установка программы	9
3.1. Установка компонента среды виртуализации.....	9
3.1.1. Настройка Apache HTTP Server.....	10
3.2. Установка комплекса средств управления	14
3.2.1. Настройка СУБД PostgreSQL	15
3.2.2. Настройка Apache HTTP Server.....	19
3.2.3. Настройка ПО комплекса средств управления.....	23
4. Настройка программы Гипервизора и конфигурация ВМ	32
4.1. Конфигурационные файлы гипервизора.....	32
4.2. Инструмент управления гипервизором - команда xl	32
4.2.1. Описание команды xl	32
4.2.2. Подкоманды домена	33
4.2.3. Подкоманды хост-системы	42
4.2.4. Команды планировщика	45
4.2.5. Команды управления пулами ЦП	48
4.2.6. Команды управления виртуальными устройствами	49
4.3. Синтаксис конфигурационного файла домена	65
4.3.1. Описание синтаксиса	65
4.3.2. Опции конфигурационного файла	66
4.4. Конфигурация параметров сети.....	93
4.4.1. Обзор синтаксиса.....	93
4.4.2. Ключевые слова	94
4.5. Конфигурация параметров дисков.....	97
4.5.1. Позиционные параметры.....	98

4.5.2. Другие параметры и флаги	100
4.6. Планировщик реального времени RTDS	102
4.7. Примеры конфигураций виртуальных машин.....	103
4.7.1. Конфигурация VM без PCI passthrough.....	103
4.7.2. Конфигурация для PV (Linux/FreeBSD) с пробросом RAID контроллера (driver domain).....	109
4.7.3. Пример работы с Storage driver domain (PVH VM).....	111
PCI Passthrough - Primary VGA (NVidia)	112
4.7.4. Primary VGA - ATI	114
4.7.5. Secondary VGA - ATI	117
5. Настройка программы – компоненты управляющего домена	120
5.1. Настройка сети для использования с VM	120
5.1.1. Виртуальные сетевые интерфейсы	120
5.1.2. Сетевой мост.....	123
5.2. Организация прямого доступа VM к PCI/PCIe устройствам	125
5.2.1. Описание прямой передачи устройств шины ввода-вывода.....	125
5.2.2. Использование прямой передачи	127
5.2.3. Дополнительная информация и часто задаваемые вопросы.....	131
5.3. Домены ввода-вывода	138
5.3.1. Драйвер-домены	138
5.3.2. Storage-driver домены.....	140
5.3.3. USB-driver домен	140
5.3.4. Сетевой driver домен	142
5.3.5. Эмуляционный домен	142
5.3.6. Stub-домен в роли USB-домена для клавиатуры и мыши	145
5.4. Создание виртуальной машины на базе образа dom0	146
5.5. Клонирование виртуальной машины со сменой имени	156
5.6. Настройка менеджера томов LVM.....	160
5.7. Сценарии настройки в гипервизор	164
5.7.1. Настройка сетевого моста	164

5.7.2. Настройка сетевых блочных устройств	165
5.7.3. Автоматическое обновление ключей SSH	167
5.7.4. Автоматический запуск виртуальных машин.....	168
5.8. Установка ОС в Виртуальную Машину	168
5.8.1. Подготовка виртуального диска и инсталляционного носителя	168
5.8.2. Настройка VM с ОС Windows	169
5.8.3. Запуск VM и установка ОС	171
5.8.4. Установка VM с ОС Linux (Ubuntu).....	173
5.8.5. Установка паравиртуальных драйверов устройств.....	176
5.9. Примеры настройки драйвер-доменов.....	177
5.9.1. Настройка образа драйвер-домена.....	178
5.9.2. Настройка дискового драйвер-домена.....	183
6. Безопасная настройка типовых конфигураций	189
6.1. Типовая конфигурация 1 - автоматический сервер VM.....	189
6.2. Типовая конфигурация 2 - АРМ инженера с прямым доступом к оборудованию (GPU).....	189
6.3. Типовая конфигурация 3 - интерактивное АРМ разработчика	190
7. Настройка организации доступа к VM с помощью интерфейса Гипервизора.....	192
Перечень терминов.....	193
Перечень сокращений	198

1. НАЗНАЧЕНИЕ ПРОГРАММЫ

Специальное программное обеспечение платформа виртуализации «V-Софт» (далее — ПВ «V-Софт») представляет собой менеджер виртуальных машин (гипервизор) первого типа, который выполняется непосредственно на аппаратуре средств вычислительной техники (далее - СВТ).

ПВ «V-Софт» предназначен для использования в автоматизированных системах, в том числе в защищенном исполнении класса «1Б» включительно, построенных с применением технологий виртуализации и паравиртуализации.

Основными функциями ПВ «V-Софт» 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. ПВ «V-Софт» может использоваться как основа для создания информационных систем более высокого уровня.

ПВ «V-Софт» представляет собой систему низкоуровневого управления и обеспечения выполнения виртуальных машин. Для создания конечных решений на его основе необходимы соответствующие задаче операционные системы и прикладное программное обеспечение (далее - ПО). ПВ «V-Софт» рассчитан на профессиональных специалистов в сфере информационных технологий (далее - ИТ) и инженеров для использования в качестве основы при создании информационных и автоматизированных систем.

2. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ

2.1. Требования к аппаратному обеспечению

Для успешного применения ПВ «V-Софт» 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, обеспечивающем максимальный уровень изоляции и производительности, процессор должен поддерживать расширенный набор инструкций для аппаратного ускорения механизмов виртуализации: **Intel VT** и **Intel EPT** для процессоров производства Intel Corporation или **AMD-V** и **RVI** для процессоров производства AMD.

2.2. Требования к программному обеспечению

Установка ПВ «V-Софт» производится в среде ОС Astra Linux Special Edition «Смоленск» версии 1.7 или выше.

Для работы ОС и прикладного ПО в среде виртуализации, создаваемой ПВ «V-Софт», необходимо:

- 1) обеспечить работу гипервизора и управляющего домена;
- 2) в виртуальных машинах использовать ОС, поддерживающие эмулируемое оборудование — как минимум, чипсеты Intel i440 и Q35;
- 3) для обеспечения максимальной производительности целевой ОС 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

ПВ «V-Софт» является сложным техническим решением, для полноценного использования всех возможностей которого необходимо владение принципами работы вычислительной техники и сетевого оборудования, наличие навыков администрирования ОС семейства UNIX/Linux, знание стандартов **RFC** и

понимание принципов работы сетевых протоколов, используемых в корпоративных информационных системах.

3. УСТАНОВКА ПРОГРАММЫ

Дистрибутив включает два основных компонента:

- 1) дистрибутив компонента «Среда виртуализации» (далее - СВ);
- 2) дистрибутив компонента «Средства управления» (далее - СУ).

ПВ «V-Софт» поставляется на DVD-диске либо на специально подготовленном USB Flash накопителе.

Дистрибутив программы представляет собой систему каталогов с компонентами функциональных модулей и представлена в таблице 1.

Таблица 1 — Структура дистрибутива программы

№ п/п	Каталог исходного кода	Описание
1.	host/	Исходный код СВ
1.1.	host/xen/	Гипервизор Xen
1.2	host/ovs/	Виртуальный коммутатор OpenvSwitch
01.03.22	host/host_rootfs_overlay.tar.gz	Средства автоматизации и конфигурация хоста
1.4.	host/websockify/	Web-socket сервер Websockify
2.	ksu/	Исходный код СУ
2.1.	ksu/dependenceis/	Зависимости сборки СУ
2.2.	ksu/hv-manager.tar.gz	Приложение СУ

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

3.1. Установка компонента среды виртуализации

Для установки СВ на виртуальную машину необходимо убедиться в среде виртуализации установлены параметры виртуальной машины допускающие вложенную виртуализацию:

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

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

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

3.1.1. Настройка Apache HTTP Server

Для выполнения команд СУ необходимо предоставить веб серверу СВ необходимые привилегии:

- добавляем пользователя www-data в группу sudo:

```
echo "www-data ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers
```

- предоставляем пользователю www-data необходимые привилегии:

```
usercaps -f www-data
```

- отключаем режим Astra Mode путем изменения параметра в файле /etc/apache2/apache2.conf:

```
# AstraMode on  
AstraMode off
```

3.1.1.1. Настройка конфигурации HTTP-сервера для локальной аутентификации

Для настройки конфигурации HTTP-сервера необходимо выполнить:

- для управления серверным приложением СВ необходимо включить модули веб 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
```

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

Для настройки конфигурации HTTP-сервера необходимо выполнить:

□ 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 СВ 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-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 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
```

3.2. Установка комплекса средств управления

Для установки СУ необходимо развернуть приложение СУ под управлением HTTP-сервера Apache:

- установить пакеты необходимые для работы СУ. Установка требует права суперпользователя и производится в режиме sudo:

```
sudo ./1_install-packages.sh
```

- установить собранные зависимости приложения СУ. Установка требует права суперпользователя и производится в режиме sudo:

```
sudo ./2_install-dependencies.sh
```

- установить непосредственно приложение СУ:

```
./3_install-ksu.sh
```

□ необходимо добавить пользователей системы / домена.

Потребуется пользователь который будет исполнять функции администратора (для примера пользователь с именем «adm»). Отдельный пользователь для планировщика (для примера пользователь с именем «sched»). А также остальные пользователи системы в зависимости от потребностей.

Пользователи добавляются средствами ОС либо контроллера домена. Им необходимо назначить права МРД и добавить в группы `ssl-cert`, `postgres`.

Пользователю планировщика необходимо выдать права суперпользователя путем добавления в группу `sudoers`.

3.2.1. Настройка СУБД PostgreSQL

Для настройки СУБД 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 -c 'create user "adm" with createrole;'
```

3.2.1.1. Настройки СУБД для локальной аутентификации пользователей

Для настройки СУБД необходимо выполнить:

- в случае, если для аутентификации в СУ будут использоваться аутентификационные данные локальных пользователей ОС сервера СУ, рекомендуется ограничить доступ для внешних пользователей путем комментирования или удаления строк в файле «/etc/postgresql/11/main/pg_hba.conf» следующим образом:

удалить или комментировать:

```
#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 СУ и СУБД должны устанавливаться на одну машину.

□ необходимо выдать пользователю postgres разрешения подсистемы parsec:

```
sudo usermod -a -G shadow postgres
sudo setfacl -d -m u:postgres:r /etc/parsec/macdb
sudo setfacl -R -m u:postgres:r /etc/parsec/macdb
sudo setfacl -m u:postgres:rx /etc/parsec/macdb
sudo setfacl -d -m u:postgres:r /etc/parsec/capdb
sudo setfacl -R -m u:postgres:r /etc/parsec/capdb
```

```
sudo setfacl -m u:postgres:rx /etc/parsec/capdb
```

□ 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
```

3.2.1.2. Настройки СУБД для аутентификации по протоколу Kerberos

Для настройки СУБД необходимо:

□ пройти аутентификацию учетной записью администратора домена:

```
sudo kinit admin
```

□ создать принципала службы СУБД (на примере имени сервера «`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:

```
id postgres
```

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

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

- перезапустить службу sssd:

```
sudo systemctl restart sssd
```

□ необходимо изменить параметры СУБД, добавить в файл /etc/postgresql/11/main/postgresql.conf следующие параметры:

```
ac_ignore_socket_maclabel = false  
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'
```

□ а также 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
```

3.2.2. Настройка Apache HTTP Server

3.2.2.1. Настройка конфигурации HTTP-сервера для локальной аутентификации

Для настройки конфигурации HTTP-сервера необходимо:

□ для управления серверным приложением СУ необходимо включить модуль авторизации следующим образом:

```
sudo a2enmod authnz_pam
```

- отключить стандартный сайт следующей командой:

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

- для управления приложением СУ 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-available/hv-manager.conf следующее содержимое:

```
<VirtualHost *:80>

    ServerAdmin webmaster@localhost

    DocumentRoot /opt/hv-manager

    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/
```

□ после завершения настройки необходимо перезапустить сервис http-сервера следующей командой:

```
sudo systemctl restart apache2
```

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

Для настройки конфигурации HTTP-сервера необходимо выполнить:

□ 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 СУ необходимо включить модуль авторизации следующим образом:

```
sudo a2enmod auth_kerb
```

□ отключить стандартный сайт следующей командой:

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

□ для управления приложением СУ 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-available/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
```

3.2.3. Настройка ПО комплекса средств управления

Для настройки ПО необходимо:

- если используется подсистема МРД 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  
chmod 666 /opt/hv_manager/error*
```

- если подсистема МРД не используется достаточно указания прав на файл логирования:

```
touch /opt/hv_manager/error.log  
chmod 666 /opt/hv_manager/error.log
```

- средствами ОС либо контроллера домена добавить пользователя планировщика СУ и пользователей операторов.
- назначить права на файлы для всех пользователей (повторять после каждого обновления СУ):

```
sudo find /opt/hv-manager/ -type d -exec chmod 777 {} \;
```

3.2.3.1. Настройка конфигурации для локальной аутентификации

Для настройки конфигурации необходимо выполнить:

- 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
WEBSOCKETIFY_RUNNER=/opt/websocketify/run
HOST_BACKUPS_DIR=/home/backup
```

`hv-manager.domain.loc` необходимо заменить на имя сервера. Параметр `MAC_ENABLE` выставляется в `True` или `False` в зависимости от того, планируется ли использование подсистемы МРД.

- для обновления информационной модели СУ необходимо выполнить миграции:

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

```
python3 manage.py collectstatic
```

□ выдать администратору права на созданные объекты базы данных:

```
sudo -u postgres psql -c 'grant pg_read_server_files to "adm";'  
sudo -u postgres psql -c 'grant infrastructure_administrator to "adm";'  
sudo -u postgres psql -c 'grant ksu_administrator to "adm";'
```

□ для работы планировщика без механизма МРД необходимо создать файл /etc/systemd/system/ksu.service и добавить в него следующее содержимое:

```
[Unit]  
Description=KSU scheduler  
After=network.target postgresql.service  
  
[Service]  
Type=simple  
User=sched  
WorkingDirectory=/opt/hv-manager/backend/  
ExecStart=python3 manage.py runapscheduler  
ExecStop=pkill -f runapscheduler  
  
[Install]  
WantedBy=multi-user.target
```

□ 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=sched
PDPLabel=N:0:0
WorkingDirectory=/opt/hv-manager/backend/
ExecStart=python3 manage.py runapscheduler
ExecStop=pkill -f runapscheduler

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

Заменить N в параметрах Description и PDPLabel на соответствующий уровень конфиденциальности. В параметре "User" нужно указать имя пользователя планировщика ранее добавленного в ОС и СУ как обычного пользователя.

□ после выполнения настроек необходимо обновить службы:

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

3.2.3.2. Настройка конфигурации для доменной аутентификации

Для настройки конфигурации необходимо выполнить:

□ 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
WEBSOCKETIFY_RUNNER=/opt/websocketify/run
HOST_BACKUPS_DIR=/home/backup
```

`hv-manager.domain.loc` необходимо заменить на имя сервера. Параметр `MAC_ENABLE` выставляется в `True` или `False` в зависимости от того, планируется ли использование подсистемы МРД.

□ для настройки авторизации необходимо пройти аутентификацию учётной записью администратора домена:

```
sudo kinit admin
```

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

```
ipa-getkeytab -s freeipa.domain.loc -k /opt/hv-manager/sched.keytab -p
sched@DOMAIN.LOC

chown sched:sched /opt/hv-manager/sched.keytab

chmod 600 /opt/hv-manager/sched.keytab
```

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

□ для обновления информационной модели СУ необходимо выполнить миграции:

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

- выдать администратору права на созданные объекты базы данных:

```
sudo -u postgres psql -c 'grant pg_read_server_files to "adm";'  
  
sudo -u postgres psql -c 'grant infrastructure_administrator to "adm";'  
  
sudo -u postgres psql -c 'grant ksu_administrator to "adm";'
```

□ для работы планировщика без механизма МРД необходимо создать файл /etc/systemd/system/ksu.service и добавить в него следующее содержимое:

```
[Unit]  
Description=KSU scheduler  
After=network.target sssd.service postgresql.service  
  
[Service]  
Type=simple  
User=sched  
WorkingDirectory=/opt/hv-manager/backend/  
RuntimeMaxSec=12h  
Restart=on-abnormal  
ExecStartPre=/usr/bin/kinit -k -t /opt/hv-manager/sched.keytab sched@DOMAIN.LOC
```

```
ExecStart=python3 manage.py runapscheduler
```

```
ExecStop=pkill -f runapscheduler
```

```
[Install]
```

```
WantedBy=multi-user.target
```

□ 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=s c hed
```

```
PDPLabel=N:0:0
```

```
WorkingDirectory=/opt/hv-manager/backend/
```

```
RuntimeMaxSec=12h
```

```
Restart=on-abnormal
```

```
ExecStartPre=/usr/bin/kinit -k -t /opt/hv-manager/s c hed.keytab s c hed@DOMAIN.LOC
```

```
ExecStart=python3 manage.py runapscheduler
```

```
ExecStop=pkill -f runapscheduler
```

```
[Install]
```

```
WantedBy=multi-user.target
```

Заменить N в параметрах Description и PDPLabel на соответствующий уровень конфиденциальности. В параметре "User" нужно указать имя пользователя планировщика ранее добавленного в домен и СУ как обычного пользователя.

□ после выполнения настроек необходимо обновить службы:

