1 Ravada delivers

2 Who is Ravada meant for?

3 Ravada VDI documentation
   3.1 Install Ravada
   3.2 Install Ravada on Fedora
   3.3 Install Ravada - Ubuntu Xenial
   3.4 Running Ravada in production
   3.5 Post Install Recommendations
   3.6 Development release
   3.7 Ubuntu installation
   3.8 Add KVM storage pool
   3.9 Configure Hypnotoad proxy
   3.10 Install Apache
   3.11 Enable apache modules
   3.12 Apache Proxy Configuration
   3.13 Apache redirect to https
   3.14 How to import a Virtualbox image
   3.15 How to create a Virtual Machine
   3.16 How to dump a hard drive to Ravada
   3.17 How to Install a local LDAP
   3.18 How to add a KVM template
   3.19 New ISO image
   3.20 How to import an OpenGnsys image
   3.21 Integrating Ravada and OpenGnsys
   3.22 Operation
   3.23 Swap Partition
   3.24 Troubleshooting frequent problems
   3.25 Upgrade Ravada
   3.26 Windows SPICE Clients
   3.27 How to change the controller driver of a Windows VM to VirtIO
   3.28 Virtual Machine Manual Migration
   3.29 Kiosk Mode
   3.30 How to do a volatile Virtual Machine
   3.31 Adding Custom Messages
   3.32 Create a custom login template
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.33</td>
<td>Create a custom footer template</td>
</tr>
<tr>
<td>3.34</td>
<td>Disable Spice Password</td>
</tr>
<tr>
<td>3.35</td>
<td>Ravada advanced settings</td>
</tr>
<tr>
<td>3.36</td>
<td>New documentation</td>
</tr>
<tr>
<td>3.37</td>
<td>Configure LDAP Authentication</td>
</tr>
<tr>
<td>3.38</td>
<td>Authentication with Active Directory</td>
</tr>
<tr>
<td>3.39</td>
<td>Set access restrictions to Virtual Machines</td>
</tr>
<tr>
<td>3.40</td>
<td>Test Active Directory</td>
</tr>
<tr>
<td>3.41</td>
<td>Tuning Ravada</td>
</tr>
<tr>
<td>3.42</td>
<td>Server Monitoring</td>
</tr>
<tr>
<td>3.43</td>
<td>Guide button step-by-step</td>
</tr>
<tr>
<td>3.44</td>
<td>Local JS and CSS files instead CDN</td>
</tr>
<tr>
<td>3.45</td>
<td>Exposing a Virtual Machine</td>
</tr>
<tr>
<td>3.46</td>
<td>Exposing Ports from a Virtual Machine</td>
</tr>
<tr>
<td>3.47</td>
<td>Install Alpine Linux</td>
</tr>
<tr>
<td>3.48</td>
<td>Install Windows 10</td>
</tr>
<tr>
<td>3.49</td>
<td>How to extend a Ravada Windows guest’s disk space</td>
</tr>
<tr>
<td>3.50</td>
<td>How to extend a Ravada Linux guest’s disk space</td>
</tr>
<tr>
<td>3.51</td>
<td>How to enable KVM virsh console access</td>
</tr>
<tr>
<td>3.52</td>
<td>Reduce the image size after cloning a physical PC</td>
</tr>
<tr>
<td>3.53</td>
<td>Qemu Guest Agent</td>
</tr>
<tr>
<td>3.54</td>
<td>Set Hostname</td>
</tr>
<tr>
<td>3.55</td>
<td>Development Tools</td>
</tr>
<tr>
<td>3.56</td>
<td>Commit Rules</td>
</tr>
<tr>
<td>3.57</td>
<td>Database Changes</td>
</tr>
<tr>
<td>3.58</td>
<td>Editor configuration rules</td>
</tr>
<tr>
<td>3.59</td>
<td>Local ISO server</td>
</tr>
<tr>
<td>3.60</td>
<td>Steps to release</td>
</tr>
<tr>
<td>3.61</td>
<td>Run Ravada in development mode</td>
</tr>
<tr>
<td>3.62</td>
<td>Testing environment</td>
</tr>
<tr>
<td>3.63</td>
<td>How to create tests</td>
</tr>
<tr>
<td>3.64</td>
<td>Translations</td>
</tr>
<tr>
<td>3.65</td>
<td>Ravada Documentation</td>
</tr>
<tr>
<td>3.66</td>
<td>Hardening Spice security with TLS</td>
</tr>
</tbody>
</table>
The chances are you’re here because you’ve been searching for free Virtual Desktop Infrastructure (VDI) documentation. Whether it is a large or a small project, you can start with VDI and see its benefits right away! We assume you do want to start your VDI project as quickly as possible. Therefore, RAVADA VDI is the perfect software for you!

Ravada VDI is a free and open-source project that allows users to connect to a virtual desktop. So it is a VDI broker.
Ravada delivers

By following the documentation and editing some configuration files, you’ll be able to deploy a VM within minutes.
Who is Ravada meant for?

Ravada is meant for sysadmins who have some background in GNU/Linux, and want to deploy a VDI project.

Note: Get started with VDI, without reinventing the wheel.

We have written some documentation and hosted it on Read the Docs for you. This documentation is on-going, so if there is something you think is missing, don’t hesitate and drop us a line! In the meantime, we are still improving RAVADA VDI and its documentation, so new sections will be popping out from time to time.

Our code uses the AGPL license and it is available on GitHub.
The main documentation for the site is divided into three main sections:

- user-docs
- Feature Documentation
- Guest VM section

Do you feel like giving us a hand? Here you have all the information you need as a developer:

- Developer Documentation

### 3.1 Install Ravada

#### 3.1.1 Requirements

**3.1.2 OS**

Ravada works in any Linux distribution but we only support the package for Ubuntu server and Fedora server. Follow this guide if you prefer Debian Jessie.

#### 3.1.3 Hardware

It depends on the number and type of virtual machines. For common scenarios are server memory, storage and network bandwidth the most critical requirements.

**Memory**

RAM is the main issue. Multiply the number of concurrent workstations by the amount of memory each one requires and that is the total RAM the server must have.
Disks

The faster the disks, the better. Ravada uses incremental files for the disks images, so clones won’t require many space.

3.1.4 Install Ravada

Follow this guide if you are only upgrading Ravada from a previous version already installed.

3.1.5 Ubuntu

Note: We only provide support for Ubuntu 18.04 LTS (bionic).

We provide deb Ubuntu packages. Download it from the UPC ETSETB repository.
Install libmojolicious-plugin-renderfile-perl package:

```bash
$ sudo apt-get install libmojolicious-plugin-renderfile-perl
```

Then install the ravada package. Choose the one that matches your OS release:

- ravada_0.5.1_ubuntu-18.04_all.deb
- ravada_0.5.1_ubuntu-18.10_all.deb
- ravada_0.5.1_ubuntu-19.04_all.deb
- ravada_0.5.1_debian-10_all.deb

When you run dpkg now it may show some errors, it is ok, keep reading.

```bash
$ wget http://infoteleco.upc.edu/img/debian/ravada_0.5.1_ubuntu-18.04_all.deb
$ sudo dpkg -i ravada_0.5.1_ubuntu-18.04_all.deb
```

The last command will show a warning about missing dependencies. Install them running:

```bash
$ sudo apt-get update
$ sudo apt-get -f install
```

Debian Release

If you are using the Debian release you must enable spice KVM manually:

```bash
$ sudo ln -s /usr/bin/kvm /usr/bin/kvm-spice
```

3.1.6 Mysql Database

MySQL server

Warning: MySql required minimum version 5.6

It is required a MySQL server, it can be installed in another host or in the same one as the ravada package.

```bash
$ sudo apt-get install mysql-server
```

After completion of mysql installation, run command:
$ sudo mysql_secure_installation

**MySQL database and user**

It is required a database for internal use. In this examples we call it *ravada*. We also need an user and a password to connect to the database. It is customary to call it *rvd_user*. In this stage the system wants you to set a password for the sql connection.

**Warning:** When installing MySQL you won't be asked for a password, you can set a password for the root user in MySQL via *mysql_secure_installation* or type your user’s password when it ask’s you for a password.

Create the database:

$ sudo mysqladmin -u root -p create ravada

Grant all permissions on this database to the *rvd_user*:

$ sudo mysql -u root -p ravada -e "create user 'rvd_user'@'localhost' identified by 'Pword12345*';"

$ sudo mysql -u root -p ravada -e "grant all on ravada.* to rvd_user@'localhost';"

The password chosen must fulfill the following characteristics:

- At least 8 characters.
- At least 1 number.
- At least 1 special character.