```
sudo systemctl daemon-reload
```

```
sudo systemctl enable ksu
```

```
sudo systemctl restart ksu
```

4. НАСТРОЙКА ПРОГРАММЫ ГИПЕРВИЗОРА И КОНФИГУРАЦИЯ VM

4.1. Конфигурационные файлы гипервизора

Большинство файлов конфигураций ПВ «V-Софт», 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, для задачи создания виртуальных устройств.

В каталоге «/etc/init.d» хранятся сценарии для запуска и остановки сервисов, используемых в работе ПВ «V-Софт».

Сценарий «/etc/init.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 гипервизора, сместив ядро **Dom0** к модульной линии. В этом файле переопределяются параметры загрузки и гипервизора, а также самой гостевой ОС. К примеру, если требуется уточнить распределение фиксированной памяти для **Dom0** с помощью опции гипервизора «dom0_mem» или увеличить количество сетевых петлевых устройств с помощью опции ядра управляющего домена «nloopbacks».

В пункте 101 приведены примеры конфигурационных файлов для различных виртуальных машин.

4.2. Инструмент управления гипервизором - команда xl

4.2.1. Описание команды xl

Существует инструмент управления гипервизором, основанный на библиотеке **LibXenlight**, которому соответствует команда xl [опции].

xl управляет гостевыми доменами: создает, приостанавливает и выключает их. 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:

`xl` подкоманда [опции] `domain-id`

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

При работе с `xl` следует помнить следующее:

-  большинство операций `xl` основаны на работе сервисов **xenstored** и **xenconsole**. Соответственно, во время загрузки **Dom0** обязательно должен быть запущен скрипт «`/etc/init.d/xencommons`», ответственный за их инициализацию;

-  в большинстве сетевых конфигураций, для создания рабочей сети, необходимо настроить мост с именем «`xenbr0`» в **Dom0**;

-  если при загрузке гипервизора объем памяти **Dom0** указан посредством параметра «`dom0_mem`», следует отключить (приравнять к нулю) параметр «`autoballoon`» в файле «`/etc/xen/xl.conf`»;

-  большинство подкоманд `xl` требуют запуск с правами администратора.

4.2.1.1. Общие опции `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).

4.2.2. Подкоманды домена

С помощью подкоманд домена (см. таблицы 2 - 13) можно управлять доменами напрямую. В качестве первого параметра большинство подкоманд используют «`domain-id`». К примеру, команда `button-press domain-id button` обозначает нажатие кнопки интерфейса ACPI для домена («Включен» или «Спящий режим»).

Таблица 2 — Подкоманда create

Категория	Описание
Синтаксис	<code>create [configfile] [опции]</code>
Назначение	Создает гостевой домен
Возвращение управления команде <code>xl</code>	Как только запущен сам домен. Это не означает, что гостевая ОС в этом домене загружена или доступна для ввода
Аргументы	<i>configfile</i> — абсолютный путь к конфигурационному файлу домена (подробная информация о таких файлах содержится в файле <code>xl.cfg</code>). Если этот аргумент не задан или файл отсутствует, <code>xl</code> создаст домен, используя значения по умолчанию для каждой опции
Опции	<code>-q --quiet</code> — запретить консольный вывод
	<code>-f=FILE, --defconfig=FILE</code> — использовать указанный файл (FILE) конфигурации
	<code>-F</code> — запустить домен после его создания
	<code>-p</code> — приостановить домен после его создания
	<code>-V, --vncviewer</code> — подключиться к VNC-серверу домена, используя <code>vncviewer</code> , запущенный в отдельном процессе
	<code>-A, --vncviewer-autopass</code> — передать пароль для <code>vncviewer</code> с помощью стандартного ввода (<code>stdin</code>)
	<code>-c</code> — 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
	<i>key=value</i> — указать опции в формате опций конфигурационного файла. Такие пары <i>ключ=значение</i> 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

В качестве примера рассмотрим команду:

«`xl create DebianLenny`».

Она создает домен с файлом `/etc/xen/DebianLenny` и возвращает управление, как только он запустится.

Следующая команда создаст домен с файлом конфигурации **`hvm.cfg`**, свяжет его с ЦП (CPU) «0-3» и предоставит прямой доступ к двум PCI-устройствам:

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

Далее будут рассмотрены подкоманды: `config-update`, `console`, `destroy`, `list`, `migrate`, `remus`, `reboot`, `restore`, `save`, `shutdown`, `vncviewer`.

Таблица 3 — Подкоманда config-update

Категория	Описание
Синтаксис	config-update domain-id [configfile] [опции]
Назначение	Обновляет сохранённую конфигурацию работающего домена (при перезапуске гостевой ОС). Эта команда полезна для того, чтобы гарантировать, что изменения, внесенные в гостевую систему, сохранятся при ее перезапуске
Аргументы	<i>configfile</i> — абсолютный путь к конфигурационному файлу домена (подробная информация о таких файлах содержится в файле xl.cfg)
Опции	<i>-f=FILE</i> , <i>--defconfig=FILE</i> — использовать указанный файл (FILE) конфигурации
	<i>key=value</i> — указать опции в формате опций конфигурационного файла. Такие пары <i>ключ=значение</i> перекрывают аналогичные, заданные в конфигурационном файле

Таблица 4 — Подкоманда console

Категория	Описание
Синтаксис	console [опции] domain-id
Назначение	Подсоединяет консоль domain-id к домену
Опции	<i>-t [pv serial]</i> — 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
	<i>-n NUM</i> — подключить к консоли с номером NUM. Номера консолей начинаются с «0»

Таблица 5 — Подкоманда destroy

Категория	Описание
Синтаксис	destroy [опции] domain-id
Назначение	Немедленно завершает работу домена, ОС домена не успевает среагировать. В большинстве случаев вместо destroy рекомендуется использовать подкоманду выключения shutdown
Опции	<i>-f</i> — разрешить уничтожение Dom0. Поскольку домен не может уничтожить сам себя, выполнение опции возможно только с использованием разделенной структуры и наиболее полезно тогда, когда аппаратный домен отделен от Dom0

Таблица 6 — Подкоманда list

Категория	Описание
Синтаксис	list [опции] [domain-id ...]
Назначение	Выводит список с информацией об одном или нескольких доменах. Если имена не указаны, выводит информацию обо всех доменах
Опции	<i>-l</i> , <i>--long</i> — вывести список не в виде таблицы, а в виде структуры данных JSON
	<i>-Z</i> , <i>--context</i> — вывести дополнительно метки защиты

Таблица 6 — Подкоманда list

Категория	Описание
	-v, -- <i>verbose</i> — вывести дополнительно UUID домена, причины выключения и метки защиты
	-c, -- <i>cripool</i> — вывести дополнительно cripool домена
	-n, -- <i>numa</i> — вывести информацию принадлежности домена хосту NUMA

Пример. Формат вывода списка:

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

В списке выводятся следующие данные:

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

В колонке **State** отображается состояние 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 гипервизора;

 *s* — *shutdown*, гостевая ОС выключена (был произведен вызов `SCHEDOP_shutdown`), но домен еще 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.

Таблица 7 — Подкоманда *migrate*

Категория	Описание
Синтаксис	<code>migrate [опции] domain-id host</code>
Назначение	Переносит домен на другую хост-машину (по умолчанию, с помощью SSH)
Опции	<code>-s sshcommand</code> — использовать <code>sshcommand</code> вместо SSH
	<code>-e</code> — не ждать в фоновом режиме полного выключения домена на новом хосте
	<code>-debug</code> — вывести информацию об отладке в процессе миграции
	<code>-c</code> — 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
	<code>-c, --crrpool</code> — вывести дополнительно <code>crrpool</code> домена
	<code>-p</code> — оставить на принимающей стороне домен на паузе после миграции

Таблица 8 — Подкоманда *remus*

Категория	Описание
Синтаксис	<code>remus [опции] domain-id host</code>
Назначение	Задействует технологию Remus HA или COLO HA на домене (по умолчанию, с помощью SSH)
Опции	<code>-i MS</code> — создавать контрольные точки памяти домена каждую миллисекунду (по умолчанию 200 мс)
	<code>-u</code> — отключить сжатие контрольных точек памяти
	<code>-s sshcommand</code> — использовать <code>sshcommand</code> вместо SSH
	<code>-e</code> — не ждать в фоновом режиме полного выключения домена на новом хосте
	<code>-N netbufscript</code> — использовать <code>netbufscript</code> для настройки сети буферизации вместо сценария по умолчанию (<code>/etc/xen/scripts/remus-netbuf-setup</code>)
	<code>-F</code> — запустить Remus в небезопасном режиме
	<code>-b</code> — продублировать контрольные точки памяти в <code>/dev/null</code> (черная дыра). Полезная опция для отладки. Требуется включения небезопасного режима
	<code>-n</code> — отключить сетевую буферизацию вывода. Требуется включения небезопасного режима

Категория	Описание
Синтаксис	remus [опции] domain-id host
	<i>-d</i> — отключить репликацию диска. Требуется включения небезопасного режима
Опции	<i>-c</i> — запустить COLO HA. Данная опция конфликтует с « <i>-i</i> » и « <i>-b</i> ». сжатие контрольных точек памяти должно быть отключено
	<i>-p</i> — использовать пользовательское пространство COLO Proxy. Эта опция должна использоваться совместно с « <i>-c</i> »

Таблица 9 — Подкоманда reboot

Категория	Описание
Синтаксис	reboot [опции] domain-id
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 « <i>-F</i> ». Процессы, сопутствующие перезапуску, настраиваются с помощью параметра «on_reboot» в конфигурационном файле домена
Возвращение управления команде xl	Подкоманда возвращает управление в момент перезагрузки (до фактической загрузки домена)
Опции	<i>-F</i> — включает откат к отправке события включения ACPI в случае, если гостевая ОС не поддерживает управление перезагрузкой с помощью PV. Требуется соответствующей настройки ОС

Таблица 10 — Подкоманда restore

Категория	Описание
Синтаксис	restore [опции] [ConfigFile] CheckpointFile
Назначение	Создает домен из файла состояния, созданного с помощью подкоманды «save»
Опции	<i>-p</i> — не активировать домен после его восстановления
	<i>-e</i> — не ждать в фоновом режиме полного выключения домена на новом хосте
	<i>-d</i> — включить отладочные сообщения
	<i>-V, --vncviewer</i> — подключиться к VNC-серверу домена, используя vncviewer, запущенный в отдельном процессе
	<i>-A, --vncviewer-autopass</i> — передать пароль для vncviewer с помощью стандартного ввода «stdin»

Таблица 11 — Подкоманда save

Категория	Описание
Синтаксис	save [опции] domain-id CheckpointFile [configfile]
Назначение	Сохраняет работающий домен для последующего восстановления. По умолчанию, домен не будет использоваться. Через параметр «configfile» передается конфигурационный файл домена

Категория	Описание
Синтаксис	save [опции] domain-id CheckpointFile [configfile]
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

Таблица 12 — Подкоманда shutdown

Категория	Описание
Синтаксис	shutdown [опции] -a domain-id
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
Возвращение управления команде xl	Подкоманда возвращает управление немедленно
Опции	-a, --all — выключить все гостевые домены. Часто используется при полном выключении всей системы
	-w, --wait — ожидать выключения домена перед возвратом из команды
	-F — включает откат к отправке события включения ACPI в случае, если гостевая ОС не поддерживает управление перезагрузкой с помощью PV. Требуется соответствующая настройка ОС

Таблица 13 — Подкоманда vncviewer

Категория	Описание
Синтаксис	vncviewer [опции] domain-id
Назначение	Подключает к VNC-серверу домена, запущенному в отдельном процессе
Опции	--autopass — передать пароль для vncviewer с помощью стандартного ввода

4.2.2.1. Дополнительные подкоманды домена

Дополнительные подкоманды домена:

 mem-max domain-id mem — указать максимальный объем памяти, который может использовать домен. Приставки: <t> для терабайт, <g> для гигабайт, <m> для мегабайт, <k> для килобайт и для байт. Значение **mem-max** может не соответствовать реальной памяти, используемой доменом, поскольку часть места используется ОС;

Окончание таблицы 10
main-id mem — настроить используемую память домена с помощью balloon-драйвера. 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, распространенные 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, определенное 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 гипервизора;

 `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 «жесткую» и «мягкую» привязку для виртуального ЦП vCPU домена с идентификатором «domain-id». Обычно виртуальные ЦП перемещаются между доступными физическими ЦП в соответствии с командами гипервизора. «Жесткая» привязка ограничивает этот процесс, чтобы убедиться, что конкретные виртуальные процессоры могут работать только на определенных физических процессорах. «Мягкая» привязка задает 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 «жесткую», так и «мягкую» привязки для всех виртуальных ЦП в домене. Символ «-» можно использовать, чтобы оставить только «жесткую» или только «мягкую» привязку. Например, команда `x1 vcpu-pin 0 3 — 6-9` установит «мягкую» привязку для виртуального ЦП «3» домена **Dom0** к физическими ЦП «6», «7», «8» и «9», оставляя «жесткую» привязку незатронутой. Команда `x1 vcpu-pin 0 3 3,4 6-9` установит параметры как «жесткой», так и «мягкой» привязки, первого к физическим ЦП «3» и «4», второго — 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.

Пример. Формат для списка:

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

4.2.3. Подкоманды хост-системы

Подкоманды хост-системы представлены в таблицах 14 - 19.

Таблица 14 — Подкоманда `debug-keys`

Категория	Описание
Синтаксис	<code>debug-keys keys</code>
Назначение	Отправляет отладочную последовательность нажатия клавиш в гипервизоре. Имеет тот же эффект, что тройное нажатие «conswitch» (по умолчанию, «Ctrl+A») и затем последовательное нажатие клавиш «keys»

Таблица 15 — Подкоманда `dmesg`

Категория	Описание
Синтаксис	<code>dmesg [-c]</code>
Назначение	Читает буфер сообщений гипервизора, аналогично «dmesg» в ОС Linux. Буфер содержит информационные сообщения, предупреждения и сообщения об ошибках, созданные во время процесса загрузки гипервизора
Опции	<code>-c</code> , <code>--clear</code> — очищает буфер сообщений гипервизора

Таблица 16 — Подкоманда `info`

Категория	Описание
Синтаксис	<code>info [-n, --numa]</code>
Назначение	Выводит информацию о хосте гипервизора в формате «имя : значение»
Опции	<code>-n</code> , <code>--numa</code> — Выводит информацию о топологии хоста «NUMA»

Пример. Вывод информации о хосте гипервизора:

```
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

Значения некоторых полей команды info:

 *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 гипервизор 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 определенное 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 гипервизоре, 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 гипервизора.

Таблица 17 — Подкоманда **top**

Категория	Описание
Синтаксис	top
Назначение	Выполняет команду « xentop », которая обеспечивает мониторинг доменов в реальном времени

Таблица 18 — Подкоманда **uptime**

Категория	Описание
Синтаксис	uptime
Назначение	Выводит текущее время бесперебойной работы запущенных доменов

Таблица 19 — Подкоманда **claims**

Категория	Описание
Синтаксис	claims
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 » и значения « outstanding_claims ». Колонка « Mem » дает совокупное значение запрошенной памяти и общего объема памяти, которое было в данный момент выделено для гостей