**Config file**

Create a config file at /etc/ravada.conf with the username and password you just declared at the previous step. Please note that you need to edit the user and password via an editor. Here, we present Vi as an example.

```
sudo vi /etc/ravada.conf
```

```
db:
    user: rvd_user
    password: Pword12345*
```

**3.1.7 Ravada web user**

Add a new user for the ravada web. Use *rvd_back* to create it. It will perform some initialization duties in the database the very first time this script is executed.

When asked if this user is admin answer *yes*.

$ sudo /usr/sbin/rvd_back --add-user user.name

**3.1.8 Client**

The client must have a spice viewer such as virt-viewer. There is a package for linux and it can also be downloaded for windows.
3.1.9 Run

The Ravada server is now installed, learn how to run and use it.

3.1.10 Help

Struggling with the installation procedure? We tried to make it easy but let us know if you need assistance. There is also a troubleshooting page with common problems that admins may face.

3.2 Install Ravada on Fedora

3.2.1 Requirements

3.2.2 OS

Ravada works in any Linux distribution.

3.2.3 Hardware

It depends on the number and type of virtual machines. For common scenarios are server memory, storage and network bandwidth the most critical requirements.

Memory

RAM is the main issue. Multiply the number of concurrent workstations by the amount of memory each one requires and that is the total RAM the server must have.

Disks

The faster the disks, the better. Ravada uses incremental files for the disks images, so clones won’t require many space.

3.2.4 Install Ravada

Follow this guide if you are only upgrading Ravada from a previous version already installed.

3.2.5 Fedora and EPEL7

You can install ravada using the ‘dnf’ package manager.

$ sudo dnf install ravada

Add link to kvm-spice

This may change in the future but actually a link to kvm-spice is required. Create it this way:

$ ln -s /usr/bin/qemu-kvm /usr/bin/kvm-spice
MySQL server

It is required a MySQL server, in Fedora we use MariaDB server. It can be installed in another host or in the same as the ravada package.

$ sudo dnf install mariadb mariadb-server

And don’t forget to enable and start the server process:

$ sudo systemctl enable --now mariadb.service
$ sudo systemctl start mariadb.service

MySQL database and user

It is required a database for internal use. In this examples we call it ravada. We also need an user and a password to connect to the database. It is customary to call it rvd_user. In this stage the system wants you to set a password for the sql connection.

Warning: If installing ravada on Ubuntu 18 or newer you should enter your user’s password instead of mysql’s root password.

Create the database:

$ sudo mysqladmin -u root -p create ravada

Grant all permissions on this database to the rvd_user:

$ sudo mysql -u root -p ravada -e "grant all on ravada.* to rvd_user@'localhost' identified by 'Pword12345*"

The password chosen must fulfill the following characteristics:

- At least 8 characters.
- At least 1 number.
- At least 1 special character.

Config file

Create a config file at /etc/ravada.conf with the username and password you just declared at the previous step. Please note that you need to edit the user and password via an editor. Here, we present Vi as an example.

```
sudo vi /etc/ravada.conf
db:
  user: rvd_user
  password: Pword12345*
```

3.2.6 Ravada web user

Add a new user for the ravada web. Use rvd_back to create it. It will perform some initialization duties in the database the very first time this script is executed.

When asked if this user is admin answer yes.

$ sudo /usr/sbin/rvd_back --add-user user.name
3.2.7 Firewall (Optional)

The server must be able to send DHCP packets to its own virtual interface.

KVM should be using a virtual interface for the NAT domains. Look what is the address range and add it to your iptables configuration.

First we try to find out what is the new internal network:
```
sudo route -n
...
192.168.122.0   0.0.0.0       255.255.255.0  U     0      0      0 virbr0
```

So it is 192.168.122.0, netmask 24. Add it to your iptables configuration:
```
$ sudo iptables -A INPUT -s 192.168.122.0/24 -p udp --dport 67:68 --sport 67:68 -j ACCEPT
```

To confirm that the configuration was updated, check it with:
```
$ sudo iptables -S
```

3.2.8 Client

The client must have a spice viewer such as virt-viewer. There is a package for linux and it can also be downloaded for windows.

3.2.9 Run

The Ravada server is now installed, learn how to run and use it.

3.2.10 Help

Struggling with the installation procedure? We tried to make it easy but let us know if you need assistance.

There is also a troubleshooting page with common problems that admins may face.

3.3 Install Ravada - Ubuntu Xenial

It is advisable to install Ravada in one of the supported platforms: Ubuntu Bionic (18.04) or Fedora. But if you want to install in another distribution it can be done.

3.3.1 Packages

Install those packages:
```
$ sudo apt-get install perl libmojolicious-perl mysql-common libauthen-passphrase-perl libdbd-mysql perl
```

In addition you need one package that it still may not be in Ubuntu repository, download from our own server at the UPC ETSETB repository and install it this way:
```
$ wget http://infoteleco.upc.edu/img/debian/libmojolicious-plugin-renderfile-perl_0.10-1_all.deb
$ sudo dpkg -i libmojolicious-plugin-renderfile-perl_0.10-1_all.deb
```

These packages are required to build some dependencies from source:
```
$ sudo apt-get install gcc gcc-4.8 make libssh2-1-dev libnet-ssh2-perl libssh2-1 libdate-calc-perl zlib1g-dev
```

Chapter 3. Ravada VDI documentation
3.3.2 Perl Modules

Some Perl modules must be compiled from source:

```
$ sudo perl -MCPAN -we 'install "Net::SSH2"'
```

3.3.3 Database and configuration

From now on you can follow the instructions for Ubuntu 18.04. Skip to the MySQL installation step.

Install Ravada in Ubuntu 18.04.

3.4 Running Ravada in production

Ravada has two daemons that must run on the production server:

- `rvd_back`: must run as root and manages the virtual machines
- `rvd_front`: is the web frontend that sends requests to the backend

3.4.1 System services

Configuration for boot start

There are two services to start and stop the two ravada daemons:

After install or upgrade you may have to refresh the systemd service units:

```
$ sudo systemctl daemon-reload
```

Check the services are enabled to run at startup

```
$ sudo systemctl enable rvd_back
$ sudo systemctl enable rvd_front
```

Start

```
$ sudo systemctl start rvd_back
$ sudo systemctl start rvd_front
```

Status

You should check if the daemons started right the very first time with the status command. See troubleshooting frequently problems if it failed to start.

```
$ sudo systemctl status rvd_back
$ sudo systemctl status rvd_front
```

Stop

```
$ sudo systemctl stop rvd_back
$ sudo systemctl stop rvd_front
```
3.4.2 Qemu

Ravada uses Qemu to manage the virtual machines. We encourage change this settings so hibernated machines are stored compressed. This way you will save a large amount of disk space on your server.

Edit the file `/etc/libvirt/qemu.conf` and uncomment and change this line:

```
save_image_format = "bzip2"
```

You have to restart libvirt after changing this file:

```
$ sudo systemctl restart libvirtd
```

3.4.3 Apache

You can reach the Ravada frontend heading to `http://your.server.ip:8081/`. It is advised to run an Apache server or similar before the frontend.

In order to make ravada use apache, you must follow the steps explained on [here](#).

3.4.4 Firewall

Ravada uses `iptables` to restrict the access to the virtual machines. These iptables rules grants access to the admin workstation to all the domains and disables the access to everyone else. When the users access through the web broker they are allowed to the port of their virtual machines. Ravada uses its own iptables chain called ‘ravada’ to do so:

```
-A INPUT -p tcp -m tcp -s ip.of.admin.workstation --dport 5900:7000 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 5900:7000 -j DROP
```

3.4.5 Help

Struggling with the installation procedure? We tried to make it easy but let us know if you need assistance.

There is also a [troubleshooting](#) page with common problems that admins may face.

If you do not know how to create a virtual machine, please read [creating virtual machines](#).

3.5 Post Install Recommendations

3.5.1 Firewall

The server must be able to send DHCP packets to its own virtual interface.

KVM should be using a virtual interface for the NAT domnains.

First we try to find out what is the new internal network:

```
sudo ip route
...
192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
```

So it is the interface virbr0.