Пример. Вывод информации о дополнительной памяти гостей:

```
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).

4.2.4. Команды планировщика

Гипервизор поставляется с набором планировщиков домена (см. таблицы 20 - 23), который может быть установлен во время загрузки с параметром «*sched=*» в командной строке гипервизора. По умолчанию, кредитный лимит (*credit*) используется для планирования.

Таблица 20 — Команда *sched-credit*

Категория	Описание
Синтаксис	<i>sched-credit</i> [опции]
Назначение	Устанавливает или получает параметры кредитного планировщика. Кредитный планировщик соответствует планировщику ЦП, выделяющему ресурсы равными долями, работающему на симметричных многопроцессорных системах. Каждому домену (включая Dom0) присваиваются вес и верхний лимит
Опции	<i>-d DOMAIN, --domain=DOMAIN</i> — указывает домен, для которого необходимо изменить или восстановить параметры планировщика. Опция обязательна для изменения параметров планировщика
	<i>-w WEIGHT, --weight=WEIGHT</i> — 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»
	<i>-c CAP, --cap=CAP</i> — 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 один физический ЦП, «50» — это половина ЦП, «400» — это четыре 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 « <i>cap</i> » 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)
	<i>-p CPUPOOL, --cpupool=CPUPOOL</i> — ограничить вывод к доменам в указанном « <i>cpupool</i> »
	<i>-s, --schedparam</i> — добавить в список или настроить параметры планировщика, относящиеся ко всему пулу
	<i>-t TSLICE, --tslice_ms=TSLICE</i> — 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»
Опции	<i>-r RLIMIT, --ratelimit_us=RLIMIT</i> — параметр «RLIMIT» пытается ограничить

Категория	Описание
Синтаксис	sched-credit [опции]
	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 мкс до 500 000 мкс. Величина «RLIMIT» должна быть меньше, чем продолжительность кванта времени «TSLICE»

Комбинации сочетаний вышеприведенных 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.

Таблица 21 — Команда sched-credit2

Категория	Описание
Синтаксис	sched-credit2 [опции]
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) присваивается вес
Опции	<code>-d DOMAIN, --domain=DOMAIN</code> — указывает домен, для которого необходимо изменить или восстановить параметры планировщика. Опция обязательна для изменения параметров планировщика
	<code>-w WEIGHT, --weight=WEIGHT</code> — 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»
Опции	<code>-p CPUPOOL, --cpupool=CPUPOOL</code> — ограничить вывод к доменам в указанном «сриpool»

Таблица 22 — Команда sched-sedf

Категория	Описание
Синтаксис	sched-sedf [опции]
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
Опции	-d DOMAIN, --domain=DOMAIN — указывает 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
Опции	-c CPUPOOL, --cpu-pool=CPUPOOL — ограничить вывод к доменам в указанном «cpu-pool»

Таблица 23 — Команда sched-rtds

Категория	Описание
Синтаксис	sched-rtds [опции]
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 «тратит» свой бюджет. Бюджет vCPU пополняется в начале каждого периода; неиспользованный бюджет «списывается» в конце каждого периода
Опции	-d DOMAIN, --domain=DOMAIN — указывает 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
Опции	-c CPUPOOL, --cpu-pool=CPUPOOL — ограничить вывод к доменам в указанном «cpu-pool»

4.2.5. Команды управления пулами ЦП

Гипервизор может группировать физические процессоры сервера в группы (или пулы) ЦП, для предоставления возможности использовать различные планировщики (таблица 24) 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 в данном пункте командой. Пулы ЦП идентифицируются либо по имени, либо по ID.

Таблица 24 — Команда `crupool-create`

Категория	Описание
Синтаксис	<code>crupool-create [опции] [configfile] [Variable=Value ...]</code>
Назначение	Создает пул ЦП на базе конфигурации из « <code>configfile</code> » или параметров командной строки. Значения переменных из « <code>configfile</code> » могут быть изменены посредством установки новых или дополнительных назначений в командной строке
Опции	<code>-f=FILE, --defconfig=FILE</code> — использовать данный файл конфигурации
	<code>crupool-list [-c --cpus] [cpu-pool]</code> — вывести список пулов ЦП на хосте. Если « <code>-c</code> » указано, <code>x1</code> выведет список процессоров, используемых пулом ЦП
	<code>crupool-rename cpu-pool <newname></code> — переименовать пул ЦП в « <code>newname</code> »
	<code>crupool-cpu-add cpu-pool cpu-nr node:node-nr</code> — добавить ЦП или все процессоры узла « <code>NUMA</code> » к пулу ЦП
	<code>crupool-migrate domain cpu-pool</code> — переместить домен, обозначенный с помощью « <code>domain-id</code> » или « <code>domain-name</code> », в пул ЦП
	<code>crupool-numa-split</code> — разделить машину таким образом, что на каждый узел « <code>NUMA</code> » приходился один пул ЦП

4.2.6. 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».

4.2.6.1. Работа с блочными устройствами

Команды для работы с блочными устройствами описаны в таблицах 25 - 29.

Таблица 25 — Команда `block-attach`

Категория	Описание
Синтаксис	<code>block-attach domain-id disc-spec-component(s) ...</code>
Назначение	Создает новое виртуальное блочное устройство
Опции	<i>domain-id</i> — идентификатор домена гостевого домена, к которому будет подключено устройство
	<i>disc-spec-component</i> — спецификация диска в формате, аналогичном используемому для параметра «disk» в конфигурационном файле домена

Таблица 26 — Команда `block-list`

Категория	Описание
Синтаксис	<code>block-list domain-id</code>
Назначение	Выводит список виртуальных блочных устройств для домена

Таблица 27 — Команда `block-detach`

Категория	Описание
Синтаксис	<code>block-detach domain-id devid [--force]</code>
Назначение	Отсоединяет виртуальное блочное устройство домена. « <i>devid</i> » — имя или числовой идентификатор устройства, присвоенный устройству доменом Dom0. Чтобы определить число, требуется запустить « <code>xl block-list</code> ». 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 « <code>--force</code> » отсоединит устройство принудительно, но, возможно, вызовет ошибки ввода-вывода в домене

Таблица 28 — Команда `cd-insert`

Категория	Описание
Синтаксис	<code>cd-insert domain-id VirtualDevice target</code>
Назначение	Вставляет CDrom в существующий виртуальный CD-привод гостевого домена. Виртуальный привод может быть пустым, но должен существовать. Работает только с HVM-доменами
Опции	<i>VirtualDevice</i> — указать, каким образом устройство должно быть показано гостевому домену (например, « <code>hdc</code> »)
	<i>target</i> — путь назначения в бэкэнд домене (обычно Dom0), который следует экспортировать (блочное устройство, файл и т. д.). Пример доступен в файле <code>docs/misc/xl-disk-configuration.txt</code>

Таблица 29 — Команда `cd-eject`

Категория	Описание
Синтаксис	<code>cd-eject domain-id VirtualDevice</code>
Назначение	Извлекает CD-rom из виртуального привода гостя. Работает только с HVM-доменами
Опции	<i>VirtualDevice</i> — указать, каким образом устройство должно быть показано гостевому домену (например, « <code>hdc</code> »)

4.2.6.2. Работа с сетевыми устройствами

Команды для работы с сетевыми устройствами описаны в таблицах 30 - 32.

Таблица 30 — Команда network-attach

Категория	Описание
Синтаксис	network-attach domain-id network-device
Назначение	Создает новое сетевое устройство для домена «domain-id». «network-device» задает устройство, которое следует присоединить (используя тот же формат, что и «vif» в файле конфигурации домена)

Таблица 31 — Команда network-detach

Категория	Описание
Синтаксис	network-detach domain-id devid mac
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-адрес

Таблица 32 — Команда network-list

Категория	Описание
Синтаксис	network-list domain-id
Назначение	Выводит список виртуальных сетевых интерфейсов для домена

4.2.6.3. Работа с канальными устройствами

Команда для работы с канальными устройствами представлена в таблице 33.

Таблица 33 — Команда channel-list

Категория	Описание
Синтаксис	channel-list domain-id
Назначение	Выводит список виртуальных канальных интерфейсов для домена

4.2.6.3.1. Прямой доступ к PCI устройствам

Список команд управления прямым доступом к PCI устройствам:

1) 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**;

2) pci-assignable-add BDF — подготовить устройство для прямого доступа из ВМ, определяемое PCI Bus/Device/Function (BDF). Эта команда свяжет устройство с драйвером pciback;

3) если устройство уже связано с каким либо драйвером, исходный драйвер будет откреплён от него и сохранён таким образом, чтобы возможно было привязать устройство к этому драйверу повторно. Если устройство подготовлено, `xl` возвращает код «success»;

4) 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;

5) `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 заново привязать устройство к его исходному драйверу, делая его вновь пригодным для использования доменом **Dom0**. Если устройство не привязано к драйверу `pciback`, `xl` возвращает код «success»;

6) `pci-attach domain-id BDF` — «горячее» подключение нового PCI-устройства для прямого доступа в определенном домене (BM). BDF — идентификатор PCI устройства, ранее подготовленного для прямого доступа;

7) `pci-detach [-f] domain-id BDF` — «горячее» отключение ранее подключенного PCI-устройства от домена, определяемого BDF. Если указана опция «-f», `xl` принудительно удалит устройство, без взаимодействия с гостем;

8) `pci-list domain-id` — вывести список PCI-устройств для прямой передачи для домена.

9) `pci-device-reservations` - реестр идентификаторов устройств **Xen** PCI.

PCI-вендор ID 0x5853 зарезервирован системами **Xen** для объявления определенного виртуального оборудования гостевым виртуальным машинам. В основном он используется с идентификатором устройства 0x0001 для объявления устройства PCI-платформы **Xen** - наличие этого виртуального устройства позволяет гостевой операционной системе (при условии наличия подходящих драйверов) использовать функции паравиртуализации, такие как дисковые и сетевые устройства и т. д.

Некоторые вендоры **Xen** предпочитают предоставлять альтернативные и/или дополнительные гостевые драйверы, которые могут быть привязаны к виртуальным устройствам [1]. Это может быть осуществлено с помощью ID вендора **Xen** PCI 0x5853 и ID конкретного PCI-устройства Xen-вендора/устройства. Данный файл записывает резервирования, сделанные в пределах диапазона ID устройства, чтобы избежать использования несколькими Xen-вендорами конфликтующих идентификаторов.

Вендор может запросить ряд ID устройств, предложив патч к этому файлу.

Распределение вендора должно быть в диапазоне «0xc000-0xffff» для уменьшения вероятности конфликтов с ID сообщества, назначенными снизу вверх. Вендор отвечает за распределения в пределах диапазона и должен попытаться записать определённые ID устройств в базы данных PCI ID, такие как «<https://pci-ids.ucw.cz>» и «<https://devicehunt.com>».

4.2.6.3.2. Резервирование

Диапазоны резервирования представлены в таблице 34.

Таблица 34 — Резервирование

Диапазон	Вендор/Продукт
0x0001	(устройство Xen Platform PCI)
0x0002	Citrix XenServer (унаследованное выделение для XenServer 6.1)
0xc000-0xc0ff	Citrix XenServer
0xc100-0xc1ff	Citrix XenClient
0xc200-0xc2ff	XCP-ng Project (https://xcp-ng.org)

Исходный **QEMU** предоставляет параметризованное устройство под названием `xen-pvdevice`, которое может использоваться для размещения гостевых драйверов.

Выполнение:

```
qemu-system-i386 -device xen-pvdevice, help
```

для списка всех параметров. Следующие параметры относятся к привязке драйвера:

```
=item vendor-id (default 0x5853)
```

ID PCI-вендора и ID вендора подсистемы устройства.

```
=item device-id (должен быть указан)
```

PCI ID устройства и ID подсистемы устройства.

```
=item revision (по умолчанию 0x01)
```

4.2.6.3.3. Версия PCI-устройства

Также параметр размера (по умолчанию «0x400000») может быть использован для указания размера одиночного MMIO BAR, который предоставляется устройством. Эта область может быть использована драйверами для отображения таблиц грантов и т. д.

Обратите внимание, что наличие PCI-устройства Xen Platform, как правило, является предварительным условием для дополнительного устройства «xen-pv», поскольку именно устройство платформы предоставляет порты ввода-вывода, необходимые для отключения эмулируемых устройств. См. «hvm-emulated-unplug.markdown» для получения подробной информации о портах ввода-вывода и протоколе отключения.

libxl обеспечивает поддержку создания одиночного дополнительного «xen-pv-device».

4.2.6.3.4. xen-pv-channel - PV-каналы Xen

Канал – это частный поток байтов с низкой пропускной способностью, схожий с линейно-последовательной передачей данных.

Типичное использование каналов:

1) для предоставления информации о начальной конфигурации виртуальной машине при загрузке (пример использования: служба ранней конфигурации **CloudStack**);

2) сигнал/запрос внутригостевого агента (пример использования: гостевой агент **oVirt**).

Каналы схожи с виртуально-серийные устройствами и эмулированными последовательными линиями передачи данных.

Каналы предназначены для использования в реализации «libvirt» при выполнении на **Xen**.

Примечание. Если приложению требуется канал с высокой пропускной способностью, то следует использовать «vchan».

4.2.6.3.4.1 .Модель использования

Рассмотрим облачное развертывание, при котором виртуальные машины клонируются из готовых шаблонов и настраиваются при первой загрузке внутригостевым агентом, который задает: IP-адрес, имя хоста, ключи ssh и т. д. Для установки системы администратор облака сначала должен:

-  установить гостя как обычно (нет необходимости в настройке канала);
-  установить внутригостевого агента, специфичного для облачного программного обеспечения. Это подготовит гостя к общению по каналу, а также подготовит его к безопасному клонированию (иногда называемому «sysprepping»);
-  выключить гостя;
-  зарегистрировать гостя в качестве шаблона с помощью программного обеспечения для оркестрации облака;
-  установить агент оркестрации облака в **dom0**.

Во время выполнения, когда облачный клиент запрашивает создание виртуальной машины из шаблона, последовательность событий будет следующей (при условии Linux **domU**):

-  виртуальная машина «клонировается» из шаблона;
-  в **dom0** выделяется уникальный путь к сокету домена Unix (например, /my/cloud/software/talk/to/domain/).

Для виртуальной машины создается конфигурация домена, в которой указывается имя канала, ожидаемое внутригостевым агентом. В синтаксисе xl это будет:

```
channel = [ «connection=socket, name=org.my.cloud.software.agent.version1, path =
/my/cloud/software/talk/to/domain/» ]
```

- 1) виртуальная машина запущена;
- 2) в **dom0** агент оркестрации облака подключается к сокету домена Unix, записывает сообщение подтверждения и ожидает ответа;
- 3) предполагая, что в гостевом ядре установлен CONFIG_HVC_XEN_FRONTEND, драйвер консоли сгенерирует событие горячего подключения;
- 4) правило **udev** активируется событием горячего подключения;

Пример. Использование правила udev:

```
SUBSYSTEM=="xen", DEVPATH=="devices/console-[0-9]", RUN+="xen-console-setup"
```

где скрипт «xen-console-setup» считывает имя канала и создает символическую ссылку в /dev/xen-channel/org.my.cloud.software.agent.version1 указывая на /dev/hvcN. «N» - это то же число, что и число в «/devices/console-[0-9]», «devices/console-2» сопоставляется с /dev/hvc2;

- 5) внутренний гостевой агент использует **inotify** для просмотра создания символической ссылки /dev/xen-channel и открывает устройство;
- 6) внутренний гостевой агент завершает подтверждение установления связи с агентом **dom0**;
- 7) агент **dom0** передает уникальную конфигурацию виртуальной машины: имя хоста, IP-адрес, ключи ssh и т. д.;
- 8) внутренний гостевой агент получает конфигурацию и применяет ее.

Использование каналов позволяет избежать использования временного дискового устройства или сетевого подключения.

4.2.6.3.5. Рекомендации по проектированию и подводные камни

Прежде чем использовать канал, необходимо установить в гостя канально-ориентированное программное обеспечение (агент). По умолчанию канал появится

как устройство, которое может быть ошибочно принято за последовательный порт или обычную консоль. Известно, что некоторые программы будут активно искать серийные порты и выдавать на них АТ-команды; необходимо убедиться, что такое программное обеспечение отключено.

Так как каналы идентифицируются по именам, авторы приложения должны убедиться, что используемые ими имена каналов уникальны, чтобы избежать конфликтов.

Рекомендуется включать в названия каналов уникальные части, относящиеся к приложениям, такие как доменные имена. Чтобы предотвратить конфликты, авторам следует добавлять свои имена в глобальный реестр каналов.

Горячее подключение и отключение каналов в настоящее время не реализовано.

4.2.6.3.6. Реестр имен каналов

Важно, чтобы имена каналов были уникальными в глобальном масштабе.

Чтобы гарантировать, что чьи-либо имена не конфликтуют с вашими, добавьте свои в этот список.

4.2.6.4. Пулы памяти tmem

Команды для работы с пулами памяти представлены в таблицах 35 - 40.

Таблица 35 — Команда tmem-list

Категория	Описание
Синтаксис	tmem-list I[<-l>] domain-id
Назначение	Выводит список пулов TMEM. Если указана опция «-l», выводит также статистику TMEM

Таблица 36 — Команда tmem-freeze

Категория	Описание
Синтаксис	tmem-freeze domain-id
Назначение	Замораживает пулы TMEM

Таблица 37 — Команда tmem-thaw

Категория	Описание
Синтаксис	tmem-thaw domain-id
Назначение	Размораживает пулы TMEM

Таблица 38 — Команда `tmem-set`

Категория	Описание
Синтаксис	<code>tmem-set domain-id [опции]</code>
Назначение	Меняет настройки TMEM
Опции	<code>-w WEIGHT</code> — вес (целое)
	<code>-c CAP</code> — верхний лимит (целое)
	<code>-p COMPRESS</code> — сжать (целое)

Таблица 39 — Команда `tmem-shared-auth`

Категория	Описание
Синтаксис	<code>tmem-shared-auth domain-id [опции]</code>
Назначение	Де/аутентифицирует поделенные пулы TMEM
Опции	<code>-u UUID</code> — определить UUID
	<code>-a AUTH</code> — <code>0=auth</code> , <code>1=deauth</code>

Таблица 40 — Команда `tmem-freeable`

Категория	Описание
Синтаксис	<code>tmem-freeable</code>
Назначение	Выводит информацию о том, сколько свободной памяти (Мбайт) использует TMEM

4.2.6.5. Команда `xenstore`

Набор утилит для взаимодействия с Xenstore включает в себя:

-  `xenstore-chmod` - настраивает разрешения для ключей;
-  `xenstore-exists` - проверка существования ключа;
-  `xenstore-list` - перечень ключей;
-  `xenstore-ls` - перечень ключей, их значений и разрешений;
-  `xenstore-read` - прочитайте значение ключа;
-  `xenstore-rm` - удалить ключи;
-  `xenstore-watch` - отслеживать изменения ключей или значений;
-  `xenstore-write` - добавить или изменить ключи или значения;
-  `xenstore-chmod` - установить разрешения для ключа **Xenstore**;
-  `xenstore-chmod [I<OPTION>]... [I<KEY>] [I<PERM>]...` - устанавливает

разрешения *Xenstore* *I<KEY>*.

I<PERM> имеет формат *I<LD>*, где *I<L>* - это буква для типа разрешения, а *I<D>* является соответствующим идентификатором домена.

Типы разрешений:

-  *r* - прочесть;
-  *w* - записать;
-  *b* - прочесть и записать;
-  *n* - нет доступа.

Первая запись разрешения - это домен, которому принадлежит ключ (владелец) *I<u>* разрешения для любого домена, явно не указанного в последующих записях. Владелец ключа всегда имеет полный доступ (чтение, запись и установка разрешений).

-  *-r* - применить права доступа к ключу и ко всем его *I<потомкам>*;
-  *-s* - подключиться к демону **Xenstore**, используя только локальный сокет;
-  *-u* - применить разрешения к ключу и всем его *I<родителям>*.

xenstore-ls - список ключей и значений **Xenstore**

B<xenstore-ls> [I<OPTION>]... [I<PATH>]...

Перечислите ключи, значения и разрешения одного или нескольких **Xenstore** *I<PATH>*, используя вложенное древовидное представление.

-  *-f* - показать полный путь для всех ключей;
-  *-p* - показать разрешения всех перечисленных ключей в виде списка через запятую.

Каждое разрешение имеет формат *I<LD>*, где *I<L>* - это буква для типа разрешения, а *I<D>* является соответствующим идентификатором домена.

Типы разрешений:

-  *r* - прочесть;
-  *w* - записать;
-  *b* - прочесть и записать;
-  *n* - нет доступа.

Первая запись разрешения - это домен, которому принадлежит ключ (владелец) *I<u>* разрешения для любого домена, явно не указанного в последующих записях. Владелец ключа всегда имеет полный доступ (чтение, запись и установка разрешений).

-s - подключитесь к демону **Xenstore**, используя только локальный сокет;
 xenstore-read - считывает значения **Xenstore**.

Считать значения одного или нескольких *Xenstore I<PATH>*.

 -p - значение префикса с именем ключа;

 -s - подключиться к демону **Xenstore** только через локальный сокет;

 -R - прочитав необработанное значение, пропустить обработку

непечатных символов (\х..).

xenstore-write - записывает значения **Xenstore**.

B<xenstore-read> [I<OPTION>]... I<PATH> I<VALUE>...

Записать *I<VALUE>* в *Xenstore I<PATH>*. Можно указать несколько пар *I<PATH> I<VALUE>* для их одновременной записи в одной транзакции **Xenstore**.

 -s Подключиться к демону **Xenstore** только через локальный сокет.

 -R Записать необработанное значение, пропустить обработку непечатных символов(\х..).

4.2.6.6. Команда xentop

Отображает в режиме реального времени информацию о **Xen** системе и доменах.

*B<xentop> [B<-h>] [B<-V>] [B<-d>SECONDS] [B<-n>] [B<-r>] [B<-v>]
 [B<-f>] [B<-f>] [B<-b>] [B<-i>ITERATIONS]*

B<Xentop> отображает информацию о системе и доменах **Xen** в постоянно обновляемой форме.

Параметры командной строки и интерактивные команды могут изменять детали и формат информации, отображаемой *B<xentop>*:

 -h, --help - отображение справки и выход;

 -V, --version - вывод информации о версии и выход;

 -d, --delay=*I<SECONDS>* - секунды между обновлениями (по умолчанию «3»);

 -n, --networks - вывод информации о сети;

 -x, --vbds - вывод данных блочного устройства «vbd»;

-  **-r, --repeat-header** - повторять заголовок таблицы перед каждым доменом;
-  **-v, --vcpus** - вывод данных виртуального ЦП;
-  **-f, --full-name** - вывести полное доменное имя (не усеченное);
-  **-b, --batch** - вывод данных в пакетном режиме (в stdout);
-  **-i, --iterations=I<ITERATIONS>** - максимальное количество итераций,

которые xentop должен произвести перед завершением.

4.2.6.6.1. Интерактивные команды

Все интерактивные команды нечувствительны к регистру:

-  **D** - установить задержку между обновлениями;
-  **N** - включить отображение сетевой информации;
-  **Q, Esc** - выйти;
-  **R** - включить заголовок таблицы перед каждым доменом;
-  **S** - порядок сортировки цикла;
-  **V** - включить отображение информации виртуального ЦП;
-  **Arrows** - прокрутить отображения домена.

4.2.6.7. Команда xentrace

xentrace [I<OPTIONS>] [I<FILE>]

Используется для захвата данных буфера трассировки из **Xen**. Данные выводятся в следующем двоичном формате (host endian):

```
CPU(uint) TSC(u64) EVENT(u32) D1 D2 D3 D4 D5 ( в с е u32)
```

где CPU - номер процессора, TSC - метка времени записи (значение счетчика цикла процессора), EVENT - это идентификатор события, D1...D5 - это данные трассировки.

Данные сбрасываются на стандартный выход (который не должен быть TTY) или I<FILE>, указанный в командной строке.

Вывод должен быть обработан с помощью инструмента xentrace_format, который может производить вывод в удобном для чтения формате ASCII.

Опции:

-  **-t *I<l>*, --log-thresh *<l>*** - устанавливает пороговое число, *I<l>*, новых записей, необходимых для инициирования записи всех новых записей на выходе;
-  **-s *I<p>*, --poll-sleep *<p>*** - устанавливает время, *I<p>*, (в миллисекундах), для ожидания между опросами буферов о новых данных;
-  **-c [*I<c>*|*I<CPU-LIST>*|*I<all>*], --cpu-mask [*I<c>*|*I<CPU-LIST>*|*I<all>*]]**.
 Это могут быть: шестнадцатеричное значение (вида 0xNNNN...) или набор диапазонов CPU, как описано ниже, или строка *I<all>*. Шестнадцатеричные значения ограничены до 32 бит. Если не указано, то будет построена маска CPU, установленная при загрузке. При использовании *I<CPU-LIST>* она ожидает десятичные числа, которые можно указать следующим образом:
 - 1) "0-3" - отслеживать только на процессорах от «0» до «3»;
 - 2) "0,2,5-7" - отслеживать только процессоры «0», «2» и «5-7»;
 - 3) "-3" - отслеживать только на процессорах от «0» до «3»;
 - 4) "-3,7" - отслеживать только на процессорах от «0» до «3» и «7»;
 - 5) "3-" - отслеживать только на процессорах от «3» до максимального количества процессоров, которые есть у хоста. При использовании *I<all>* будут применяться все процессоры, которые есть у хоста;
-  **-e *I<mask>*, -evt-mask<mask>** - устанавливает маску перехвата событий.
 Если не указано, будет использоваться TRC_ALL;
-  **-?, --help** - выводится подсказка;
-  **--usage** - выводится краткое сообщение об использовании;
-  **-V, --version** - выводится версия программы.

4.2.6.7.1. Классы события (маски)

Следующие классы событий (маски) могут быть использованы для фильтрации событий, собираемых xentrace, см. таблицу 41.

Таблица 41 — Классы событий

ID	Описание
0x0001f000	TRC_GEN
0x0002f000	TRC_SCHED

0x0004f000	TRC_DOM0OP
0x0008f000	TRC_HVM
0x0010f000	TRC_MEM
0xfffff000	TRC_ALL

4.2.6.7.2. Подклассы событий (Больше масок)

Следующие подклассы событий (маски) также могут быть использованы для фильтрации событий, собираемых xentrace, см. таблицу 42.

Таблица 42 — Подклассы событий

ID	Описание
0x00081000	TRC_HVM_ENTRYEXIT
0x00082000	TRC_HVM_HANDLER

4.2.6.7.3. События

В *<xentrace>* собирает из буфера трассы следующие события, см. таблицу 43.

Таблица 43 — События

ID	Описание
0x0001f001	TRC_LOST_RECORDS
0x0002f001	TRC_SCHED_DOM_ADD
0x0002f002	TRC_SCHED_DOM_REM
0x0002f003	TRC_SCHED_SLEEP
0x0002f004	TRC_SCHED_WAKE
0x0002f005	TRC_SCHED_YIELD
0x0002f006	TRC_SCHED_BLOCK
0x0002f007	TRC_SCHED_SHUTDOWN
0x0002f008	TRC_SCHED_CTL
0x0002f009	TRC_SCHED_ADJDOM
0x0002f010	TRC_SCHED_SWITCH
0x0002f011	TRC_SCHED_S_TIMER_FN
0x0002f012	TRC_SCHED_T_TIMER_FN
0x0002f013	TRC_SCHED_DOM_TIMER_FN
0x0002f014	TRC_SCHED_SWITCH_INFPREV
0x0002f015	TRC_SCHED_SWITCH_INFNEXT
0x00081001	TRC_HVM_VMENTRY
0x00081002	TRC_HVM_VMEXIT
0x00082001	TRC_HVM_PF_XEN
0x00082002	TRC_HVM_PF_INJECT
0x00082003	TRC_HVM_INJ_EXC
0x00082004	TRC_HVM_INJ_VIRQ
0x00082005	TRC_HVM_REINJ_VIRQ
0x00082006	TRC_HVM_IO_READ

Таблица 43 — События

ID	Описание
0x00082007	TRC_HVM_IO_WRITE
0x00082008	TRC_HVM_CR_READ
0x00082009	TRC_HVM_CR_WRITE
0x0008200A	TRC_HVM_DR_READ
0x0008200B	TRC_HVM_DR_WRITE
0x0008200C	TRC_HVM_MSR_READ
0x0008200D	TRC_HVM_MSR_WRITE
0x0008200E	TRC_HVM_CPUID
0x0008200F	TRC_HVM_INTR
0x00082010	TRC_HVM_NMI
0x00082011	TRC_HVM_SMI
0x00082012	TRC_HVM_VMMCALL
0x00082013	TRC_HVM_HLT
0x00082014	TRC_HVM_INVLPG
0x0010f001	TRC_MEM_PAGE_GRANT_MAP
0x0010f002	TRC_MEM_PAGE_GRANT_UNMAP

xentrace_format - печать данных трассировки **Xen** с заданным форматированием.

Формат команды:

xentrace_format [I<DEFS-FILE>]

xentrace_format обрабатывает данные трассировки в двоичном формате xentrace со стандартного ввода и переформатирует их в соответствии с правилами в файле определений (I<DEFS-FILE>), выводя на стандартный вывод.

Правила в I<DEFS-FILE> должны иметь формат, приведенный ниже:

I<event_id> I<whitespace> I<format>

Каждое правило должно начинаться с новой строки.

Строка формата может содержать спецификаторы формата, например:

% (cpu) d, % (tsc) d, % (event) d, % (1) d, % (2) d, % (3) d, % (4) d, % (5) d.

Спецификатор формата «d» выводится в десятичной форме, «x» - в шестнадцатеричной и «o» - в восьмеричной. Спецификаторы соответствуют номеру ЦП, ID события, счетчику меток времени и пяти полям данных из записи

трассировки. Для каждого типа событий должно быть одно такое правило (события, не имеющие правил форматирования, игнорируются).

В зависимости от системы и ограничений скорости, с которой собираются данные трассировки, этот скрипт может не успевать за выводом xentrace, если он передается напрямую. В этом случае необходимо производить вывод xentrace в файл для off-line обработки.

4.3. Синтаксис конфигурационного файла домена

4.3.1. Описание синтаксиса

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:

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

Значения бывают следующих типов:

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

4.3.2. Опции конфигурационного файла

Обязательные ключи:

📄 *name*="NAME" — задает 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.

4.3.2.1. Общие параметры

4.3.2.1.1. Распределение ЦП

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.

Варианты указания диапазонов *CPU-LIST*:

📄 *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»;

📄 *nodes: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 приведенным 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». Для примера, приведенного выше, это будут процессоры «1», «4», «5», «7»;

📄 «2», «3-8,^5» — запросить определенный 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 привязка VCPU к физическому ЦП не задается, и виртуальные процессоры гостя могут работать на всех ЦП хоста. Если эта опция указана, пересечение пиннинговой маски VCPU, приведенной здесь, с маской схожести ПО, приведенной 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, приведенной здесь, с маской схожести ПО, приведенной 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 возможном 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.

4.3.2.1.2. Планирование ЦП

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 «0» (по умолчанию) 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**;

📖 *latency=N* — пересчитанный период, если домен выполняет интенсивный ввод-вывод. Поддерживается планировщиком **sedf**;

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

4.3.2.1.3. Распределение памяти

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 произойдет сбой.

4.3.2.1.4. События

События:

 *on_poweroff*="ACTION" — действия домена при отключении. Действия могут быть следующими:

- 1) *destroy* — уничтожить домен;
- 2) *restart* — уничтожить домен и создать новый домен с такой же конфигурацией;
- 3) *rename-restart* — переименовать домен, который перестал работать, и создать новый домен с той же конфигурацией;
- 4) *preserve* — сохранить домен. Он может быть проверен и позже уничтожен с помощью опции *xl destroy*;
- 5) *coredump-destroy* — записать дамп ядра домена в */var/xen/dump/NAME* и затем уничтожить домен;
- 6) *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 гипервизора. Значение по умолчанию: *destroy*;

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

4.3.2.1.5. 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 `ramdisk`;

📄 `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.

4.3.2.1.6. Дополнительные опции

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 дает возможность задействовать некоторые определенные функции, которые несовместимы с миграцией. На текущий момент ограничено инвариантным флагом функции **TSC** команды `cruid`, 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.

4.3.2.2. 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 гипервизора), 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 гипервизора), 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 разделенные запятой настройки `KEY=VALUE` из следующего списка:

7) `vnc=BOOLEAN` — разрешить доступ к дисплею через протокол VNC. Это позволяет активировать другие настройки, относящиеся к VNC. По умолчанию разрешено;

8) `vnclisten="ADDRESS[:DISPLAYNUM]"` — IP-адрес (и опционально VNC-номер дисплея). Если номер дисплея указан здесь, `vncdisplay` не потребуется;

9) `vncdisplay=DISPLAYNUM` — номер дисплея VNC для использования. Фактическое число TCP-порта будет равно «`DISPLAYNUM+5900`»;

10) `vncunused=BOOLEAN` — запросить VNC-поиск настройки VNC-дисплея для использования свободного порта TCP. Реально используемый дисплей может быть доступен с `xl vncviewer`;

11) `vncpasswd="PASSWORD"` — пароль для VNC-сервера;

12) `sdl=BOOLEAN` — указывает, что дисплей должен быть представлен через окно X (используя Simple DirectMedia Layer). По умолчанию этот режим не включен;

13) `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;

14) `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`»;

15) `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 фронтенда при подключении приложения к правильному каналному 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 бэкэнд. `connection=SOCKET` — бэкэнд подключится к сокету домена Unix (на пути, указанном в `path=PATH`), вызывая, слушая и принимая соединения. Бэкэнд будет функционировать как служба прокси между каналом и присоединенным сокетом. `connection=PTY` — бэкэнд создаст **pty** и **proxy** данных между каналом и главным устройством. Можно использовать команду `x1 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:

1) *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;

2) *@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 назначить устройство для *pciback* повторно, если оно еще 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 отключен, 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 остается 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**-защищенного 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 гипервизора 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).

4.3.2.2.1. 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);

 *e820_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 дает 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».

4.3.2.3. Особые опции HVM-гостя

Следующие опции применимы только к HVM-гостям.

4.3.2.3.1. Устройство загрузки

boot=[c|d|n] — выбирает эмулируемое виртуальное устройство для загрузки. Вариантами здесь являются жесткий 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 жестком диске. Значение по умолчанию: *cd*.

4.3.2.3.2. 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.

Nap=BOOLEAN — включает или выключает аппаратно поддерживаемый пейджинг (использование аппаратного вложенного страничного элемента таблицы). Выполняет функцию **EPT** (англ. *Extended Page Tables*, также расширенные таблицы страниц) в процессорах Intel и функцию **NPT** (англ. *Nested Page Tables*, также вложенные таблицы страниц) или **RVI** (англ. *Rapid Virtualisation Indexing*) в процессорах AMD. Если опцию отключить, гипервизор запустит гостя в режиме «таблицы теневых страниц», где обновления таблицы страниц гостя и/или флеш

TLB будут эмулированы. Использование NAP 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 гипервизоре: можно отключить эту функцию. Использование несинхронизованных таблиц памяти в случаях, когда гипервизор считает это целесообразным, является значением по умолчанию.

shadow_memory=MBYTES — посчитать количество мегабайт, отведенное для теневого копирования гостевых таблиц страниц (эффективно действующее в качестве кэш-памяти переведенных страниц), или использовать для состояния NAP. 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.

4.3.2.3.3. 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 включенной 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 определенных 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 гипервизора. 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) «1» -> заставить соответствующий бит принять «1»;
- 2) «0» -> установить в «0»;
- 3) «х» -> получить безопасное значение (транзит и маска с политикой по умолчанию);
- 4) «k2» -> пройти через битное значение хоста;
- 5) «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, fl6c, 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, page1gb, pat, pbe, pclmulqdq, pdcmt, pge, popcnt, pse, pse36, psn, rdtscp, skinit, smx, ss, sse, sse2, sse3, sse4_1, sse4_2, sse4a, ssse3, svm, svm_decode, svm_lbrv, svm_npt, svm_nrrips, svm_pausefilt, svm_tscrate, svm_vmcbsclean, syscall, sysenter, tbm, tm, tm2, topoext, tsc, vme, vmx, wdt, x2apic, xop, xsave, xtptr.

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, объединенных 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 определенных 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, объединенных 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.

4.3.2.3.4. 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 «rdtsc» эмулируется в исходной частоте, если не поддерживается аппаратно;

 *native_paravirt* — аналогично режиму *native*, за исключением того, что гипервизор управляет регистром TSC_AUX, чтобы домен мог определить, когда произошло восстановление/миграция, и предполагает, что домен получает/использует механизм, подобный rvclock, 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», время 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.

4.3.2.3.5. 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 распространенное большое значение равно «3072».

4.3.2.3.6. Поддержка паравиртуализации

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

1) *xen_platform_pci=BOOLEAN* — включить или отключить PCI-устройство платформы гипервизора. 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;

2) *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 MSR. 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.

4.3.2.3.7. 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 видеопамати 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 видеопамати 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 `x1 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», время 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.

4.3.2.3.8. 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 включен. Должен быть предоставлен по крайней мере один из *spiceport* или *spicetls_port*, если SPICE включен. 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*. Создает 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 определен конкретный контроллер, будет автоматически добавлен контроллер USB2. Значение по умолчанию «0».

4.3.2.3.9. 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).

usbctrl=["USBCTRL", "USBCTRL", ...] — Добавляет USB-контроллеров. Каждый (*USBCTRL*) представляет собой список разделенных запятыми параметров USB-контроллеров. Такой тип USB-контроллеров поддерживает «горячее» подключение USB устройств. Минимальные параметры добавления USB-

контроллеров в гостевую ОС *version=VERSION* указывающий версию USB-контроллера и *ports=PORTS* указывающий общее количество портов usb-контроллера.

usbdev=["USBDEV", "USBDEV", ...] — Добавляет USB устройства методом PVUSB, который использует паравиртуализированный внешний или внутренний интерфейс, аналогичный традиционным сетевым и дисковым протоколам Xen PV. В этом случае *usbdev* указывает USB-устройства, которые будут подключены к гостю при загрузке. Каждый (*USBDEV*) представляет собой список указателей USB устройств разделенных запятыми. Хост-устройства могут быть переданы в гостевую систему с использованием двух параметров *hostbus=BUSNUM* указывающий номер шины USB-устройства с точки зрения хоста и *hostaddr=DEVNUM* указывающий *devnum* устройства USB с точки зрения хоста, которые можно получить с помощью **lsusb**. Например, для устройства USB на шине «008» устройства «002» получим: *hostbus=8, hostaddr=2*.

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=C000*), 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*).

4.3.2.3.10. 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 бэкэнда 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:

1) *qemu-xen* — использовать модели устройств, объединенные в вышестоящие версии проекта **QEMU**. Эта модель устройства выбрана по умолчанию для Linux **Dom0**;

2) *Qemu-xen-traditional* — использовать модель устройства, основанную на вилке **QEMU** гипервизора. 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 передается 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 передается в качестве опции к модели устройства.

4.3.2.3.11. 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 lt nl pl ru th de
en-us fi fr-be hr it lv nl-be pt sl tr
```

Значение по умолчанию «en-us».

4.4. Конфигурация параметров сети

4.4.1. Обзор синтаксиса

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>,*].

```
"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 разделенных 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».

4.4.2. Ключевые слова

4.4.2.1. MAC-адрес

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

4.4.2.1.1. Алгоритм 1

Порядок действий:

-  создайте случайную последовательность из 6 байт;
-  установите локально управляемый бит (бит 2 первого байта);
-  очистите бит многоадресной передачи (бит 1 первого байта). Первый байт должен иметь битовый паттерн *xxxxxx10* (x — созданный рандомно бит), а остальные 5 байт генерируются случайным образом.

4.4.2.1.2. Алгоритм 2

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

4.4.2.1.3. 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.

4.4.2.2. 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.

4.4.2.3. 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`.

4.4.2.4. 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.

4.4.2.5. 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;

 *e1000* — Intel E1000;

 другая модель, поддерживаемая устройством.

4.4.2.6. Имя бэкэнд устройства

Ключевое слово **vifname** определяет имя бэкэнд 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*.

4.4.2.7. Скрипт

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

4.4.2.8. 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.

4.4.2.9. Бэкэнд домен

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

4.4.2.10. 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 задает частоту, на которой пополняются кредитные передачи VIF-устройства. По умолчанию это 50 ms.

Лимит VIF-скорости — значение «1MB/s@20ms» показывает, что доступный кредит будет равен трафику, который используется на 1 Мбайт/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.

4.5. Конфигурация параметров дисков

В разделе приведено описание опции 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 разделенных 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.

4.5.1. Позиционные параметры

4.5.1.1. 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 CD-rom отсутствие этого атрибута подразумевает пустой привод.

Специальный синтаксис:

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 подчеркивания, и не будут допустимыми как «`vdevs`». Цели, которые могут соответствовать этому синтаксису, не должны указываться в качестве позиционных параметров.

4.5.1.2. **format**

Описание: Определяет формат файла образа.

Поддерживаемые значения: **raw**, **qcow**, **qcow2**, **vhd**.

Устаревшие значения: нет.

Значение по умолчанию: **raw**.

4.5.1.3. **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.

4.5.1.4. **access**

Описание: Информация по управлению 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»).

4.5.2. Другие параметры и флаги

4.5.2.1. **devtype=<devtype>**

Описание: Определяет тип виртуального устройства.

Поддерживаемые значения: **cdrom**.

Устаревшие значения: отсутствуют.

Обязательный параметр: нет.

4.5.2.1.1. **cdrom**

Дополнительное имя для «devtype=cdrom».

4.5.2.2. **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.

4.5.2.3. **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.

4.5.2.4. **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 «/»).

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

4.5.2.5. **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.

4.5.2.6. **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 неразрезанными, чтобы избежать фрагментации файла.

4.6. 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 остается 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-rt ds` может использоваться для настройки параметров гостевого планировщика для каждой ВМ:

 `xl sched-rt ds -d <domain>` - вывести список параметров указанного <domain>;

 `xl sched-rt ds -d <domain> -p <period> -b <budget>` - установить бюджет каждого VCPU на <budget> и период на <period> для указанного <domain>.

4.7. Примеры конфигураций виртуальных машин

Далее приводятся примеры различных вариантов конфигурации виртуальных машин.

4.7.1. Конфигурация ВМ без PCI passthrough

```
builder=' hvm'

memory = 8192

# Область для MMIO диапазонов устройств. Для
ВМ без PCI passthrough добавлять данную опцию не т
н у ж д ы.

#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-qn.2017-05.ru.russoft-
nn.macavity:win-vm01-lun-0,hda,w'

# Локальные ISO – один с инсталлятором ОС,
другой – с драйверами 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 имени 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"
# Д о с т у п н о  с н а р у ж и ,  а  н е  т о л ь к о  localhost.
vnclisten=" 0.0.0.0"
# vncpasswd не задаем - вход без пароля.