Add it to your iptables configuration. This will allow some traffic between the host and the virtual machines: DHCP, DNS and ping.
$ sudo iptables -A INPUT -i virbr0 -p udp -m udp --dport 67:68 -j ACCEPT
$ sudo iptables -A INPUT -i virbr0 -p udp -m udp --dport 53 -j ACCEPT
$ sudo iptables -A INPUT -i virbr0 -p udp -m udp --dport 5353 -j ACCEPT
$ sudo iptables -A INPUT -i virbr0 -p tcp -m tcp --dport 53 -j ACCEPT
$ sudo iptables -A INPUT -i virbr0 -p tcp -m tcp --dport 5353 -j ACCEPT
$ sudo iptables -A INPUT -i virbr0 -p icmp -m icmp --icmp-type 8 -j ACCEPT
$ sudo iptables -A OUTPUT -o virbr0 -p udp -m udp --sport 67:68 -j ACCEPT
$ sudo iptables -A OUTPUT -i virbr0 -p udp -m udp --sport 53 -j ACCEPT
$ sudo iptables -A OUTPUT -i virbr0 -p udp -m udp --sport 5353 -j ACCEPT
$ sudo iptables -A OUTPUT -i virbr0 -p icmp -m icmp --icmp-type 8 -j ACCEPT

To confirm that the configuration was updated, check it with:

$ sudo iptables -S

3.5.2 Configuration

The frontend has a secret passphrase that should be changed. Cookies and user session rely on this. You can have many passphrases that get rotated to improve security even more.

Change the file /etc/rvd_front.conf line secrets like this:

```
, secrets => ['my secret 1', 'my secret 2']
```

3.6 Development release

**Note:** If you are not sure, you probably want to install the stable release. Follow this [guide](#).

You can get the development release cloning the sources.

```
$ git clone https://github.com/UPC/ravada.git
```

3.6.1 Possible development scenarios where to deploy

Obviously if you can deploy on a physical machine will be better but it is not always possible. In that case you can test on a nested KVM, that is, a KVM inside another KVM.

**Note:** KVM requires **VT-X / AMD-V**.

```
$ sudo apt install cpu-checker
$ sudo kvm-ok
```

**Warning:** Do not consider VirtualBox in this situation, because it doesn’t pass VT-X / AMD-V to the guest operating system.
3.6.2 Ubuntu required packages

Check this file at the line `depends` for a list of required packages. You must install it running:

```
$ sudo apt-get install perl libmojolicious-perl mysql-common libauthen-passphrase-perl libdbd-mysql-perl ...
```

In addition you need one package that it still may not be in Ubuntu repository, download from our own server at the UPC ETSETB repository and install it this way:

```
$ wget http://infoteleco.upc.edu/img/debian/libmojolicious-plugin-renderfile-perl_0.10-1_all.deb
$ sudo dpkg -i libmojolicious-plugin-renderfile-perl_0.10-1_all.deb
```

3.6.3 Mysql Database

MySQL server is required to run ravada. You can use one from another server you already have or you can install it in the same host as Ravada.

MySQL user

Create a database named “ravada”. In this stage the system wants you to identify a password for your sql.

```
$ mysqladmin -u root -p create ravada
```

Grant all permissions to your user:

```
mysql -u root -p
mysql> grant all on ravada.* to rvd_user@'localhost' identified by 'choose a password';
exit
```

3.6.4 Config file

Create a config file at `/etc/ravada.conf` with the username and password you just declared at the previous step.

```
db:
  user: rvd_user
  password: *****
```

When developing Ravada, your username must be able to read the configuration file. Protect the config file from others and make it yours.

```
$ sudo chmod o-rx /etc/ravada.conf
$ sudo chown your_username /etc/ravada.conf
```

3.6.5 Ravada web user

Add a new user for the ravada web. Use `rvd_back` to create it.

```
$ cd ravada
$ sudo ./bin/rvd_back.pl --add-user user.name
```
### 3.6.6 Firewall(Optional)

The server must be able to send DHCP packets to its own virtual interface.

KVM should be using a virtual interface for the NAT domains. Look what is the address range and add it to your iptables configuration.

First we try to find out what is the new internal network:

```
sudo route -n
```

```
192.168.122.0 0.0.0.0 255.255.255.0 U 0 0 0 virbr0
```

So it is 192.168.122.0, netmask 24. Add it to your iptables configuration:

```
-A INPUT -s 192.168.122.0/24 -p udp --dport 67:68 --sport 67:68 -j ACCEPT
```

### 3.6.7 Client

The client must have a spice viewer such as virt-viewer. There is a package for Linux and it can also be downloaded for Windows.

### 3.6.8 Daemons

Ravada has two daemons that must run on the production server:

- `rvd_back`: must run as root and manages the virtual machines
- `rvd_front`: is the web frontend that sends requests to the backend

Run each one of these commands in a separate terminal

```
$ morbo ./rvd_front.pl
$ sudo ./bin/rvd_back.pl
```

Now you must be able to reach ravada at the location `http://your.ip:3000/`

If you wish to create a script to automatize the start and shutdown of the ravada server, you can use these two bash scripts:

```
#!/bin/bash
#script to initialize ravada server

display_usage()
{
    echo "./start_ravada 1 (messages not prompting to terminal)
    echo "./start_ravada 0 (prompts enables to this terminal)

if [ $# -eq 0 ]
then
    display_usage
    exit 1
else
    SHOW_MESSAGES=$1
    if [ $SHOW_MESSAGES -eq 1 ]
    then
```

(continues on next page)
morbo ./rvd_front.pl > /dev/null 2>&1 &
sudo ./bin/rvd_back.pl > /dev/null 2>&1 &
else
  morbo ./rvd_front.pl &
sudo ./bin/rvd_back.pl &
fi
fi

shutdown_ravada.sh:

#!/bin/bash
#script to shutdown the ravada server
sudo kill -15 $(pidof './rvd_front.pl')
sudo kill -15 $(pidof -x 'rvd_back.pl')
echo "Server closed succesfully"

3.7 Ubuntu installation

This document aims to demonstrate how to install Ubuntu operating system on user computer.

Tip: You can try Ubuntu Desktop or Server, last is recommended.

3.7.1 Steps

1. The user needs at least 4.5 GB of free space on their computer.
2. Connect your USB or DVD containing Ubuntu program.
3. When you turn on your computer the below image must show up automatically or by pressing F12.
4. Make sure you are connected to internet, then the below image is shown. Mark both options and click on “continue”.
5. Below shows how to Use the checkboxes to choose whether you’d like to Install Ubuntu alongside another operating system, delete your existing operating system and replace it with Ubuntu. in our case we select “Something Else” and click on “continue”.
6. In this stage, you will create partitions.
7. The last step is choosing your language and region. After doing so and restarting your computer you can start using Ubuntu.

3.8 Add KVM storage pool

If you run out of disk space you may add a new disk.

Note: KVM mush then be informed about this new space available by creating a new storage pool.
3.8.1 Add the drive to the system

After booting with the new drive, check dmesg to find out the name of the new disk. It will probably be called /dev/sdSOMETHING.

Double check this is actually the new disk, if not you may erase all the contents of the system. Type df to see the old disk partitions.

Create a new partition with fdisk. It should show it as empty. Add only one primary partition for all the free space.

Replace sdX by the real name of the new device:

```
$ sudo fdisk /dev/sdX
```

Format it with large files tunning:

```
$ sudo mkfs.ext4 -m 0.001 -T largefiles /dev/sdX1
```

3.8.2 Mount the new partition

Add this new partition to the filesystem table:

```
sudo mkdir /var/lib/libvirt/images.2
sudo vim /etc/fstab
/dev/sdb1 /var/lib/libvirt/images.2 ext4 auto 0 3
```

It will mount it next time you boot, but it can be used without rebooting issuing:

```
$ sudo mount -a
```

3.8.3 Add the drive to the Virtual Manager

```
$ sudo virsh pool-define-as pool2 dir - - - - /var/lib/libvirt/images.2
$ sudo virsh pool-autostart pool2
$ sudo virsh pool-start pool2
$ sudo virsh pool-list
```

And that’s it, now Ravada will use the pool that has more empty space the next time it needs to create a volume. If you want to fine tune what storage pool is used by default follow the advanced settings documentation.

Run Hypnotoad service and Apache as a proxy for it.

3.9 Configure Hypnotoad proxy

First of all you need to tell hypnotoad we are behind a proxy. This allows Mojolicious to automatically pick up the X-Forwarded-For and X-Forwarded-Proto headers.

Edit the file /etc/rvd_front.conf and make sure there is a line with proxy => 1 inside hypnotoad.

```
hypnotoad => {
    pid_file => '/var/run/ravada/rvd_front.pid'
    ,listen => ['http://*:8081']
    ,proxy => 1
}
```

Restart the front server to reload this configuration:

```
$ sudo systemctl restart rvd_front
```
3.10 Install Apache

# apt-get install apache2

3.11 Enable apache modules

# a2enmod ssl proxy proxy_http proxy_connect proxy_wstunnel headers

3.12 Apache Proxy Configuration

Link the https configuration and add the proxy lines.