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

usb=1

localtime=1

# Пусть знает, что в виртуалке, включаем
часть ABI Hyper-V
viridian=[ "all" ]
# И с п о л ь з у е м  EPT
hap=1
# У п р а в л е н и е  п и т а н и е м
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
```

Secondary VGA - NVidia - наиболее часто применяемый и стабильный режим

```
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

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

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.
```

4.7.2. Конфигурация для PV (Linux/FreeBSD) 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 прообразываем
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

# Прообразываем 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-a).
```

```
# 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.
```

4.7.3. Пример работы с Storage driver domain (PVH VM)

Примечание. 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'
```

4.7.4. PCI Passthrough - 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 принципе можно попробовать и установить с пробросом.

Примечание. 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
```

```
# К р и т и ч н о ,   п р о б р а с ы в а е м а я   к а р т а   -   н а   C P U 0
```

```
# 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' ]
```

4.7.5. 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 так же: 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 драйверы 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-е 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' ]
```

4.7.6. 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

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' ]
```

5. НАСТРОЙКА ПРОГРАММЫ – КОМПОНЕНТЫ УПРАВЛЯЮЩЕГО ДОМЕНА

Ниже приводится описание настройки компонентов для кросс-платформенного взаимодействия в информационной системе с применением ПВ «V-Софт» 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 ПВ «V-Софт». 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.

5.1. Настройка сети для использования с ВМ

5.1.1. Виртуальные сетевые интерфейсы

5.1.1.1. 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.

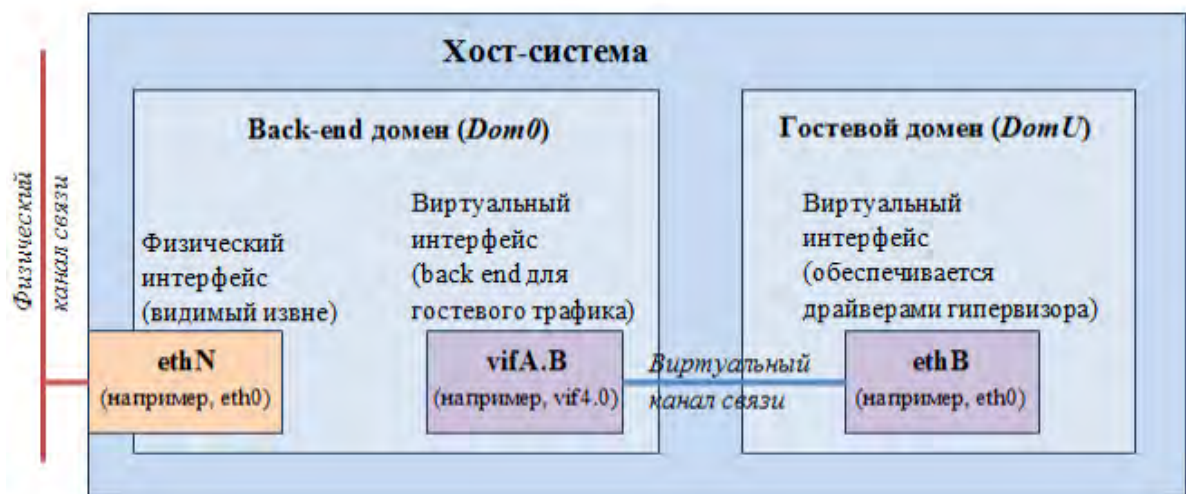
Каждое PV-устройство состоит из фронтенд устройства в гостевом домене и бэкэнд устройства в основном домене (обычно **Dom0**).

Фронтенд устройств, работающих в Linux, связан с драйвером **xen-netfront** и представлены устройством **ethN**.

В названии бэкэнд 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 фронтенд и бэкэнд связаны виртуальным каналом связи. Гостевая сеть устанавливается путем организации трафика, переходящего от бэкэнд устройства на более широкую сеть с помощью, к примеру, бриджинга (см. подпункт 92), маршрутизации или преобразования сетевых адресов (NAT).

На рис. 1 представлена схема доменов PV-устройства.



Рисунок

5.1.1.2. 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 еще 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 бэкэнд домене как сетевое устройство типа **tap**. Такие устройства также обозначаются как «vifDOMID=DEVID», что подчеркивает парную связь PV-домена и эмулированного устройства.

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

Термины PV-устройство и эмулированное устройство взаимозаменяемы.

5.1.1.3. MAC-адреса

Виртуализированным сетевым интерфейсам в доменах присваиваются MAC-адреса Ethernet. В зависимости от набора инструментов (англ. toolstack) гипервизора, 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:00:00:00:00:11'].

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

 задать адрес из диапазона, связанного с уникальным идентификатором организации (OUI);

 создать случайную последовательность из шести 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 *xxxxxx10*, где *x* — случайно сгенерированный бит. Остальные пять байт генерируются также случайным образом;

☞ задать случайный адрес внутри пространства *00:16:3e:xx:xx:xx*, где *00:16:3e* — **OUI**, назначенный для проекта гипервизора, 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 подключен хост гипервизора). 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.

5.1.2. Сетевой мост

По умолчанию, конфигурация гипервизора использует сетевой мост (также мост, бридж) в бэкэнд домене. Таким образом, все домены в сети отображаются в виде отдельных хостов.

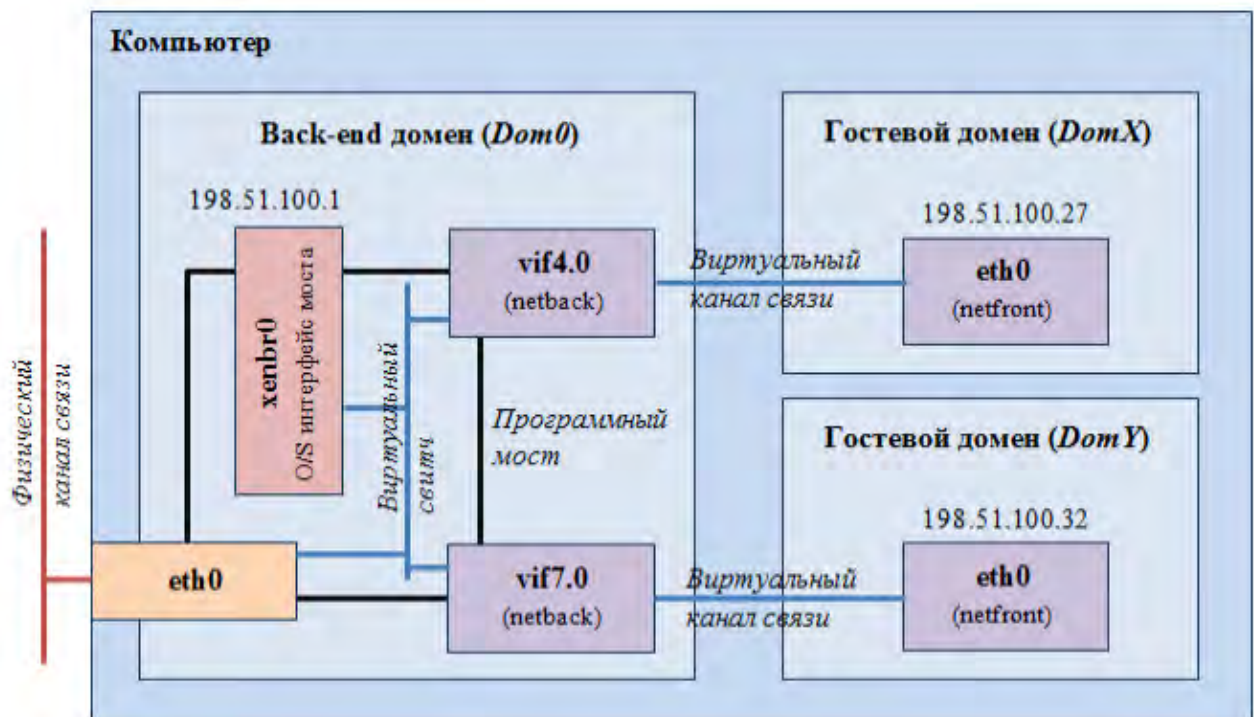
В этой конфигурации программный мост также создается в бэкэнд домене. Бэкэнд виртуальные сетевые устройства (*vifDOMID.DEVID*) добавляются к мосту наряду с физическим Ethernet-устройством — для обеспечения еще одного канала связи, помимо хоста. Опуская физическое устройство, можно создать изолированную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

Существуют две общепринятые схемы назначения имен 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 представлена схема, иллюстрирующая данные примеры.



5.1.2.1. 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.

5.1.2.2. 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». Например,

```
vif=[ "bridge=mybridge" ]
```

ИЛИ

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

Также можно создать разные интерфейсы, присоединенные к разным мостам:

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

5.1.2.3. Мостовые петли

Отключение протокола STP на мостах гипервизора является распространенной (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.

5.2. Организация прямого доступа ВМ к PCI/PCIe устройствам

5.2.1. 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, НВА, 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 расчетов 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 передает 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 разрешен 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 гипервизора; 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 гипервизору 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**.

5.2.2. Использование прямой передачи

5.2.2.1. 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.

5.2.2.1.1. 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.

5.2.2.1.2. Динамическое назначение с xl

Если модуль хен-рсібаск загружен как модуль и не встроен в ядро, убедитесь, что в **Dom0** загружен рсібаск-модуль:

```
# modprobe хен-рсібаск
```

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
```

5.2.2.1.3. Динамическое назначение с sysfs

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

Убедитесь, что в **Dom0** загружен рсібаск-модуль:

```
# modprobe хен-рсібаск
```

Основные этапы:

-  отмените привязку к старому драйверу;
-  создайте новый слот в рсібаск для устройства;
-  создайте привязку к рсібаск.

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
```

5.2.2.2. 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
```

5.2.2.3. 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.

5.2.2.3.1. 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' ]
```

5.2.2.3.2. 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>
```

5.2.2.4. 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. К примеру,

```
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.

5.2.2.4.1. Разрешающий режим для xl

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 определенного 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"
```

5.2.3. Дополнительная информация и часто задаваемые вопросы

В старшей версии Linux 3.1.0 (и более поздних, например, стандартной для гипервизор 5.2.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
```

5.2.3.1. 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.

5.2.3.2. 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** прочтет строку неправильно.

5.2.3.3. 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.

5.2.3.4. 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 гипервизора (`/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 гипервизора в **grub.cfg** и перезагрузите систему.

5.2.3.5. В моем оборудовании/плате имеется IOMMU, но гипервизор не включает аппаратно поддерживаемую виртуализацию IO

К сожалению, 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 гипервизора в **grub.conf**;

 прочтите сообщения о загрузке гипервизора, чтобы узнать, включена или отключена виртуализация IO;

 если гипервизор сообщает о неисправном BIOS, отправьте отчет об этом поставщикам системы/платы.

5.2.3.6. Как проверить, поддерживает ли устройство 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 окружающих строк следует изменить так, чтобы увидеть, к какому устройству относится флаг.

Пример.

```
root@dom0image:~# lspci -vv | grep -C 22 -i flreset+ MultHdrRecCap- MultHdrRecEn-
TLPPfxPres- HdrLogCap- HeaderLog: 00000000 00000000 00000000 00000000 Kernel driver
in use: pciback 03:00.0 Non-Volatile memory controller: Samsung Electronics Co Ltd
NVMe SSD Controller SM951/PM951 (rev 01) (prog-if 02 [NVM Express]) Subsystem:
Samsung Electronics Co Ltd Device a801 Physical Slot: 5 Control: I/O+ Mem+ BusMaster+
SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR+ FastB2B- DisINTx+ Status: Cap+
66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
INTx- Latency: 0, Cache Line Size: 32 bytes Interrupt: pin A routed to IRQ 40 NUMA
node: 0 Region 0: Memory at fb210000 (64-bit, non-prefetchable) [size=16K] Region 2:
I/O ports at d000 [size=256] Expansion ROM at fb200000 [disabled] [size=64K]
Capabilities: [40] Power Management version 3 Flags: PMEClk- DSI- D1- D2-
AuxCurrent=0mA PME (D0-, D1-, D2-, D3hot-, D3cold-) Status: D0 NoSoftRst+ PME-Enable-
DSel=0 DScale=0 PME- Capabilities: [50] MSI: Enable- Count=1/8 Maskable- 64bit+
Address: 0000000000000000 Data: 0000 Capabilities: [70] Express (v2) Endpoint, MSI
00 DevCap: MaxPayload 128 bytes, PhantFunc 0, Latency L0s unlimited, L1 unlimited
ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset+ SlotPowerLimit 25.000W DevCtl: Report
errors: Correctable+ Non-Fatal+ Fatal+ Unsupported+ RlxdOrd- ExtTag- PhantFunc-
AuxPwr- NoSnoop+ FLReset- MaxPayload 128 bytes, MaxReadReq 512 bytes DevSta: CorrErr+
UncorrErr- FatalErr- UnsuppReq+ AuxPwr+ TransPend- LnkCap: Port #0, Speed 8GT/s,
Width x4, ASPM L1, Exit Latency L1 <64us ClockPM+ Surprise- LLActRep- BwNot-
ASPMOptComp+ LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- CommClk+ ExtSynch-
ClockPM- AutWidDis- BWInt- AutBWInt- LnkSta: Speed 8GT/s, Width x4, TrErr- Train-
```

SlotClk+ DLActive- BWMgmt- ABWMgmt- DevCap2: Completion Timeout: Not Supported,
 TimeoutDis+, LTR+, OBFF Not Supported AtomicOpsCap: 32bit- 64bit- 128bitCAS- DevCtl2:
 Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled AtomicOpsCtl:
 ReqEn- LnkCtl2: Target Link Speed: 8GT/s, EnterCompliance- SpeedDis- Transmit Margin:
 Normal Operating Range, EnterModifiedCompliance- ComplianceSOS- Compliance De-
 emphasis: -6dB LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete+,
 EqualizationPhase1+ EqualizationPhase2+, EqualizationPhase3+,
 LinkEqualizationRequest- Capabilities: [b0] MSI-X: Enable+ Count=9 Masked- Vector
 table: BAR=0 offset=00003000 PBA: BAR=0 offset=00002000 Capabilities: [100 v2]
 Advanced Error Reporting -- 04:00.0 Ethernet controller: Intel Corporation I350
 Gigabit Network Connection (rev 01) Subsystem: Super Micro Computer Inc Device 1521
 Physical Slot: 8 Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr-
 Stepping- SERR+ FastB2B- DisINTx+ Status: Cap+ 66MHz- UDF- FastB2B- ParErr-
 DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx- Latency: 0, Cache Line
 Size: 32 bytes Interrupt: pin A routed to IRQ 26 NUMA node: 0 Region 0: Memory at
 fb120000 (32-bit, non-prefetchable) [size=128K] Region 2: I/O ports at c020 [size=32]
 Region 3: Memory at fb144000 (32-bit, non-prefetchable) [size=16K] Capabilities: [40]
 Power Management version 3 Flags: PMEClk- DSI+ D1- D2- AuxCurrent=0mA PME (D0+, D1-, D2-
 , D3hot+, D3cold+) Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=1 PME- Capabilities:
 [50] MSI: Enable- Count=1/1 Maskable+ 64bit+ Address: 0000000000000000 Data: 0000
 Masking: 00000000 Pending: 00000000 Capabilities: [70] MSI-X: Enable+ Count=10
 Masked- Vector table: BAR=3 offset=00000000 PBA: BAR=3 offset=00002000 Capabilities:
 [a0] Express (v2) Endpoint, MSI 00 DevCap: MaxPayload 512 bytes, PhantFunc 0, Latency
 L0s <512ns, L1 <64us ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset+ SlotPowerLimit
 0.000W DevCtl: Report errors: Correctable+ Non-Fatal+ Fatal+ Unsupported+ RlxdOrd-
 ExtTag- PhantFunc- AuxPwr- NoSnoop+ FLReset- MaxPayload 256 bytes, MaxReadReq 512
 bytes DevSta: CorrErr+ UncorrErr- FatalErr- UnsupReq+ AuxPwr+ TransPend- LnkCap:
 Port #2, Speed 5GT/s, Width x4, ASPM L0s L1, Exit Latency L0s <4us, L1 <32us ClockPM-
 Surprise- LLActRep- BwNot- ASPMOptComp+ LnkCtl: ASPM Disabled; RCB 64 bytes Disabled-
 CommClk+ ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt- LnkSta: Speed 5GT/s, Width
 x4, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt- DevCap2: Completion Timeout:
 Range ABCD, TimeoutDis+, LTR+, OBFF Not Supported
 AtomicOpsCap: 32bit- 64bit- 128bitCAS-
 DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled
 AtomicOpsCtl: ReqEn-
 LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-
 Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
 Compliance De-emphasis: -6dB
 LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete-,
 EqualizationPhase1-
 EqualizationPhase2-, EqualizationPhase3-, LinkEqualizationRequest-
 Capabilities: [100 v2] Advanced Error Reporting
 UESa: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC-
 UnsupReq- ACSViol-
 UEMsk: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC-

UnsupReq- ACSViol-

UESvrt: DLP+ SDES+ TLP- FCP+ CmpltTO- CmpltAbrt- UnxCmplt- RxOF+ MalftTLP+ ECRC-

UnsupReq- ACSViol-

--

04:00.1 Ethernet controller: Intel Corporation I350 Gigabit Network Connection (rev 01)

Subsystem: Super Micro Computer Inc Device 1521

Physical Slot: 8

Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR+ FastB2B- DisINTx+

Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-

Latency: 0, Cache Line Size: 32 bytes

Interrupt: pin B routed to IRQ 28

NUMA node: 0

Region 0: Memory at fb100000 (32-bit, non-prefetchable) [size=128K]

Region 2: I/O ports at c000 [size=32]

Region 3: Memory at fb140000 (32-bit, non-prefetchable) [size=16K]

Capabilities: [40] Power Management version 3

Flags: PMEClk- DSI+ D1- D2- AuxCurrent=0mA PME (D0+, D1-, D2-, D3hot+, D3cold+)

Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=1 PME-

Capabilities: [50] MSI: Enable- Count=1/1 Maskable+ 64bit+

Address: 0000000000000000 Data: 0000

Masking: 00000000 Pending: 00000000

Capabilities: [70] MSI-X: Enable+ Count=10 Masked-

Vector table: BAR=3 offset=00000000

PBA: BAR=3 offset=00002000

Capabilities: [a0] Express (v2) Endpoint, MSI 00

DevCap: MaxPayload 512 bytes, PhantFunc 0, Latency L0s <512ns, L1 <64us

ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset+ SlotPowerLimit 0.000W

DevCtl: Report errors: Correctable+ Non-Fatal+ Fatal+ Unsupported+

RlxdOrd- ExtTag- PhantFunc- AuxPwr- NoSnoop+ FLReset-

MaxPayload 256 bytes, MaxReadReq 512 bytes

DevSta: CorrErr+ UncorrErr- FatalErr- UnsupReq+ AuxPwr+ TransPend-

LnkCap: Port #2, Speed 5GT/s, Width x4, ASPM L0s L1, Exit Latency L0s <4us, L1

<32us

ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+

LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- CommClk+

ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-

LnkSta: Speed 5GT/s, Width x4, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-

DevCap2: Completion Timeout: Range ABCD, TimeoutDis+, LTR+, OBFF Not Supported

AtomicOpsCap: 32bit- 64bit- 128bitCAS-

DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled

AtomicOpsCtl: ReqEn-

LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete-,

EqualizationPhase1-

EqualizationPhase2-, EqualizationPhase3-, LinkEqualizationRequest-

Capabilities: [100 v2] Advanced Error Reporting

UESta: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmpl- RxOF- MalfTLP- ECRC-
UnsupReq- ACSViol-

UEMsk: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmpl- RxOF- MalfTLP- ECRC-
UnsupReq- ACSViol-

UESvrt: DLP+ SDES+ TLP- FCP+ CmpltTO- CmpltAbrt- UnxCmpl- RxOF+ MalfTLP+ ECRC-
UnsupReq- ACSViol-

CESta: RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+

CEMsk: RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+

AERCap: First Error Pointer: 00, ECRGGenCap+ ECRGGenEn- ECRCCbkCap+ ECRCCbkEn-

В выводе команды выше видно, что три 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.