# a2ensite default-ssl

Edit /etc/apache2/sites-enabled/default-ssl.conf

```bash
<IfModule mod_ssl.c>
    <VirtualHost _default_:443>
        ProxyRequests Off
        ProxyPreserveHost On
        ProxyPass /ws/ ws://localhost:8081/ws/ keepalive=On
        ProxyPass / http://localhost:8081/ keepalive=On
        ProxyPassReverse / http://localhost:8081/
        RequestHeader set X-Forwarded-Proto "https"
    </VirtualHost>
</IfModule>
```

3.13 Apache redirect to https

Redirect all the connections to https.

Edit /etc/apache2/sites-enabled/000-default.conf

```bash
<VirtualHost *:80>
    ServerName hostname.domainname
    Redirect / https://hostname.domainname/
</VirtualHost>
```

Tip: Remember restart Apache2 service, with systemctl restart apache2 or services apache2 restart.

$ sudo systemctl restart apache2

3.14 How to import a Virtualbox image

Note: In this example we have VirtualBox machine called EXAMPLE.
3.14.1 Create an empty Virtual Machine

From the Ravada admin form, create a new virtual machine with the same operative system as the one installed in the virtual box machine.

Do not install anything in that machine, keep it off. Check what is the name of the disk volume and remove the other volumes.

Check the contents file attribute with the command `virsh edit EXAMPLE`,

```bash
source file='/var/lib/libvirt/images/EXAMPLE-vda-id8Q.img'/
```

Remove the swap, cdrom and other disk volumes.

This is the SW AP volume, notice its name ends in .SWAP.img.

```xml
<disk type='file' device='disk'>
  <driver name='qemu' type='raw' cache='none'/>
  <source file='/var/lib/libvirt/images/EXAMPLE-aGam.SWAP.img'/>
  <target dev='vdb' bus='virtio'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>
</disk>
```

This is the cdrom disk drive, remove it too.

```xml
</disk>
```

Remove also the SW AP image file:

```bash
$ sudo rm /var/lib/libvirt/images-celerra1/EXAMPLE-_G_m.SWAP.img
```

3.14.2 Convert the image file

Make sure the VirtualBox machine is down, then convert the VDI to raw, then to qcow2.

This converted image will be used by the empty virtual machine that was created before.

**DIRECTLY VDI TO QCOW2**

```bash
$ qemu-img convert -p -f vdi -O qcow2 EXAMPLE.vdi EXAMPLE.qcow2
```

**OR IN TWO STEPS**

1. **Convert to raw**

```bash
$ VBoxManage clonehd --format RAW EXAMPLE.vdi EXAMPLE.img
```

2. **Convert to qcow2**

Convert to qcow2 using the name you saw before in the XML definition of the machine:

```bash
$ sudo qemu-img convert -p -f raw EXAMPLE.vdi -O qcow2 /var/lib/libvirt/images/EXAMPLE-vda-id8Q.img
```
3.15 How to create a Virtual Machine

It is probable that when you start reading this document, you have set up and install Ravada successfully and now it is time to create a virtual machine.

3.15.1 Steps

1. First of all, log into Ravada using the username and password you created in previous steps of installation.

2. On top right of the page, click on “Admin Tools” and then “Machines”.

![Image of login screen]

3. In the new page, click on “New Machine”.

3.15. How to create a Virtual Machine
4. Choose a name for your virtual machine and choose an option for the ISO image. The selected image needs to be installed accordingly.
5. Now, you can see a list of available machines and the operations of each.

**Note:** Please note that you can see any upcoming error or problems in section “messages”.

### 3.16 How to dump a hard drive to Ravada

#### 3.16.1 Introduction

#### 3.16.2 Tools

- External hard disk or something similar, you will need to save about 60GB or 100GB.
3.16.3 Procedure

See the video: https://www.youtube.com/watch?v=KtgdWsyNemA

3.17 How to Install a local LDAP

3.17.1 Install 389-ds

$ sudo apt-get install 389-ds-base

3.17.2 Configure directory server

Release 1.3 [old]

This is the configuration tool for older releases of 389 directory server. If there is no setup-ds tool in your system you probably have the new release, skip to Release 1.4 instruction bellow.

$ sudo setup-ds

When requested the server name, answer with the full qualified domain name of the host: hostname.domainname. In the next step you must supply the domain name as base for the configuration. So if your domain name is “foobar.com”, the base will be “dc=foobardc=com”.

Release 1.4 [new]

From release 1.4 we provide an example configuration file for creating the new directory instance. Review it at /etc/ds389.conf and use it with dscreate:

$ sudo dscreate from-file /etc/ds389.conf

Enable and Start the service

$ sudo systemctl start dirsrv@localhost
$ sudo systemctl enable dirsrv@localhost

3.17.3 Add a LDAP section in the config file

The config file usually is /etc/ravada.conf. Add this configuration:

```
ldap:
  admin_group: test.admin.group
  admin_user:
    dn: cn=Directory Manager
    password: 12345678
    base: 'dc=example,dc=com'
```

3.17.4 Insert one test user

The ravada backend script allows creating users in the LDAP

$ sudo ./bin/rvd_back.pl --add-user-ldap jimmy.mcnulty
3.18 How to add a KVM template

ISO images are required to create KVM virtual machines. They can be placed or downloaded at run time.

3.18.1 Placing your own ISO image

Copy the .iso file to the KVM storage, it is /var/lib/libvirt/images by default. Make sure everybody can read it

```
# chmod 755 file.iso
```

Get the md5 for the ISO file, you will need it for the next step:

```
# md5sum file.iso
```

Add an entry to the SQL table:

```
mysql -u rvd_user -p ravada
mysql> INSERT INTO iso_images (name, description, arch, xml, xml_volume, md5, sha256, device)
VALUES ('name','the description', 'i386', 'name.xml' ,'name-vol.xml','bbblamd5sumjustgenerated','mysha256sum','/var/lib/libvirt/images/file.iso');
```

3.18.2 XML file

A XML template file is required if you want to create machines from this ISO. In the directory /var/lib/ravada/xml there are examples. You can make new ones creating a new machine from another tool like virt-manager. Once it is done dump the xml with

```
# virsh dumpxml machine > name.xml
```

3.18.3 XML Volume file

Create a new xml volume file based in another one from /var/lib/ravada/xml.

3.18.4 URL based ISO (simplified)

For most Linux based distributions, you won’t need to manually download the ISO. Here we’re assuming that there are a valid VM definition XML and a volume XML files (based on Ubuntu 16.04 Xenial Xerus).

```
mysql -u rvd_user -p ravada
mysql> INSERT INTO iso_images (name, description, arch, xml, xml_volume, url, sha256_url)
VALUES ('Mint 18.2 BETA Mate 64 bits','Mint Serena 18.2 BETA with Mate Desktop based on Ubuntu Xenial 64 bits', 'amd64', ... 'https://ftp.heanet.ie/mirrors/linuxmint.com/testing/sha256sum.txt');
```

3.18.5 Windows specifics

For Windows you will need the virtio ISO that can be downloaded from https://fedorapeople.org/groups/virt/virtio-win/direct-downloads/stable-virtio/virtio-win.iso

Save it to /var/lib/libvirt/images and change the owner as you did for the Windows ISO.

```
# chmod 755 /var/lib/libvirt/images/virtio-win-0.1.126.iso
```

Then edit your Windows xml file and point the second CD drive to that ISO. For the current stable virtio version, it looks like this: virsh edit machinename
<disk type='file' device='cdrom'>
    <driver name='qemu' type='raw'/>
    <source file='/var/lib/libvirt/images/virtio-win-0.1.126.iso'/>
    <target dev='hdc' bus='ide'/>
    <readonly/>
    <address type='drive' controller='0' bus='1' target='0' unit='0'/>
</disk>

You should also ensure that the system disk cache is set to ‘directsync’:

<driver name='qemu' type='qcow2' cache='directsync' io='native'/>

If you’re using the NEC xhci USB controller (the default one in our environment), you’ll need to obtain a suitable driver for the µPD720200 chipset. Plugable.com has it here http://plugable.com/drivers/renesas (2nd entry).

### 3.19 New ISO image

In order to use an ISO file when you create a new machine, you must first place it inside the KVM directory:

/var/lib/libvirt/images

Then you have to tell the storage engine that you changed a file manually.

```
sudo virsh pool-list
```

<table>
<thead>
<tr>
<th>Name</th>
<th>State</th>
<th>Autostart</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>active</td>
<td>yes</td>
</tr>
<tr>
<td>pool2</td>
<td>active</td>
<td>yes</td>
</tr>
</tbody>
</table>

```
sudo virsh pool-refresh default
sudo virsh pool-refresh pool2
```

Reload the new machine form so the file you just uploaded shows up in the ISO list.

After that, Ravada is able to use he ISO when selecting it while creating a machine. Also, ISOs that were downloaded from Ravada can also be found in this directory.

If you want to include a KVM templated instead, use this guide.

If you need the xml or the volume (for when you wanna add a new template), you can extract them via virsh using the following commands:

```
$ sudo virsh pool-list #in order to see the list of available pools
$ sudo virsh vol-list default #in order to see the list of volumes available
$ sudo virsh vol-dumpxml --pool default name_of_the_existing_machine.qcow2 #obtain the dump of a selected volume
$ sudo virsh dumpxml name_of_the_existing_machine #obtain the xml of a existing selected machine
```

### 3.20 How to import a OpenGnsys image

First of all, copy the .img OpenGnsys image file to your Ravada system.

The .img OpenGnsys files are raw disk dump compressed with lzop. You can see the contents of a img lzop file with:

```
$ lzop -l B5part3dataUbuntu.img
```

<table>
<thead>
<tr>
<th>method</th>
<th>compressed</th>
<th>uncompr. ratio</th>
<th>uncompressed_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>LZO1X-1</td>
<td>4536669058</td>
<td>7962881402</td>
<td>57.0% B5part3dataUbuntu.img.raw</td>
</tr>
</tbody>
</table>

Now, we will decompress the file. We have to force it because it doesn’t have a .lzop extension:
As you can see, the raw file have no extension.

Now, we have the raw content of our image disk. Opengnsys uses partclone to create an image disk. The next step is dump this raw file to a qcow2 disk using partclone.

Tip: You can get the partclone utilities from opengnsys, you can download from the web: https://partclone.org/download/, or extract from a partclone package for your Linux distribution.

You can inspect the raw file with:

```bash
$ ./partclone.info ./B5part3dataUbuntu
$ Partclone v0.2.38 http://partclone.org
$ unknown mode
$ File system: EXTFS
$ Device size: 69.8 GB
$ Space in use: 69.7 GB
$ Free Space: 85.1 MB
$ Block size: 4096 Byte
$ Used block: 17008739
```

Now, we have to create an empty qcow2 file and dump the raw file inside. First of all, create the qcow2 file. It's important to check the size to ensure that the dump will fit in.

```bash
$ qemu-img create -f qcow2 B5part3dataUbuntu.qcow2 70G
```

Now, we mount the qcow2 file in your system, to dump it.

Tip: You can follow this guide to do it: How to mount a qcow2 disk image

```bash
$ qemu-nbd --connect=/dev/nbd0 ./B5part3dataUbuntu.qcow2
```

Now, who can create the partition structure of your disk. After create it, this is the result:

```bash
$ fdisk /dev/nbd0
$ Disk /dev/nbd0: 90 GiB, 96636764160 bytes, 188743680 sectors
$ Units: sectors of 1 * 512 = 512 bytes
$ Sector size (logical/physical): 512 bytes / 512 bytes
$ I/O size (minimum/optimal): 512 bytes / 512 bytes
$ Disklabel type: dos
$ Disk identifier: 0xc0545c3a
$ Device Boot Start End Sectors Size Id Type
$ /dev/nbd0p1 2048 182454271 182452224 87G 83 Linux
$ /dev/nbd0p2 182454272 188743679 6289408 3G 82 Linux swap / Solaris
```

Now, we have 2 partitions, /dev/nbd0p1 and /dev/nbd0p2. To dump the img disk we have to use the partclone.ext3 utility:

Command to restore:
$ ./partclone.ext3 -s ./B5part3dataUbuntu -O /dev/nbd0p1 -r
$ Partclone v0.2.38 http://partclone.org
$ Starting to restore image (. /B5part3dataUbuntu) to device (/dev/nbd0p1)
$ Calculating bitmap... Please wait... done!
$ File system: EXTFS
$ Device size: 69.8 GB
$ Space in use: 69.7 GB
$ Free Space: 85.1 MB
$ Block size: 4096 Byte
$ Used block : 17008739

The process begin, and you can follow the logs:

$ 00:00:07, Remaining: 00:05:36, Completed: 2.04%, Rate: 12.16GB/min,
$ Elapsed 00:00:01, Completed: 99.97%, Rate: 1.23GB/min,
$ Elapsed: 00:56:28,
$ Remaining: 00:00:00, Completed: 99.98%, Rate: 1.23GB/min,
$ Elapsed: 00:56:29,
$ Remaining: 00:00:00, Completed:100.00%, Rate: 1.23GB/min,
$ Elapsed: 00:56:29, Remaining: 00:00:00,
$ Completed:100.00%, Rate: 1.23GB/min,
$ Total Time: 00:56:29, Ave. Rate: 1.2GB/min, 100.00% completed!
$ Total Time: 00:56:29, Ave. Rate: 1.2GB/min, 100.00% completed!
$ Syncing... OK!
$ Partclone successfully restored the image (. /B5part3dataUbuntu) to the device (/dev/nbd0p1)
$ Cloned successfully.
$ root@willow: /ssd/estegoxCloneC6root@willow:/ssd/estegoxCloneC6#

$ Now, you can verify the filesystem, mounting it:

$ mount /dev/nbd0p1 /mnt/suse
$ ls -als /mnt/suse/
$ total 168
$ 4 drwxr-xr-x 4 root root 4096 Mar 1 13:55 ..
$ 4 drwxr-xr-x 2 root root 4096 Feb 3 2017 assig
$ 4 -rw------ 1 root root 199 Mar 2 11:42 .bash_history
$ 4 drwxr-xr-x 2 root root 4096 Feb 2 11:51 bin
$ 4 drwxr-xr-x 4 root root 4096 Mar 2 12:30 boot
$ 4 drwxr-xr-x 3 root root 4096 May 10 2017 mnt
$ 20 -rw-r--r-- 1 root root 19732 Sep 23 2015 ogAdmLnxClient.log
$ 4 drwxr-xr-x 80 root root 4096 Feb 19 11:33 opt
$ ...

Maybe didn’t full the entire disk. You can expand it to fit all the disk:

$ umount /mnt/suse
$ e2fsck /dev/nbd0p1
$ e2fsck 1.43.5 (04-Aug-2017)
$ /dev/nbd0p1: clean, 1897474/5701632 files, 16969078/22806528 blocks
$ resize2fs /dev/nbd0p1

Now, unmount que qcow2 file:

$ qemu-nbd --disconnect /dev/nbd0
And that’s all! Now you can create a Ravada vm and attach the disk.

It’s possible that the system needs some extra adjustments. One typical problem is modify the `/etc/fstab` to change the `/dev/sda` references to `/dev/vda`. Another common problem is recreate the grub boot or add support to `/dev/vda` devices.

### 3.21 Integrating Ravada and OpenGnsys

Opengnsys is an open source project for remote deployment. This is a project developed for many Spanish universities to provide a full tool to deploy, manage, clone and manage remote computers. Opengnsys allow distribute and install many different operating systems.

Opengnsys is based in a PXE boot and a Linux graphical agent that allows manage remotely the computer from a centralized console. Here, we will explain how adapt our RAVA system to support boot from Opengnsys. The final objective is automate the creation a virtual machine with the same image that we have created for our classrooms.

#### 3.21.1 DHCP boot options

First of all, we have to provide the dhcp options `next-server` and `filename` to our dhcp server. Ravada is a KVM-based solutions, so, the dhcp server is the standard integrated in KVM. The DHCP-KVM server allows some configurations. Edit the KVM network configuration and add these options to the dhcp section:

```bash
$ virsh#virsh net-edit default
$ <network>
$ <name>default</name>
$ <uuid>85909d3b-f219-4055-92a3-d36c0c57810c</uuid>
$ <forward mode='nat'/>
$ <bridge name='virbr0' stp='on' delay='0'/>
$ <mac address='52:54:00:1a:06:50'/>
$ <ip address='192.168.122.1' netmask='255.255.255.0'>
$ <tftp root='/'/>
$ <dhcp>
$ <range start='192.168.122.30' end='192.168.122.254'/>
$ <bootp file='grldr' server='<opengnsys-server-ip'/>}
$ </dhcp>
$ </ip>
$ </network>
```

`grldr` is the standard boot loader for Opengnsys.

#### 3.21.2 Create and empty virtual machine

Now, you have to create and empty virtual machine. And empty machine boots from the iPXE network boot firmware client integrated in KVM. This is a snapshot of a vm booting process:
3.21.3 NAT adaptation

Now, we have detected that TFTP doesn’t work with the default KVM NAT configuration. You have to add support for it. This document explain it: https://beaveryoga.wordpress.com/2016/12/10/iptables-and-tftp-howto/

3.21.4 Create the virtual machine in the Opengnsys console

We have to create the support configuration to this virtual PC in the Opengnsys console.