5.2.3.7. Использование pci-stub

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

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

Тем не менее, остается один из вариантов использования pci-stub, когда нужно экранировать какое-либо устройство при загрузке без менеджера ВМ, т. е. только «dom0kernel» и «dom0image».

5.2.3.8. Передача нескольких устройств PCI

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

5.3. Домены ввода-вывода

5.3.1. Драйвер-домены

Драйвер-домен — это непривилегированный домен ввода-вывода, который отвечает за определенную часть аппаратного обеспечения. Он работает на минимальном ядре с одним конкретным аппаратным драйвером и бэкэнд драйвером для данного класса устройств. При падении аппаратного драйвера остальные домены (включая **Dom0**) продолжают функционировать; после перезагрузки драйвер-домена они могут снова использовать аппаратное обеспечение.

Подробные примеры настройки драйвер-доменов приведены в подразделе 174.

5.3.1.1. Преимущества

5.3.1.1.1. Производительность

Размещение всех бэкэнд 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).

5.3.1.1.2. Повышенная надежность

Аппаратные драйверы являются самой ненадежной 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.

5.3.1.1.3. 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 VM.

5.3.1.1.4. Использование проприетарных драйверов и микропрограмм

Сертифицированная для работы с гостайной система 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 защищенным режимом, и используется для запуска недоверенного кода, например OS Windows. Он обеспечивает полноценную изоляцию к памяти СВТ, 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.

5.3.1.2. 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.

5.3.2. 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**, передается как **/dev/xvda** в **Dom0** (идентификатор домена — «0»).

5.3.3. 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.

5.3.3.1. Создание конфигурационного файла домена

Пример конфигурационного файла 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' ]
```

5.3.3.2. Определение идентификатора шины USB-устройства

Выведите список устройств с идентификаторами их шин. Например,

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

5.3.3.3. Предоставление доступа к USB-устройству

Предоставьте доступ к устройству, указав нужный идентификатор шины. Например,

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

5.3.3.4. 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. Например,

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

5.3.3.5. Отключение USB-устройства со стороны гостевой VM

```
usbip detach --port <порт>
```

5.3.4. 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" ]
```

5.3.5. Эмуляционный домен

Stub-домен (также эмуляционный домен) — это специализированный домен гипервизора, 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, гипервизор 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.

5.3.5.1. Преимущества

5.3.5.1.1. 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`).

5.3.5.1.2. 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.

5.3.5.1.3. 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 дожидаться, когда планировщик гипервизора передаст управление управляющему домену, а затем планировщик управляющего домена передаст управление процессу модели устройств.

5.3.5.1.4. 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.

5.3.5.2. Конфигурационные параметры stub-домена

Конфигурационные параметры для stub-доменов представлены в таблице 44.

Таблица 44 — Конфигурационные параметры для stub-доменов

Параметр	Описание
kernel	tmem-freeable
ramdisk	Выводит информацию о том, сколько свободной памяти (Мбайт) использует TМЕМ
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. <i>Extended Page Tables</i>) у Intel и NPT (англ. <i>Nested Page Tables</i>) или RVI (англ. <i>Rapid Virtualisation Indexing</i>) у AMD. Работает только для HVM-гостей. Если выключено, то гипервизор будет запускать гостя в режиме <i>shadow page table</i> , 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 <i>memory</i>
rdm	См. руководство по xl.cfg
pci	См. руководство по xl.cfg

Конфигурация при использовании stub-доменов разбивается на два 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.

5.3.5.3. 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
device_model_stubdomain = 1
device_model_stubdomain_override = 1
device_model_version = "qemu-xen"
stubdomain_version = "linux"
stubdomain_config_file = <путь_к_конфигурационному_файлу>
```

5.3.5.4. Конфигурационный файл stub-домена

```
memory = 1024 maxmem = 1024 stubdom_disk = "stubdom.img" kernel = "/boot/dom0kernel"
cmdline = "debug console=hvc0 rootfstype=ext4 root=/dev/xvdz init=/init" vif =
[ "mac=00:11:22:33:44:55" ]
```

5.3.5.5. 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.

5.3.6. Stub-домен в роли USB-домена для клавиатуры и мыши

При организации прямого доступа к USB-клавиатуре и USB-мышь 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' ]
```

5.4. 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:

 войти в систему под учетной записью администратора (root):

```
Hypervisor Dom0 image dom0image login: root Password: root@dom0image:~#
```

 отключить режим SELinux «enforcing»:

```
root@dom0image:~# setenforce 0
```

 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
```

 произвести разбиение диска на разделы:

```
1. 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
```

 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
```

 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.
```

 создать точку монтирования и примонтировать образ ВМ:

```
root@dom0image:~# mkdir -p /mnt root@dom0image:~# mount /dev/mapper/vg0-testvm01p2
/mnt
```

 создать файл /etc/fstab в образе ВМ из имеющегося в /etc/fstab, исключив монтируемые с **UUID** разделы:

```
root@dom0image:~# cat /etc/fstab | grep -v "^UUID" >/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
```

 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», 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/S70libvirt' -> "/mnt/etc/init.d/disabled/S70libvirt"
' /mnt/etc/init.d/S71virtlogd' -> "/mnt/etc/init.d/disabled/S71virtlogd"
' /mnt/etc/init.d/S72libvirt-guests' -> "/mnt/etc/init.d/disabled/S72libvirt-
guests"
З а д а т ь и м я х о с т а :

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
```



размонтировать «/mnt»:

```
root@dom0image:~# umount /mnt
```



создать конфигурационный файл для ВМ следующего содержания:

```
#builder=' hvm' memory = 1024 vcpus = 1 name = "testvm01" kernel =
"/boot/dom0kernel" 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»;



запустить виртуальную машину **testvm01** в паравиртуальном режиме (строка *builder="hvm"* закомментирована, используется прямая загрузка ядра /boot/dom0kernel):

```
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 (early) [ 0.000000] x86/fpu: Supporting
XSAVE feature 0x001: "x87 floating point registers" (early) [ 0.000000] x86/fpu:
Supporting XSAVE feature 0x002: "SSE registers" (early) [ 0.000000] x86/fpu:
Supporting XSAVE feature 0x004: "AVX registers"
(early) [ 0.000000] x86/fpu: xstate_offset[2]: 576, xstate_sizes[2]: 256
(early) [ 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.
....
[ 1.188044] netconsole: network logging started
[ 1.188049] hctosys: unable to open rtc device (rtc0)
[ 1.192170] EXT4-fs (xvda2): mounted filesystem with ordered data mode. Opts:
(null)
[ 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
INIT: version 2.88 booting
```

```
[ 1.225484] EXT4-fs (xvda2): re-mounted. Opts: (null)
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:
```



войти в виртуальную машину с правами администратора (root) - пароль идентичный:

Hypervisor Dom0 image testvm01 login: root Password: root@testvm01:~#



создать запись в /etc/fstab для первого раздела диска VM:

```
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
```



примонтировать /boot и проверить:

```
root@testvm01:~# mount /boot root@testvm01:~# df -h | grep /boot /dev/xvda1      119M
1,6M  109M   2% /boot
```



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:~#
```



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 /dom0kernel 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 введенным содержимым;



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 /dom0kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4
selinux=0 nomodeset } root@testvm01:~#
```



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:~#
```



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/dom0kernel" #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;



скопировать ядро *dom0image* в раздел загрузки дискового образа ВМ:

```
root@dom0image:~# mount /dev/mapper/vg0-testvm01p1 /mnt root@dom0image:~# cp -v
/boot/dom0kernel /mnt ' /boot/dom0kernel' -> "/mnt/dom0kernel" root@dom0image:~#
umount /mnt
```



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=/dom0kernel 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, using "standard" format. [ 0.000000] x86/fpu: Using "eager"
FPU context switches. [ 0.000000] e820: BIOS-provided physical RAM map:
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[ 0.000000] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000003ffffeff] usable
[ 0.000000] BIOS-e820: [mem 0x00000000003ffff000-0x00000000003fffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000fc000000-0x00000000ffffffff] reserved
[ 0.000000] NX (Execute Disable) protection: active [ 0.000000] SMBIOS 2.4
present. [ 0.000000] Hypervisor detected: Xen [ 0.000000] 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;
[ 0.357482] xvda: xvda1 xvda2
[ 0.448109] console [netcon0] enabled
[ 0.448121] netconsole: network logging started
[ 0.448166] rtc_cmos 00:02: setting system clock to 2018-06-22 15:55:29 UTC
(1529682929)
[ 0.450228] EXT4-fs (xvda2): mounted filesystem with ordered data mode. Opts:
(null)
[ 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;

 зайти с учетной записью администратора (root):

```
Hypervisor Dom0 image testvm01 login: root Password: root@testvm01:~#
```

 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)
```

 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 этих команд подготовлена тестовая виртуальная машина **testvm01**.

5.5. Клонирование виртуальной машины со сменой имени

Для клонирования виртуальной машины **testvm01** и смены имени «клона» на **testvm02** необходимо выполнить следующие действия:

 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 создает том **testvm02** идентичного **testvm01** размера, вторая команда делает копию **testvm01** в **testvm02**.

 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
  1      1049kB  134MB   133MB   primary ext4
  2      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
```

 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
```

 размонтировать корневой раздел с образа диска **testvm02**:

```
root@dom0image:~# umount /mnt
```

 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/dom0kernel" #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
```



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
[ 0.000000] Command line: BOOT_IMAGE=/dom0kernel 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, using "standard" format. [ 0.000000] x86/fpu: Using "eager"
FPU context switches. [ 0.000000] e820: BIOS-provided physical RAM map:
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x0000000000009fbff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000000009fc00-0x0000000000009ffff] reserved
[ 0.000000] BIOS-e820: [mem 0x000000000000f0000-0x000000000000fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000000100000-0x000000000003ffffefff] usable
[ 0.000000] BIOS-e820: [mem 0x000000000003ffff000-0x000000000003fffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x00000000fc0000000-0x00000000ffffffffff] reserved
[ 0.000000] NX (Execute Disable) protection: active [ 0.000000] SMBIOS 2.4
present. [ 0.000000] Hypervisor detected: Xen [ 0.000000] 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
[ffff8844800f25e0]
[ 0.000000] Scanning 1 areas for low memory corruption
[ 0.000000] Using GB pages for direct mapping
[ 0.000000] ACPI: Early table checksum verification disabled
....
[ 0.341672] microcode: sig=0x306f2, pf=0x1, revision=0x3c
[ 0.341703] microcode: Microcode Update Driver: v2.01
<tigran@aivazian.fsnet.co.uk>, Peter Oruba
[ 0.341803] registered taskstats version 1
[ 0.344825] random: fast init done
```

```
[ 0.351104] blkfront: xvda: flush diskcache: enabled; persistent grants: enabled;
indirect descriptors: enabled;
[ 0.352179] xvda: xvda1 xvda2
[ 0.444030] console [netcon0] enabled
[ 0.444042] netconsole: network logging started
[ 0.444088] rtc_cmos 00:02: setting system clock to 2018-06-22 16:26:59 UTC
(1529684819)
[ 0.445955] EXT4-fs (xvda2): mounted filesystem with ordered data mode. Opts:
(null)
[ 0.522588] VFS: Mounted root (ext4 filesystem) readonly on device 202:2.
[ 0.522791] devtmpfs: mounted
[ 0.523311] Freeing unused kernel memory: 1328K
[ 0.523326] Write protecting the kernel read-only data: 12288k
[ 0.523671] Freeing unused kernel memory: 1920K
[ 0.526858] Freeing unused kernel memory: 1992K
INIT: version 2.88 booting
[ 0.543696] EXT4-fs (xvda2): re-mounted. Opts: (null)
[ 0.553095] 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
testvm02 login:
```

 войти в тестовую машину **testvm02** с учетной 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 видно, что после выключения ВМ управление вернулось к консоли Гипервизора.

Описанный выше способ — клонирование тома ВМ и замена параметров конфигурации — может быть использован для создания дополнительных ВМ на основе образца по мере необходимости.

5.6. Настройка менеджера томов LVM

По умолчанию гипервизор устанавливается таким образом, что он занимает все свободное пространство на целевом носителе. Это позволяет локально хранить инсталляционные образы дисков ПО, образы дисков с данными, но иногда требуется использование блочных устройств для образов дисков виртуальных машин (как минимум, по соображениям производительности VM).

Порядок настройки менеджера томов **LVM** для проведения тестирования приведен ниже. В примере используется блочное устройство «/dev/nvme0n1» — твердотельный SSD NVMe накопитель. На другом оборудовании блочное устройство, использованное при установке гипервизор, может отличаться.

Приведенная далее процедура предполагает размер накопителя, на который произведена инсталляция, не менее 240 GB:

 войти в систему под учетной записью администратора (root):

```
Hypervisor Dom0 image dom0image login: root Password: root@dom0image:~#
```

 отключить «форсированный» режим SELinux:

```
root@dom0image:~# setenforce 0
```

 размонтировать «/home» и убедиться, что операция выполнена:

```
root@dom0image:~# umount /home root@dom0image:~# 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
```

 проверить разбиение диска, и поменять тип раздела «4» на **lvm**:

```
root@dom0image:~# parted /dev/nvme0n1 u MiB p Model: Unknown (unknown) Disk
/dev/nvme0n1: 488386MiB Sector size (logical/physical): 512B/512B Partition Table:
```

```

gpt Disk Flags: Number Start End Size File system Name Flags
1 1,00MiB 16,0MiB 15,0MiB LegacyBoot bios_grub, legacy_boot
2 16,0MiB 1024MiB 1008MiB ext4 dom0boot 3 1024MiB 9216MiB
8192MiB ext4 dom0var 4 9216MiB 488386MiB 479170MiB ext4
dom0data root@dom0image:~# parted /dev/nvme0n1 set 4 lvm on Information: You may need
to update /etc/fstab. root@dom0image:~# parted /dev/nvme0n1 u MiB p Model: Unknown
(unknown) Disk /dev/nvme0n1: 488386MiB Sector size (logical/physical): 512B/512B
Partition Table: gpt Disk Flags: Number Start End Size File system
Name Flags 1 1,00MiB 16,0MiB 15,0MiB LegacyBoot
bios_grub, legacy_boot 2 16,0MiB 1024MiB 1008MiB ext4 dom0boot 3
1024MiB 9216MiB 8192MiB ext4 dom0var 4 9216MiB 488386MiB
479170MiB ext4 dom0data lvm

```

 преобразовать раздел «4» в физический том **LVM** (на другом оборудовании UUID и размеры могут отличаться):

```

root@dom0image:~# wipefs /dev/nvme0n1p4 DEVICE OFFSET TYPE UUID
LABEL nvme0n1p4 0x438 ext4 356a4fc3-f071-44a4-97bf-39997fd9158c dom0home
root@dom0image:~# wipefs -a /dev/nvme0n1p4 /dev/nvme0n1p4: 2 bytes were erased at
offset 0x00000438 (ext4): 53 ef root@dom0image:~# pvcreate /dev/nvme0n1p4 Physical
volume "/dev/nvme0n1p4" successfully created. root@dom0image:~# pvs PV VG
Fmt Attr PSize PFree /dev/nvme0n1p4 lvm2 --- 467,94g 467,94g

```

 создать группу томов «vg0», а в ней том **home** размером 64GB:

```

root@dom0image:~# vgcreate vg0 /dev/nvme0n1p4 Volume group "vg0" successfully created
root@dom0image:~# lvcreate -n home -l 16384 vg0 Logical volume "home" created.
root@dom0image:~# lvs LV VG Attr LSize Pool Origin Data% Meta% Move Log
Cpy%Sync Convert home vg0 -wi-a----- 64,00g

```

 создать в томе **home** файловую систему «ext4», определить **UUID**, обновить /etc/fstab и примонтировать «/home»:

```

root@dom0image:~# 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@dom0image:~# cat /etc/fstab | grep -v "/home" >/etc/fstab.new

```

```

root@dom0image:~# printf "UUID=%s¥ t/home¥ text4¥
tdefaults, noexec, noatime, nodiratime, discard¥ t0 2¥ n" adf27af7-076d-4ed3-b96e-
d8f3fd8a7222 >>/etc/fstab.new root@dom0image:~# 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@dom0image:~# cat
/etc/fstab.new >/etc/fstab root@dom0image:~# mount /home root@dom0image:~# df -h
Filesystem                Size      Used Avail Use% Mounted on /dev/root                488M
328M  146M  70% / devtmpfs                  1,6G      0  1,6G   0% /dev tmpfs
1,6G  388K  1,6G   1% /run tmpfs                1,6G  4,0K  1,6G   1% /tmp
/dev/nvme0n1p2             977M     72M   839M   8% /boot /dev/nvme0n1p3         7,9G   51M
7,4G   1% /var tmpfs              5,0M      0  5,0M   0% /run/lock tmpfs
653M      0 653M   0% /run/shm /dev/mapper/vg0-home        63G   53M   60G   1% /home

```



восстановить контексты безопасности, сохранить настройки системы и перезагрузить тестовую ЭВМ:

```

root@dom0image:~# restorecon -R / root@dom0image:~# ./save_dom0img.sh Н а й д е н
о б р а з dom0image по п у т и /boot/dom0image... Р а с п а к о в к а
о б р а з а dom0image: /boot/dom0image (1/1) 100 %          56,4 MiB / 512,0 MiB =
0,110  129 MiB/s      0:03 dom0image п р и м о н т и р о в а н в
/tmp/tmp.MDx848SyAP. С о х р а н е н и е и з м е н е н и й...
П е р е м а р к и р о в к а Ф С dom0image м а н д а т н ы м и м е т к а м и в
с о о т в е т с т в и и с к о н ф и г у р а ц и е й...
Р а з м о н т и р о в а н и е с л у ж е б н ы х Ф С и о б р а з а...
С ж а т и е о б р а з а dom0image: /tmp/dom0image (1/1) 100 %          56,9 MiB /
512,0 MiB = 0,111  4,1 MiB/s      2:06 С о х р а н я е м к о н т р о л ь н ы е
с у м м ы : /boot/dom0image /boot/dom0kernel /boot/hypervisor.gz /boot/policy.mls
г о т о в о . И з м е н е н и я в н а с т р о й к а х с и с т е м ы
с о х р а н е н ы в /boot/dom0image... root@dom0image:~# shutdown -r now Broadcast
message from root@dom0image (pts/1) (Thu Jun 13 01:54:30 2018): The system is going
down for reboot NOW!

```



после перезагрузки, зайти в систему с учетной записью администратора (root) и проверить корректность настроек после перезагрузки:

```

root@dom0image:~# lvs LV VG Attr LSize Pool Origin Data% Meta% Move Log
Cpy%Sync Convert home vg0 -wi-ao----- 64,00g root@dom0image:~# df -h Filesystem
Size Used Avail Use% Mounted on /dev/root 488M 328M 146M 70% /
devtmpfs 1,6G 0 1,6G 0% /dev tmpfs 1,6G 388K
1,6G 1% /run tmpfs 1,6G 4,0K 1,6G 1% /tmp /dev/nvme0n1p2
977M 73M 838M 8% /boot /dev/nvme0n1p3 7,9G 51M 7,4G 1% /var tmpfs
5,0M 0 5,0M 0% /run/lock
tmpfs 653M 0 653M 0% /run/shm
/dev/mapper/vg0-home 63G 53M 60G 1% /home
root@dom0image:~# vgsdisplay vg0
--- Volume group ---
VG Name vg0
System ID
Format lvm2
Metadata Areas 1
Metadata Sequence No 2
VG Access read/write
VG Status resizable
MAX LV 0
Cur LV 1
Open LV 1
Max PV 0
Cur PV 1
Act PV 1
VG Size 467,94 GiB
PE Size 4,00 MiB
Total PE 119792
Alloc PE / Size 16384 / 64,00 GiB
Free PE / Size 103408 / 403,94 GiB
VG UUID TMYBiz-yFzk-mRAU-5wb0-Vp3m-heKe-ImJnnq

```

Как видно из вывода команды выше, размер «/home» стал 63 GiB (64 GiB 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.

5.7. Сценарии настройки в гипервизор

Ниже приводится описание примеров настроек, присутствующих в гипервизоре.

5.7.1. 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 два сетевых интерфейса — один для управления гипервизор и доступа администратора, и второй (или более) — для подключения виртуальных машин.

По умолчанию в образе *dom0image* 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 **eth1**, приведен ниже:

```
### В примере ниже автоматически создается
сетевой мост для виртуальных машин. if ! brctl show
${ bridge} 2>/dev/null 1>/dev/null ; then printf "¥ нС о з д а е м б р и д ж
${ bridge} ..." brctl addbr ${ bridge} printf " done.¥ н" else printf "¥ нБ р и д ж
${ bridge} у ж е с о з д а н:¥ н" 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
"Д о б а в л я е м и н т е р ф е й с ${ bridge_iface} в б р и д ж
${ bridge} ..." brctl addif ${ bridge} ${ bridge_iface} printf " г о т о в о.¥ н"
else printf "И н т е р ф е й с ${ bridge_iface} у ж е в б р и д ж е
${ bridge} .¥ н" 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.

5.7.2. Настройка сетевых блочных устройств

В сценарии `/etc/rc.local` также приведен 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:

# Создаем каталог /home:
mkdir -p /home

# Далее идет 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**, и подключение к нему.

В случае использования устройств iSCSI как томов виртуальных машин в АСЗИ, следует назначить подключенному устройству (в сценарии — *device*) соответствующую мандатную метку.

5.7.3. 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
```

5.7.4. 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
BM!" ;; esac done
```

5.8. Установка ОС в Виртуальную Машину

Ниже приводится краткая инструкция по установке ОС в виртуальную машину, включая подготовку к работе с PCI Passthrough и пара-виртуальными драйверами.

5.8.1. Подготовка виртуального диска и инсталляционного носителя

Для установки системы необходимо определить требуемый объем дискового пространства для целевой ОС. Например, в MS Windows, с учетом того, что не требуется использование режима гибернации, объема 64 GiB для системного диска — более чем достаточно. ОС семейства UNIX/Linux требуют значительно меньше

дискового пространства, для типовой инсталляции достаточно 16GiB 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 двух ВМ — с ОС Windows и ОС Linux (Ubuntu).

Для создания группы томов **LVM** необходимо выполнить шаги, предусмотренные в подразделе 158 настоящего документа.

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 — «vg0», размер указывается в блоках 4 К, для 64 GiB это будет 16384, для 16 GiB — 4096 и зададим имена томов ВМ — **vm01** и **vm02**:

```
root@dom0image:~# lvcreate -n vm01 -l 16384 vg0 Logical volume "vm01" created.
root@dom0image:~# lvcreate -n vm02 -l 16384 vg0 Logical volume "vm02" created.
root@dom0image:~# 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
```

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@dom0image:~# 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 н о я 15 07:35 ubuntu-mate-16.04.3-desktop-
amd64.iso
```

5.8.2. Настройка VM с ОС Windows

Для настройки VM необходимо выполнить следующее:

 перед запуском VM необходимо убедиться, что создан сетевой мост «xenbr0» (см. пункт 162):

```
root@dom0image:~# brctl show bridge name bridge id STP enabled interfaces xenbr0
8000.0cc47a98b3dd no eth1
```

 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.
Требуется 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 # 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 винды – 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 приведенном выше конфигурационном файле закомментированы строки, которые понадобятся далее, для организации прямого доступа к GPU.

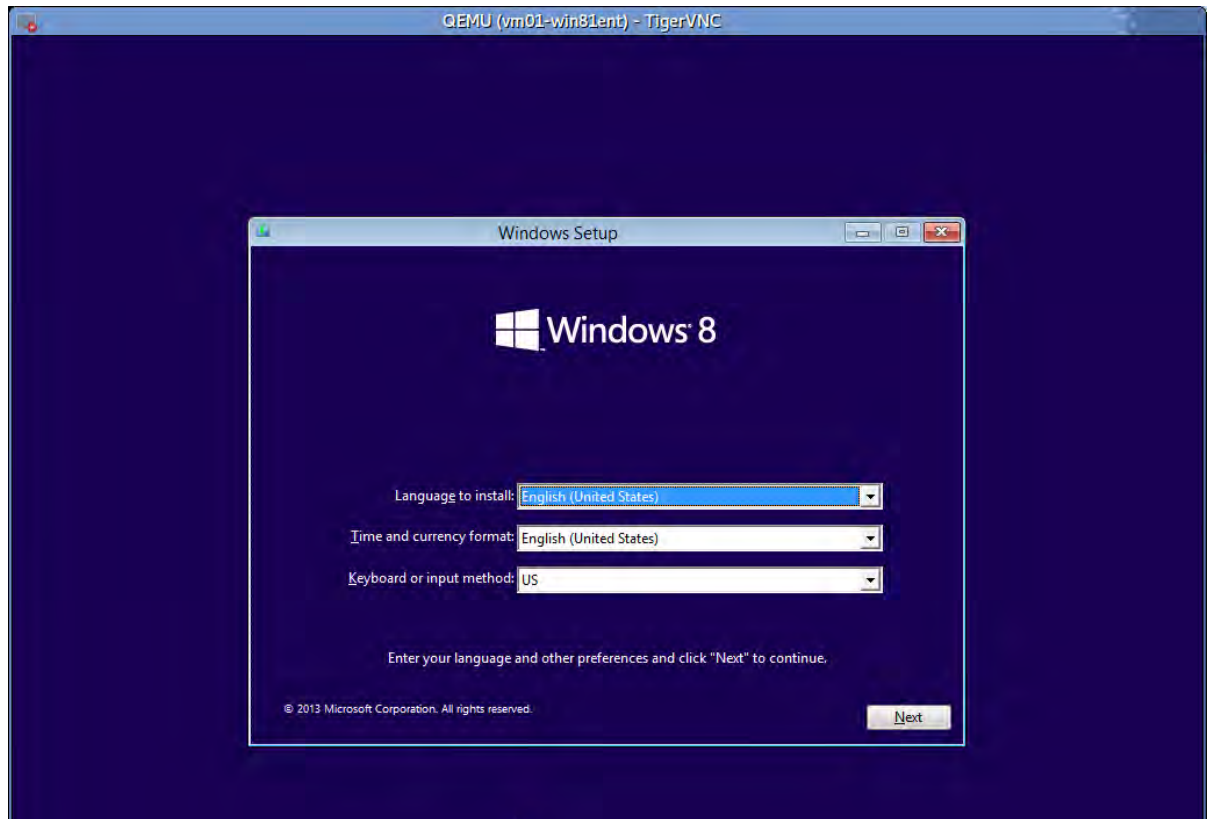
5.8.3. Запуск ВМ и установка ОС

- 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 СБТ (в примере ниже подключение производится из АРМ с Linux, адрес ЭВМ с установленным гипервизором - «192.168.0.184»):

```
$ vncviewer 192.168.0.184:1
```

3) после некоторого ожидания, на экране появится следующее изображение, (рис. 3). С другой версией ОС изображение будет другим;

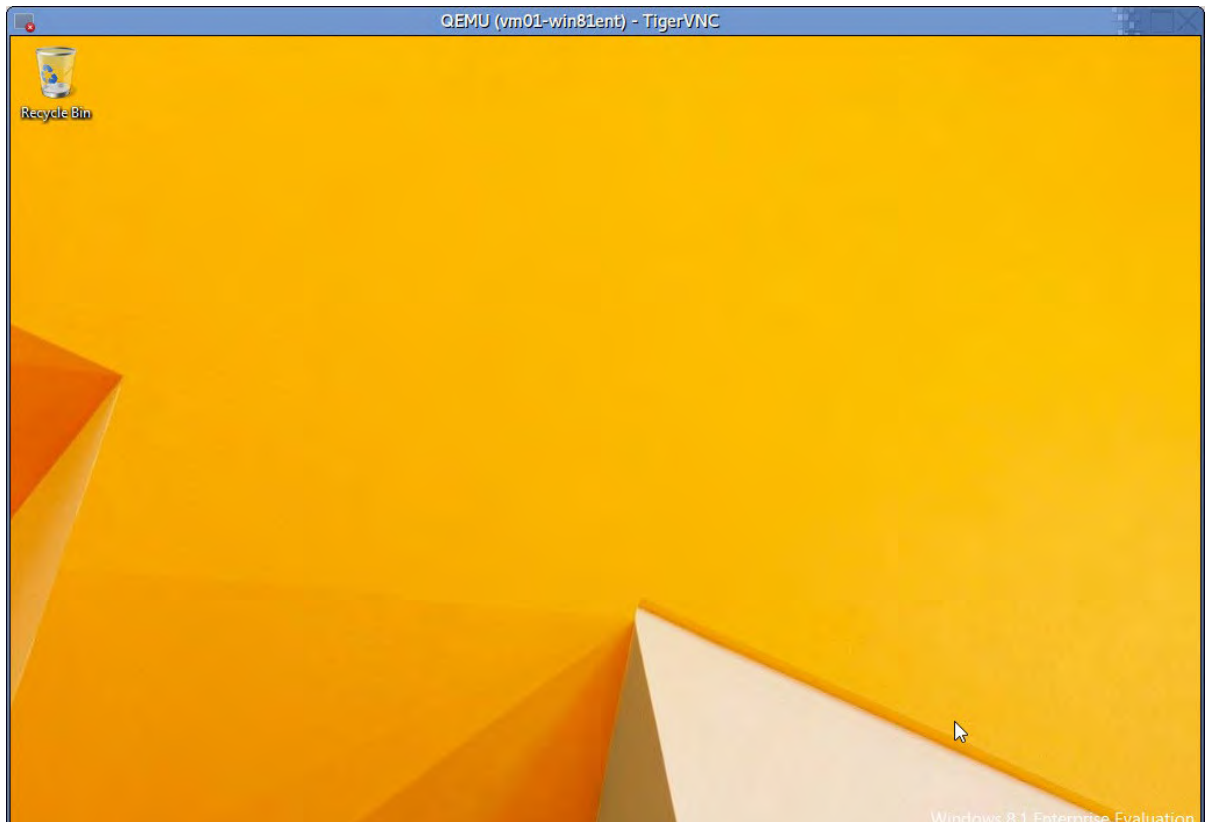
Установка гостевой ОС (Win 8.1 Enterprise)



4) 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 заново подключиться при помощи команды `vncviewer`;

5) после установки системы, при входе через VNC, получаем следующее изображение, (рис. 4). С другой версией ОС результат может быть иным.

Работа гостевой ОС (Win 8.1 Enterprise)



5.8.4. Установка VM с ОС Linux (Ubuntu)

Для установки VM с ОС Ubuntu Linux необходимо выполнить следующие действия:

1) создать конфигурационный файл VM «/home/vm02-ubuntu.cfg» со следующим содержанием:

```
builder = "hvm" memory = 16384 name = "vm02-ubuntu" vcpus = 8 mmio_hole = 3072
machine = "q35" vif = [ # Только виртуальный адаптер.
Требуется 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/ISO/ubuntu-mate-16.04.3-desktop-
amd64.iso,hdc:cdrom,r" , ] # Первым грузим CDRом - инсталляция
boot=' cd'

usb = 1

# VNC options
vnc = 1
# Первый экран - соответствует номеру VM.
```

```
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:

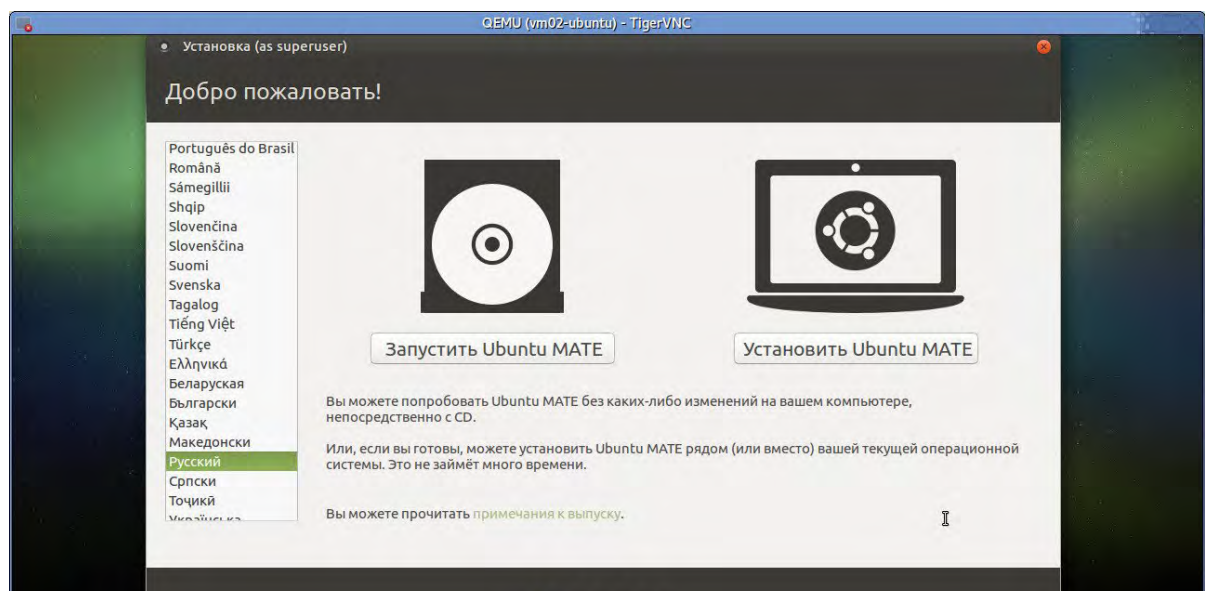
```
root@dom0image:~# xl create /home/vm02-ubuntu.cfg Parsing config from /home/vm02-ubuntu.cfg
```

3) после запуска виртуальной машины, примерно через 3-5 секунд необходимо подключиться к ней по протоколу VNC, используя IP адрес установленного СБТ (в примере ниже подключение производится из АРМ с Linux, адрес ЭВМ с установленным гипервизором - «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). С другой версией ОС результат может быть иным;

Установка гостевой ОС (Ubuntu Linux 16.04)



5) следуя инструкциям инсталлятора, установить ОС.

После перезагрузки гостевой ОС, подключения через VNC и входа в систему, можно увидеть следующее изображение (рис. 6).

Современные ОС семейства Linux не требуют дополнительной специальной настройки для работы с гипервизором.

Работа гостевой ОС (Ubuntu Linux 16.04)



5.8.5. Установка паравиртуальных драйверов устройств

Для наиболее эффективной работы гостевой системы рекомендуется установить паравиртуальные драйверы устройств ввода-вывода, предоставляемые гипервизор. Паравиртуальные драйверы позволяют обращаться к устройствам напрямую, используя гипервызовы, и минуя эмуляцию, реализуемую при помощи QEMU. гипервизор 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 гипервизором;
- 3) в конфигурационном файле «/home/vm01-win81ent.cfg» 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* — созданный образ ISO с драйверами;

- 4) выключить ВМ средствами гостевой ОС (в данный момент ОС еще не может принимать команды от Гипервизора, выключение доступно только через ACPI);
- 5) запустить ВМ, убедиться, что CDROM-диск с драйверами подключен;
- 6) в зависимости от ОС, установить паравиртуальные драйверы.

5.9. Примеры настройки драйвер-доменов

Ниже приводится описание примеров настроек, присутствующих в гипервизоре.

5.9.1. Настройка образа драйвер-домена

Для настройки дискового драйвер-домена необходимо создать виртуальную машину, как описано в подразделе 144, с учетом указания другого имени ВМ, например, «drvdom», выполнив стандартные этапы создания домена на основе *dom0image*:



вход в систему под учетной записью администратора (root):

```
Hypervisor Dom0 image dom0image login: root Password: root@dom0image:~#
```



включение режима SELinux «enforcing»:

```
root@dom0image:~# setenforce 0
```



создание тома **LVM** для драйвер-домена:

```
root@dom0image:~# lvcreate -n drvdom -l 256 vg0 Logical volume "drvdom" created.
root@dom0image:~# lvscan | grep drvdom ACTIVE                "/dev/vg0/drvdom" [1,00
GiB] inherit
```



разбиение диска на разделы

```
root@dom0image:~# parted -s /dev/mapper/vg0-drvdom mklabel msdos root@dom0image:~#
parted -s /dev/mapper/vg0-drvdom mkpart p ext2 1MiB 128MiB root@dom0image:~# parted -
s /dev/mapper/vg0-drvdom mkpart p ext2 128MiB 100% root@dom0image:~# 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 1 1049kB
134MB 133MB primary 2 134MB 1074MB 940MB primary
```



создание файловой системы для загрузчика в первом разделе:

```
root@dom0image:~# 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
```



скопировать образ **dom0** (/boot/dom0image) во второй раздел и изменить размер ФС:

```

root@dom0image:~# lzcat /boot/dom0image | 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@dom0image:~# 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 dom0image: ***** FILE SYSTEM WAS MODIFIED ***** dom0image: 8676/32768
files (0.1% non-contiguous), 90308/131072 blocks root@dom0image:~# 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.