The virtual machine runs inside a NATed network, usually with a 192.168.122.0/24 IP address. Then, these vms uses the Ravada server as gateway. We have to create an new classroom with the NAT configuration to allow opengnsys to assign correctly the network mask and the gateway. This is the ravada-classroom configuration:
• gateway: 192.168.122.1 (KVM NAT default gateway)
• netmask: 255.255.255.0 (KVM NAT default netmask)
• IP multicast: your multicast group
• Menu: your page menu
• Repository: your image repository

Now, we have to create a computer inside your ravada classroom that is your virtual machine. Copy the MAC address of your empty machine:

```
$ virsh net-dhcp-leases default
```

```
+------------+-----------------+-----------------+-----------------+-----------------+
<table>
<thead>
<tr>
<th>Expiry Time</th>
<th>MAC address</th>
<th>Protocol</th>
<th>IP address</th>
<th>Hostname</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-11-27 09:11:39</td>
<td>52:54:00:a7:49:34</td>
<td>ipv4</td>
<td>192.168.122.178/24</td>
<td>-</td>
</tr>
</tbody>
</table>
```
And now, re-generate the TFTPBOOT files:
In this example, we have assigned the new PC to the ogAdmin group.

Now, you can boot the empty machine:

We have detected that the new machine boots, but it hangs just when the menu had to appear. After debugging, we have detected that the virtual machine don’t have access to the http server with the menus. This a problem with routing.

We have resolved creating a fake computer with the IP and MAC address of the KVM external NAT:

```bash
$ ifconfig
$ br0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
$ inet 10.10.73.24 netmask 255.255.255.0 broadcast 10.10.73.255
$ inet6 fe80::20a:f7ff:feba:c980 prefixlen 64 scopeid 0x20<link>
$ ether 00:0a:f7:ba:c9:80 txqueuelen 1000 (Ethernet)
$ RX packets 11251336 bytes 196755808380 (196.7 GB)
$ RX errors 0 dropped 0 overruns 0 frame 0
$ TX packets 11875794 bytes 4220061188 (4.2 GB)
$ TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```
• IP: external NAT address of your RAVADA system

• MAC: external MAC address of your RAVADA system

This is our standard menu:
Now, you can boot your standard images in a virtual environment of Ravada. You have to be sure that your images have support to run in a virtualized system. In Linux images, the kernel have support /dev/vda devices. In Windows systems, you have to add the virtio drivers.

### 3.21.5 Special script adaptation

Our images boots ok, but our opengnsys instance doesn’t detect the virtual disk. The problem was in our system, wich is very old (v1.0.5). To add support to detect /dev/vda devices, we have patched the /opt/opengnsys/client/lib/engine/bin/Disk.lib library:

```bash
$ # Listar dispositivo para los discos duros (tipos: 3=hd, 8=sd 253=vda). inLab 2018
$ ALLDISKS=$(awk '($1==3 || $1==8 || $1==253) && $4!~/[0-9]/ printf "/dev/%s ",$4' /proc/partitions)
$ VOLGROUPS=$(vgs -a --noheadings 2>/dev/null | awk 'printf "/dev/%s ",$4')
$ ALLDISKS="$ALLDISKS $VOLGROUPS"
```

This patch adds vda disk detection to the ogDiskToDev function. (minor 253 -> vda devices). This problem was fixed in later versions.

### 3.22 Operation

3.22. Operation
3.22.1 Create users

$ sudo ./usr/sbin/rvd_back --add-user=username
$ sudo ./usr/sbin/rvd_back --add-user-ldap=username

3.22.2 Import KVM virtual machines

Usually, virtual machines are created within ravada, but they can be imported from existing KVM domains. Once the domain is created:

$ sudo ./usr/sbin/rvd_back --import-domain=a

It will ask the name of the user the domain will be owned by.

3.22.3 View all rvd_back options

In order to manage your backend easily, rvd_back has a few flags that lets you made different things (like changing the password for an user).

If you want to view the full list, execute:

$ sudo rvd_back --help

3.22.4 Admin

Note: Must run from the frontend

Create Virtual Machine

Go to Admin -> Machines and press New Machine button.

If anything goes wrong check Admin -> Messages for information from the Ravada backend.

ISO MD5 mismatch

When downloading the ISO, it may fail or get old. Check the error message for the name of the ISO file and the ID.

Option 1: clean ISO and MD5

- Remove the ISO file shown at the error message
- Clean the MD5 entry in the database:

$ mysql -u rvd_user -p ravada mysql > update iso_images set md5='' WHERE id=*ID*

Then you have to create the machine again from scratch and make it download the ISO file.
Option 2: refresh the ISO table

If you followed Option 1 and it still fails you may have an old version of the information in the isoimages table. Remove that entry and insert the data again:

```
$ mysql -u rvd_user -p ravada -e "DELETE FROM iso_images WHERE id=_ID_"
```

Insert the data from the SQL file installed with the package:

```
$ mysql -u rvd_user -p ravada -f /usr/share/doc/ravada/sql/data/insert_iso_images.sql
```

It will report duplicated entry errors, but the removed row should be inserted again.

### 3.22.5 Create base of a Virtual Machine

Go to Admin tools -> Virtual Machines

**1st Base**

If you have configured your Virtual Machine, now you can do the Base:

- Select the Base checkbox.

The Virtual Machine will be published if you select the Public checkbox.

**2nd base or more**

In this case, you have a previous Base and you’ve made some changes at the machine. Now you must prepare a Base again.

Steps:

1. Remove all clones of this Virtual Machine.
2. Select the Base checkbox to prepare base.

### 3.23 Swap Partition

Though installing the Operating System in a Virtual Machine may be the same as in real machines, carefully planning the swap partition of the virtual machines will save a lot of disk space. Follow those guidelines.

#### 3.23.1 Swap Volume

Mark at the creation of the Virtual Machine that swap disk volume. The desired max size must be declared there. That way a different disk will be created with that purpose. This volume is different than regular data disk volumes: it will be created only at the start of the machine and it will be destroyed at shutdown. Also, this volume won’t keep incremental changes from the base, as data volumes do.

#### 3.23.2 Partitioning

Later on we will address particular considerations for swap space in different operating systems. By now, keep in mind that the best practice is to keep a disk volume *only for swap*. 
3.23.3 Linux

It is recommended to keep the swapping to the less possible. If possible remove the swap partitions and the swap configuration in /etc/fstab.

Some software on Linux requires some swap to run. If so, set the `swappiness` to the minimum this way:

```
$ sudo sysctl vm.swappiness=1
```

To make this change permanent add it to the file: `/etc/sysctl.conf`

3.24 Troubleshooting frequent problems

3.24.1 Could not access KVM kernel module:

The system shows this message on trying to start a virtual Machine:

```
Could not access KVM kernel module: Permission denied failed to initialize KVM:
˓→Permission denied
```

That means the host has no virtual capabilities or are disabled. Try running:

```
$ sudo tail -f /var/log/syslog
$ sudo modprobe kvm-intel
```

If it shows a message like this it means the BIOS Virt feature must be enabled:

```
kvm: disabled by bios
```

or try: `kvm-ok` command

```
kvm-ok
INFO: /dev/kvm does not exist
HINT: sudo modprobe kvm_intel
INFO: Your CPU supports KVM extensions
INFO: KVM (vmx) is disabled by your BIOS
HINT: Enter your BIOS setup and enable Virtualization Technology (VT),
and then hard poweroff/poweron your system
KVM acceleration can NOT be used
```

3.24.2 Dealing with permissions

The system may deny access to some directories.

**On Screenshots (requires review)**

That problem showed up in Vanilla Linux 4.10.

When running the screenshot command it returns:

```
failed to open file '/var/cache/libvirt/qemu/qemu.screendump.31DvW9': Permission denied
```
Apparmor

At the file: /etc/apparmor.d/usr.lib.libvirt.virt-aa-helper
/var/cache/libvirt/qemu/ rw,
/var/cache/libvirt/qemu/** rw,

3.24.3 Error with MySQL version < 5.6

For example the following message:

DBD::mysql::db do failed: Invalid default value for 'date_send' at /usr/share/perl5/Ravada.pm line 276.

DEFAULT CURRENT_TIMESTAMP support for a DATETIME (datatype) was added in MySQL 5.6.
Upgrade your MySQL server or change: datetime for timestamp

date_send datetime default now(),

More information about.

3.24.4 Spice-Warning Error in certificate chain verification

(spicec:/usr/bin/remote-viewer:2657): Spice-Warning **: ssl_verify.c:429:openssl_verify:
  Error in certificate chain verification: self signed certificate in certificate
  chain (num=19:depth1:/C=IL/L=Raanana/O=Red Hat/CN=my CA)

spicec looks for %APPDATA%spicec/spice_truststore.pem / $HOME/.spicec/spice_truststore.pem. This needs to be identical to the ca-cert.pem on the server, i.e. the ca used to sign the server certificate. The client will use this to authenticate the server.

3.24.5 Network is already in use

If running VMs crash with that message:

  libvirt error code: 1, message: internal error: Network is already in use by interface

You are probably running Ravada inside a virtual machine or you are using the private network that KVM uses for another interface. This is likely to happen when running Ravada in a Nested Virtual environment.

Solution: Change the KVM network definition. Edit the file /etc/libvirt/qemu/networks/default.xml and replace all the 192.168.122 network instances by another one, ie: 192.168.123.

sudo virsh net-edit default
<ip address='192.168.122.1' netmask='255.255.255.0'>
  <dhcp>
    <range start='192.168.122.2' end='192.168.122.254'/>
  </dhcp>
</ip>

Then reboot the whole system.

3.24. Troubleshooting frequent problems
3.24.6 Copy & paste integration does not work

Make sure that the VM has a Spice communication channel (com.redhat.spice.0) and that the guest additions have been installed.

The Spice channel can be added through virt-manager’s Add Hardware wizard or editing the XML:

```
<channel type='spicevmc'>
  <target type='virtio' name='com.redhat.spice.0'/>
  <address type='virtio-serial' controller='0' bus='0' port='1'/>
</channel>
```

Linux guests must install the spice-vdagent package, while Windows guests require this installer (source)

3.24.7 Resizing the viewer window does not change the guest display resolution

This feature requires the Spice communication channel and the guest additions. See above for instructions.

3.24.8 Windows 10 perfomance issues

*thanks to @rlunardo*

- Windows10 Enterprise ISO image (Home/Professional/Enterprise) before April 2017: if you install Enterprise version, it does not reach the end of installation. Issue posted on 30/10/2017. The Professional version does complete the installation. Recent Enterprise ISO image release completes the installation also.

- Windows 10 tuning after installation: There are several web site where we can find informations and solutions to solve CPU, RAM, Disk overload on Windows 10. Here some links:
  - https://youtu.be/iHzEp8a8w10

3.24.9 Problems with the time of the VM guest

You create a VM and you set the time correctly. After this VM becomes base and the time appears altered (-2h, +2h,...)

This is due to the parameter:

```
<clock offset='utc'>  vs  <clock offset='localtime'>
```

You can modify XML file from the command:

# virsh edit <machine_name>

3.25 Upgrade Ravada

In order to upgrade Ravada, you have to do a few steps:
3.25.1 Steps for a clean update

Step 1

Download the *deb* package of the new version found at the UPC ETSETB repository.

Step 2

Install the *deb* package.

```
$ sudo dpkg -i <deb file>
```

On some upgrades may be required to install some dependencies. You will see because the packaging system will warn about it:

```
dpkg: dependency problems prevent configuration of ravada:
  ravada depends on libdatetime-perl; however:
  Package libdatetime-perl is not installed.
```

If so, install those dependencies automatically running:

```
$ sudo apt-get -f install
```

Step 3

Reconfigure the systemd.

```
$ sudo systemctl daemon-reload
```

Step 4

Restart the services.

```
$ sudo systemctl restart rvd_back
$ sudo systemctl restart rvd_front
```

3.26 Windows SPICE Clients

3.26.1 Download Virt Viewer

Windows clients requires the *virt-viewer* tool to connect to their Virtual Machine.

3.26.2 Fix Windows registry

If *virt-viewer* won’t start automatically after when viewing the virtual machine, add this to the Registry, or download *spice.reg*. (Thanks to @gmiranda).

```
Windows Registry Editor Version 5.00

[HKEY_CLASSES_ROOT\spice]
@="URL:spice"
"URL Protocol"=""  
```

(continues on next page)
3.27 How to change the controller driver of a Windows VM to VirtIO

1. Add a new virtio disk (a small one, even 100mb will be enough. We will not be using it)
   • Use virt manager
   • or virsh edit
2. Boot the machine
3. Make sure Windows recognises the disk controller and has the drivers for it (the disk should be visible in diskmgmt.msc)
4. Set boot to fail safe mode
   • Launch an elevated command prompt
   • Type `bcdedit /set {current} safeboot minimal`
5. Shut down
6. Change the main disk to virtio
7. Boot. Now get into the admin prompt and disable failsafe mode.
   • Launch an elevated command prompt
   • Type `bcdedit /deletevalue {current} safeboot`  
8. Reboot and make sure everything works.
9. Now you can shut down the machine and remove the small virtio disk

Source: http://triplescomputers.com/blog/uncategorized/solution-switch-windows-10-from-raidide-to-ahci-operation/

3.28 Virtual Machine Manual Migration

If you have several Ravada servers you may want to copy a virtual machine from one to another.

In this example we copy the base for a virtual machine called Lubuntu-1704.
3.28.1 Temporary space in destination

At the destination server, create a temporary directory so you can store the volumes when you copy them. This
directory must belong to a user that can do ssh from origin to destination:

```
root@destination:~ # mkdir /var/lib/libvirt/images/tmp
root@destination:~ # chown frankie /var/lib/libvirt/images/tmp
```

3.28.2 Import the Base

Copy the Base definition

First copy the base definition file from server origin to destination. You need an user in the destination machine and
ssh connection from each other.

```
root@origin:~ # virsh dumpxml Lubuntu1704 > Lubuntu1704.xml
root@origin:~ # scp Lubuntu1704.xml frankie@dst.domain:
```

Copy the volumes

The volumes have a backing file, you must find out what it is so you can copy to destination.

```
root@origin:~ # grep source Lubuntu1704.xml
<source file="/var/lib/libvirt/images/Lubuntu1704-vda-X18J.img"/>
root@origin:~ # qemu-img info /var/lib/libvirt/images/base-vda-X18J.img | grep -i __backing
backing file: /var/lib/libvirt/images/Lubuntu1704-vda-X18J.ro.qcow2
root@origin:~ # rsync -av /var/lib/libvirt/images/Lubuntu1704-vda-X18J.ro.qcow2 frankie@dst.
   --domain=/var/lib/libvirt/images/tmp
root@origin:~ # rsync -av /var/lib/libvirt/images/Lubuntu1704-vda-X18J.img frankie@dst.
   --domain=/var/lib/libvirt/images/tmp
```

Move the volumes on destination

You just copied the data on a temporary directory available to the user. That must be copied to the actual storage pool
as root. Make sure you don’t have similar volumes there because that procedure will overwrite them:

```
root@dst:/home/frankie# cd /var/lib/libvirt/images/tmp
root@dst:/var/lib/libvirt/images/tmp# mv Lubuntu1704-* ../
root@dst:/var/lib/libvirt/images/tmp# chown root ../Lubuntu1704-*
```

Define the base on destination

Go to the destination server and define the virtual machine base with the XML config you copied before

```
root@dst:~ # cd -frankie/
root@dst:/home/frankie# virsh define Lubuntu1704.xml
Domain base defined from Lubuntu1704.xml
```
**RavadaVDI**

**Import the base to Ravada on destination**

Run this command and you should see the base on the Ravada web admin page.

```
root@dst:~# rvd_back --import-domain Lubuntu1704
```

### 3.28.3 Importing clones

Now if you want to import a clone too, first you have to ask the clone owner to start the machine on destination. Then you have to copy the volumes from origin and overwrite what has just been created on destination.

**Create a clone**

The owner of the original clone must create a clone in destination using Ravada. That will create a basic virtual machine with the same name owned by the correct user. Stop the domain on destination:

```
root@dst:~# virsh shutdown Lubuntu1704-juan-ramon
```

Make sure it is stopped

```
root@dst:~# virsh dominfo Lubuntu1704-juan-ramon
```

**Copy the clone volumes**

Find out what are the clone volume files, and copy them to the temporary space in destination:

```
root@origin:~# virsh dumpxml Lubuntu1704-juan-ramon | grep "source file" | grep -v ".→ro."
<source file='/var/lib/libvirt/images/Lubuntu1704-juan-ramon-vda-kg.qcow2'/>
root@origin:~# rsync -av /var/lib/libvirt/images/Lubuntu1704-juan-ramon-vda-kg.qcow2_
˓→frankie@dst:/var/lib/libvirt/images/tmp/
```

**Start the clone on destination**

First move the volumes to the right place, notice in destination the volumes have different names.