```



создание точки монтирования и монтирование раздела драйвер-домена:

```

root@dom0image:~# mkdir -p /mnt/drvidom root@dom0image:~# mount /dev/mapper/vg0-
drvidom2 /mnt/drvidom

```



создание файла /etc/fstab:

```

root@dom0image:~# cat >/mnt/drvidom/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

```



создание файла /etc/inittab:

```

root@dom0image:~# cat > /mnt/drvidom/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

```



запрет запуска ряда стартовых сценариев:

```

root@dom0image:~# mkdir -p /mnt/etc/init.d/disabled root@dom0image:~# mv -v
/mnt/drvidom/etc/init.d/{ S07*, S10auditd, S30*, S4*, S50*, S51*, S7*}
/mnt/drvidom/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/S70libvirt' ->
"/mnt/etc/init.d/disabled/S70libvirt" ' /mnt/etc/init.d/S71virtlogd' ->
"/mnt/etc/init.d/disabled/S71virtlogd" ' /mnt/etc/init.d/S72libvirt-guests' ->
"/mnt/etc/init.d/disabled/S72libvirt-guests"

```



задание имени хоста:

```

root@dom0image:~# echo "drvidom" >/mnt/drvidom/etc/hostname root@dom0image:~# cat
/mnt/drvidom/etc/hosts 127.0.0.1 drvidom

```



размонтировать /mnt/drvidom:

```

root@dom0image:~# umount /mnt/drvidom

```



создать конфигурационный файл для драйвер-домена:

```

root@dom0image:~# cat > /home/drvidom.cfg #builder=' hvm' memory = 1024 vcpus = 1
name = "drvidom" kernel = "/boot/dom0kernel" cmdline = "debug earlyprintk=xen
console=hvc0 rootfstype=ext4 selinux=0 root=/dev/xvda2" disk =
[ "phy:/dev/vg0/drvidom,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
```

 запустить виртуальную машину **testvm01** в паравиртуальном режиме (строка *builder="hvm"* закомментирована, используется прямая загрузка ядра */boot/dom0kernel*):

```
root@dom0image:~# 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="dom0image" UUID="053c9cec-afdf-473a-b559-146fe30870da" TYPE="ext4"
PARTUUID="ffaca1bf-02"
```

 запретить запуск чего либо в **rc.local**:

```
root@drvdom:~# printf "#!/bin/bash¥ nexit 0' >/etc/rc.local
```

 добавить строку для «/boot»:

```
root@drvdom:~# echo "/dev/xvda1 /boot ext4 defaults,noatime,nodiratime,discard 0
1" >>/etc/fstab
```

 примонтировать «/boot» и установить **GRUB**:

```
root@drvdom:~# mount /boot root@drvdom:~# grub-install /dev/xvda
Выполняется установка для платформы i386-пс.
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 /dom0kernel console=hvc0 root=/dev/xvda2 rootfstype=ext4 selinux=0
nomodeset }
```

 выключить 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
```



скопировать ядро **dom0**:

```
root@dom0image:~# mount /dev/mapper/vg0-drvdom1 /mnt/drvdom/ root@dom0image:~# cp -va
/boot/dom0kernel /mnt/drvdom/ ' /boot/dom0kernel' -> "/mnt/drvdom/dom0kernel"
root@ws00:~# umount /mnt/drvdom
```



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/dom0kernel"
#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.

5.9.2. 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:



создать блочное устройство для диска драйвер-домена:

```
root@ws00:~# lvcreate -n sasdom -L 1GiB vg0 Logical volume "sasdom" created.
```



скопировать содержимое созданного ранее шаблона драйвер-домена:

```
root@ws00:~# dd if=/dev/mapper/vg0-sasdom of=/dev/mapper/vg0-usbdom bs=1M
```



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
```



задать имя хоста:

```
boot@ws00:~# echo "sasdom" >/mnt/sasdom/etc/hostname boot@ws00:~# echo "127.0.0.1
sasdom localhost" >/mnt/sasdom/etc/hosts
```



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' ]
```



запустить дисковый драйвер-домен и проверить наличие дисков, подключенных к контроллеру (в данном случае это два 1 ТБ 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=/dom0kernel 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
Password:
root@sasdom:~# lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda          8:0    0 931,5G  0 disk
sdb          8:16    0 931,5G  0 disk
xvda        202:0    0     2G  0 disk
├─xvda1 202:1    0   511M  0 part /boot
└─xvda2 202:2    0    1,5G  0 part /

```



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

```



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=/dom0kernel 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 ... 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:
done

Init LVM volumes, if any:
  1 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:~#

```

 завершить сеанс работы в консоли (комбинация клавиш «Ctrl+J») и проверить доступность блочного устройства для остальных доменов (проверить блочные устройства **Dom0**, затем подключить том из «sasdom», проверить его доступность, и отключить):

```

root@ws00:~# lsblk NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT sda
8:0      0 223,6G  0 disk  └─sda1                8:1      0   15M  0 part  └─sda2
8:2      0  1008M  0 part  /boot └─sda3              8:3      0    8G  0 part  /var

```

92386160.02001-01 90 01

```

└─sda4                8:4    0 214,6G  0 part /home sr0                11:0    1
1024M  0 rom nvme0n1    259:0    0 477G  0 disk └─nvme0n1p1    259:1
0 476G  0 part └─vg0-vm01    254:0    0 32G  0 lvm └─vg0-sasdom    254:1
0 2G  0 lvm | └─vg0-sasdom1 254:3    0 511M  0 part | └─vg0-sasdom2 254:5
0 1,5G  0 part └─vg0-usbdom 254:2    0 2G  0 lvm └─vg0-usbdom1 254:4
0 511M  0 part └─vg0-usbdom2 254:6    0 1,5G  0 part root@ws00:~# xl block-
attach 0

```

```

"format=raw,backendtype=phy,backend=sasdom,vdev=xvda,target=/dev/vg.sas/lvol00'

```

```

root@ws00:~# lsblk NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT sda
8:0    0 223,6G  0 disk └─sda1                8:1    0 15M  0 part └─sda2
8:2    0 1008M  0 part /boot └─sda3                8:3    0 8G  0 part /var
└─sda4                8:4    0 214,6G  0 part /home sr0                11:0    1
1024M  0 rom xvda      202:0    0 64G  0 disk nvme0n1    259:0    0
477G  0 disk
└─nvme0n1p1    259:1    0 476G  0 part
  └─vg0-vm01    254:0    0 32G  0 lvm
  └─vg0-sasdom  254:1    0 2G  0 lvm
  | └─vg0-sasdom1 254:3    0 511M  0 part
  | └─vg0-sasdom2 254:5    0 1,5G  0 part
  └─vg0-usbdom  254:2    0 2G  0 lvm
    └─vg0-usbdom1 254:4    0 511M  0 part
    └─vg0-usbdom2 254:6    0 1,5G  0 part

```

```

root@ws00:~#

```

```

root@ws00:~# xl block-list 0

```

```

Vdev BE handle state evt-ch ring-ref BE-path

```

```

51712 6 0 4 78 8 /local/domain/6/backend/vbd/0/51712

```

```

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

```

```

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 223,6G 0 disk
└─sda1 8:1 0 15M 0 part
└─sda2 8:2 0 1008M 0 part /boot
└─sda3 8:3 0 8G 0 part /var
└─sda4 8:4 0 214,6G 0 part /home
sr0 11:0 1 1024M 0 rom
nvme0n1 259:0 0 477G 0 disk
└─nvme0n1p1 259:1 0 476G 0 part
  └─vg0-vm01 254:0 0 32G 0 lvm
  └─vg0-sasdom 254:1 0 2G 0 lvm
  | └─vg0-sasdom1 254:3 0 511M 0 part
  | └─vg0-sasdom2 254:5 0 1,5G 0 part
  └─vg0-usbdom 254:2 0 2G 0 lvm

```

—vg0-usbdom1	254:4	0	511M	0 part
—vg0-usbdom2	254:6	0	1,5G	0 part

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).

6. БЕЗОПАСНАЯ НАСТРОЙКА ТИПОВЫХ КОНФИГУРАЦИЙ

Разделы ниже приводят описание процедуры настройки типовых конфигураций, соответствующих этим вариантам, но не ограничивающих использование гипервизора только приведенными конфигурациями.

6.1. Типовая конфигурация 1 - автоматический сервер ВМ

Данная конфигурация обеспечивает вариант применения гипервизор, 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:

- 1) подготовить менеджер томов для создания образов виртуальных машин в соответствии с подразделом 158;
- 2) создать сетевые мосты для требуемых сегментов сетей с различными уровнями конфиденциальности, согласно подразделу 119 и пункту 162;
- 3) произвести установку одной или нескольких гостевых ОС в соответствии с подразделом 165;
- 4) согласно пункту 165, настроить автоматический запуск виртуальных машин.

При корректной настройке системы, виртуальные машины будут автоматически запущены при старте СВТ.

6.2. 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 физ. 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 158;
-  Создать сетевые мосты для требуемого сегмента сетей с соответствующим уровнем конфиденци, см. подраздел 119 и пункт 162;
-  Произвести установку гостевой ОС в соответствии с подразделом 165;
-  Определить устройство (GPU), для которого требуется прямой доступ из ВМ, и добавить соответствующие параметры в конфигурационный файл ВМ, согласно подразделу 124, 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 пункту 165, настроить автоматический запуск виртуальных машин.

При корректной настройке системы, виртуальные машины с прямым доступом к оборудованию будут автоматически запущены при старте СВТ.

6.3. 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 158;

-  создать сетевые мосты для требуемых сегментов сетей с различными уровнями конфиденциальности, см. подраздел 119 и пункт 162;

-  произвести установку одной или нескольких гостевых ОС в соответствии с подразделом 165;

-  завести соответствующую учетную запись для пользователя — разработчика;

-  проинициализировать отчуждаемые устройства, настроить соответствующие записи в /etc/fstab для сопоставления пользователя с устройством.

7. НАСТРОЙКА ОРГАНИЗАЦИИ ДОСТУПА К ВМ С ПОМОЩЬЮ ИНТЕРФЕЙСА ГИПЕРВИЗОРА

Настройка производится в соответствии с разделом 3.4.8 Руководства оператора 92386160.02001-01 34 01.

ПЕРЕЧЕНЬ ТЕРМИНОВ

ACPI	— Advanced Configuration and Power Interface - усовершенствованный интерфейс управления конфигурацией и питанием — открытый промышленный стандарт, впервые выпущенный в декабре 1996 года и разработанный совместно компаниями HP, Intel, Microsoft, Phoenix и Toshiba, который определяет общий интерфейс для обнаружения аппаратного обеспечения, управления питанием и конфигурации материнской платы и устройств
AHCI	— Advanced Host Controller Interface — механизм, используемый для подключения накопителей информации по протоколу Serial ATA, позволяющий пользоваться расширенными функциями, такими, как встроенная очередность команд (NCQ) и горячая замена
AMD_Vi	— технология AMD для виртуализации IOMMU
AMD-V	— технология виртуализации AMD, часто обозначается аббревиатурой SVM (Secure Virtual Machines) на платформе x86, ранее известная под кодовым названием «Pacifica»
API	— Application Programming Interface — программный интерфейс доступа
BIOS	— Basic Input/Output System - Базовая система ввода-вывода
Dom0	— первый запускаемый домен (виртуальная машина), который автоматически создается 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
EPT	— Extended Page Table - технология виртуализации Intel для страничных таблиц, сходная с AMD RVI. Обеспечивает управление теневыми страницами и оптимизацию производительности при работе с PTE
EPTE	— EPT плюс PTE
EXT4	— журналируемая файловая система, обычно используемая в ОС с ядром Linux
GPU	— Graphic Processing Unit - устройство обработки графической информации. Может работать как с выводом результата на монитор, так и без.

		Также может использоваться для выполнения параллельных вычислений, существенно увеличивая вычислительную мощность СБТ, такая разновидность называется GPGPU - General Purpose Graphic Processing Unit
HVM	—	Hardware Virtual Machine - виртуальная машина, работающая в режиме полной виртуализации
Intel VT	—	технология виртуализации Intel на платформе x86, ранее известная под кодовым названием «Vanderpool»
Intel VT-d	—	технология Intel для виртуализации IOMMU
IOMMU	—	Input Output Memory Management Unit — Устройство управления памятью ввода-вывода, работающее аналогично MMU центрального процессора, используя таблицы страниц (PT) и специальную таблицу отображения DMA (DMA remapping table — DMAR), которую гипервизор получает от BIOS через ACPI
MMU	—	Memory Management Unit — блок управления памятью или устройство управления памятью — компонент аппаратного обеспечения компьютера, отвечающий за управление доступом к памяти, запрашиваемым центральным процессором
Multi-OS GPU	—	Multi OS GPU - технология NVidia, позволяющая использовать профессиональные GPU в среде виртуализации. Также, в соответствии с маркетинговой политикой NVidia, драйвер не будет работать не с MultiOS GPU, если обнаружена виртуализация. Фактически, можно использовать и GPU без такой технологии (с аппаратной модификацией PCI ID), но при этом могут возникать проблемы с переинициализацией карты и некорректной работой драйвера, связанной с несоответствием PCI ID фактическому оборудованию
MultiUser GPU	—	MultiUser GPU - технология AMD/ATI, позволяющая использовать GPU в среде виртуализации и выполнять аппаратное разбиение одной видеокарты на несколько «виртуальных» vGPU
PCI	—	Peripheral Component Interconnect - шина передачи данных для подключения периферийных устройств
PCI Express	—	Peripheral Component Interconnect Express —

		шина передачи данных для подключения периферийных устройств, существенно усовершенствованная и расширенная, по сравнению с PCI
PTE	—	Page Table Entry - элемент таблицы страниц - данные, хранящиеся в структурах таблиц страниц, используются при работе MMU
PVH	—	режим виртуализации с аппаратной поддержкой, аналогичный режиму полной виртуализации, но без эмуляции аппаратной платформы
QEMU	—	программа, обеспечивающая эмуляцию заданной аппаратной платформы для виртуальной машины. В состав эмулируемой платформы входят: контроллер шины PCI (при эмуляции чипсетов i440FX и Q35) и шины PCIe (при эмуляции чипсета Q35), дискового контроллера, видеоадаптера, и т.д.
RFC	—	Request For Comment - запрос на комментарии (историческое название) - класс документов, описывающих стандарты и протоколы в области информационных технологий. Де факто являются основным источником информации и стандартизации протоколов и механизмов. Разрабатываются сообществом и научными организациями, такими как IEEE, ACM, NSI, и т.д.
RVI	—	Rapid Virtualization Indexing — технология виртуализации для страничных таблиц (ранее известная как Nested Page Tables) AMD. В дальнейшем адаптирована Intel как EPT
SPICE	—	Simple Protocol for Independent Computing Environment - вариант механизма удаленного доступа к графическому экрану пользователя, сходный с VNC, но имеющий возможность дополнительной передачи аудио-потокa и перенаправления доступа к USB устройствам
UEFI	—	Unified Extensible Firmware Interface — современный вариант подсистемы инициализации и ввода-вывода CBT, пришедший на смену BIOS
VMCS	—	Virtual machine control structure — структура управления виртуальной машины. VMCS — структура данных в памяти, существующая в точности в одном экземпляре на одну виртуальную машину и управляемая

	гипервизором. С каждым изменением контекста выполнения между разными ВМ структура данных VMCS восстанавливается для текущей виртуальной машины, определяя состояние виртуального процессора ВМ
VNC	— Virtual Network Computing - X11 сервер, клиент и протокол для удаленного доступа к графическому экрану пользователя. Обычно реализуется в виде независимого X11 сервера с виртуальным фреймбуфером, содержимое которого (изображение) передается по соответствующему протоколу на клиент, а с клиента передаются события ввода - перемещения манипулятора «мышь», нажатия кнопок клавиатуры, и т. д.
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
ВМ, виртуальная машина	— программная система, эмулирующая аппаратное обеспечение целевой платформы и исполняющая программы для целевой платформы на хост-платформе
Гипервизор	— основной компонент функционального модуля гипервизор, обеспечивающий одновременное, параллельное выполнение нескольких виртуальных машин на одном и том же компьютере (хосте)
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, доступным по сети (сетевые блочные устройства iSCSI, SRP и т.д.) и предоставление его ресурсов посредством back end драйвера
Паравиртуализация	— техника виртуализации, при которой гостевые операционные системы подготавливаются для исполнения в виртуализированной среде, для чего их ядро незначительно модифицируется

Полная виртуализация	— технология, используемая для предоставления определенной виртуальной среды, которая обеспечивает полное симулирование базового оборудования. Использует аппаратную поддержку механизмов виртуализации, предоставляемую ЦП, и эмуляцию аппаратной платформы (чипсет, шина PCI/PCI Express, и т. д.), в программе «V-Софт» предоставляемую QEMU
Хост	— устройство, предоставляющее сервисы формата «клиент-сервер» в режиме сервера по каким-либо интерфейсам и уникально определенное на этих интерфейсах. В контексте программы «V-Софт»- сервер или АРМ, на котором работает Гипервизор, и ресурсы которого используются для выполнения ВМ
Эмуляция	— комплекс программных, аппаратных средств или их сочетание, предназначенное для копирования (или эмулирования) функций одной вычислительной системы (гостя) на другой, отличной от первой, вычислительной системе (хосте) таким образом, чтобы эмулированное поведение как можно ближе соответствовало поведению оригинальной системы (гостя)

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АРМ	—	автоматизированное рабочее место
АС	—	автоматизированная система
АСЗИ	—	автоматизированная система в защищенном исполнении
ИС	—	информационная система
КСЗ	—	комплекс средств защиты
НСД	—	несанкционированный доступ
ОС	—	операционная система - разновидность программного обеспечения, обеспечивающая среду исполнения для прикладного программного обеспечения
ППО	—	прикладное программное обеспечение — программы, выполняющие необходимые конечному пользователю задачи, работающие под управлением специального программного обеспечения
ПРД	—	правила разграничения доступа, политика безопасности
СВТ	—	средства вычислительной техники
СУБД	—	система управления базой данных
ТЗ	—	техническое задание
СУ	—	средства управления
СВ	—	среда виртуализации
ФС	—	файловая система - способ организации данных на устройстве хранения в виде файлов.
ФСТЭК	—	федеральная служба по техническому и экспортному контролю
ЦП	—	центральный процессор

[illegible]