```
root@dst:~# virsh dumpxml Lubuntu1704-juan-ramon | grep source
<source file='/var/lib/libvirt/images.2/Lubuntu1704-juan-ramon-vda-nz.qcow2'/>
root@dst:~# cd /var/lib/libvirt/images/tmp/
root@dst:/var/lib/libvirt/images/tmp# mv Lubuntu1704-juan-ramon-vda-jz.qcow2 ..
˓→Lubuntu1704-juan-ramon-vda-nz.qcow2
root@dst:~# chown root /var/lib/libvirt/images/Lubuntu1704-juan-ramon-
```

Hopefully then you can start the clone. It is a delicate procedure that must be followed carefully, please consider helping with this document if you have any suggestions.

### 3.29 Kiosk Mode

Kiosk (or anonymous) allows any user, not logged in, to create a volatile virtual machine. Once this machine is shut down, it is destroyed automatically.
3.29.1 Setting

This *kiosk* mode must be defined for some bases in some networks.

**Note:** Unfortunately kiosk mode configuration has not been added to the frontend. Anyway it can be set only from within the database.

Follow these steps carefully.

3.29.2 Backup the Database

As we are going to change the database, any mistake can be fatal. Backup before. If you want to have the data handy do it right now:

```
# mysqldump -u root -p ravada domains > domains.sql
# mysqldump -u root -p ravada networks > networks.sql
```

3.29.3 Define a Network

You can allow kiosk mode from any network, but you can define a new network where this mode is allowed.

```
# mysql -u root -p ravada
mysql> insert into networks (name, address) values ('classroom','10.0.68.0/24');
```

3.29.4 Find the ids

You must find what is the id of the network and the virtual machine where kiosk mode is enabled. This domain must be a base and allowed public access.

```
mysql> select id,name from domains where name='blablabla' and is_base=1 and is_public=1;
+----+-------------------+
| id | name              |
+----+-------------------+
| 22 | blablabla         |
+----+-------------------+
mysql> select id,name from networks;
+----+-----------+
| id | name      |
+----+-----------+
| 1  | localnet  |
| 4  | all       |
| 6  | classroom  |
+----+-----------+
```

3.29.5 Allow anonymous mode

```
mysql> insert into domains_network(id_domain, id_network,anonymous) VALUES(33, 6, 1);
```
3.29.6 Access

Access now to the anonymous section in your ravada web server. http://your.ip:8081/anonymous
You should see there the base of the virtual machine you allowed before.

3.30 How to do a volatile Virtual Machine

This means that the virtual machine will be removed when shutdown.

Note: If you enable this option in a base machine, the clones that are created thereafter will be volatile. The VMs cloned so far, if they existed, are not affected.

3.30.1 Enable/disable this option

You can enable or disable this option from the VM settings in the options tab.

3.31 Adding Custom Messages

You can add a custom header and text at the login screen.

3.31.1 Configuration

Configure the file rvd_front.conf like this:

```json
{
    login_header => 'Login',
    login_message => 'Login to Start a Machine'
}
```
3.32 Create a custom login template

If you need custom login template create one and save it in /usr/share/ravada/templates/main/custom, e.g. custom_login.html.ep

3.32.1 Configuration

Add your template in /etc/rvd_front.conf

| Warning: | Check that rvd_front.conf exists. If you work on a Development release you have an example here etc/rvd_front.conf.example. |

| Warning: | Do not include the extension file .html.ep in the path. E.g. custom_login.html.ep -> custom_login |

,login_custom => 'main/custom/custom_login'

3.32.2 Path for CSS, js and images

If CSS, js or images are needed save in: public/css/custom, public/js/custom or public/img/custom respectively.

| Note: | Make sure your CSS, JS or images in custom template refers to those paths. |

3.32.3 Restart frontend

Finally restart rvd_front:

$ sudo systemctl restart rvd_front

3.33 Create a custom footer template

If you need custom footer template create one and save it in /usr/share/ravada/templates/main/custom/, e.g. custom/custom_footer.html.ep

3.33.1 Configuration

Add your template in /etc/rvd_front.conf

| Warning: | Check that rvd_front.conf exists. If you work on a Development release you have an example here etc/rvd_front.conf.example. |
Warning: Do not include the extension file `.html.ep` in the path. E.g. `custom_footer.html.ep` -> `custom_footer`

```
,footer => 'main/custom/custom_footer'
```

### 3.33.2 Path for CSS, js and images

If CSS, js or images are needed save in: `public/css/custom`, `public/js/custom` or `public/img/custom` respectively.

**Note:** Make sure your CSS, JS or images in custom template refers to those paths.

### 3.33.3 Restart frontend

Finally restart `rvd_front`:

```
$ sudo systemctl restart rvd_front
```

### 3.34 Disable Spice Password

When the users start a virtual machine, a password is defined for the spice connection. This can be disabled for a given network.

Unfortunately this settings must be configured directly through SQL commands. There is still no GUI section for this.

#### 3.34.1 Define the network

Define a network with no password setting the `requires_password` field to 0:

```
mysql -u root -p ravada
mysql> insert into networks (name, address, requires_password) values ('classroom','10.0.68.0/24', 0);
```

#### 3.34.2 Applying settings

This settings applies on starting a new virtual machine. So running virtual machines will keep the former settings. Shutting them down and up will trigger the new configuration.

#### 3.34.3 Default setting

Any other network requires password as defined by the ‘0.0.0.0/0’ network setting.

#### 3.34.4 Why is that?

Ravada opens SPICE connections and manages iptables to make sure no one can connect to another user’s virtual machine. This is also enforced through the password setting. Please consider disabling it only in controlled, seat-unique ip environments.
3.35 Ravada advanced settings

3.35.1 Display IP

On a server with 2 IPs, the configuration file allows the administrator define which one is used for the display. Add the entry display_ip to /etc/ravada.conf with the public address of the server.

```
display_ip: public.display.ip
```

3.35.2 NAT

The Ravada server can be behind a NAT environment.

```
   RVD _______________ NAT ________________ client
    Server 1.1.1.1  2.2.2.2
```

Configure this option in /etc/ravada.conf

```
display_ip: 1.1.1.1
nat_ip: 2.2.2.2
```

3.35.3 Auto Start

Virtual machines can be configured to start automatically when the physical host boots.

```
Machine Name | Hide density | Base | Public | Autostart | Status | Actions
-------------|--------------|------|--------|-----------|--------|----------
UbuntuBionic | ✓            | ✓    |        |           |         | Down     
UbuntuBionic-abel [Cloned] | | | | | | Down
UbuntuBionic-jeff [Cloned] | | | | | | Down
UbuntuBionic-trey [Cloned] | | | | | | Down
```

You can enable the auto start column at the frontend configuration file at /etc/rvd_front.conf. Reboot the frontend with systemctl restart rvd_front to display the changes.

```
/etc/rvd_front.conf
{
    admin => {
        autostart => 1
    }
}
```

3.35.4 Choosing Storage Pool
Default Storage Pool

When creating virtual machines, Ravada chooses the storage pool with more free space available. If you want to force another, change the settings updating the table vms in the database like this.

First check the id field of the Virtual Manager in the table vms, then set a default_storage pool this way:

```sql
mysql -u rvd_user -p ravada
mysql> select * from vms;
+----+---------------+-----------------+
| id | name          | default_storage |
+----+---------------+-----------------+
| 1  | KVM_localhost |                 |
+----+---------------+-----------------+
mysql> UPDATE vms set default_storage='pool2' where id=1;
```

Then restart rvd_back running `systemctl restart rvd_back`.

Specific Storage Pools

Specific storages for bases and clones can be defined. This way you can use small and fast disk drives for bases and big but slower disks for clones.

**Warning:** If you set base and clone storages here, the default storage setting is ignored.

Add and define the storage pools as described in the add kvm storage pool manual. Then change the values in the database directly.

First check the id field of the Virtual Manager in the table vms, then set a base_storage or clone_storage pools this way:

```bash
root@ravada:~# virsh pool-list
  Name State Autostart
-----------------------
  pool_ssd active yes
  pool_sata active yes

mysql -u rvd_user -p ravada
mysql> select * from vms;
+----+---------------+-----------------+--------------+---------------+
| id | name          | default_storage | base_Storage | clone_storage |
+----+---------------+-----------------+--------------+---------------+
| 1  | KVM_localhost |                 |              |              |
+----+---------------+-----------------+--------------+---------------+
mysql> UPDATE vms set base_storage='pool_ssd' where id=1;
mysql> UPDATE vms set clone_storage='pool_sata' where id=1;

**Warning:** These change will apply to new virtual machines. Old virtual machines have already the disk volumes defined and can’t be easily changed.

3.35.5 Chek free memory ( from v0.3 )

Before start the domain, free memory of the Virtual Manager can be checked. This feature is only available in the development release.
First check the id field of the Virtual Manager in the table `vms`, then set the minimum of free available memory. In this example we require a minimum of 2 GB free:

```
mysql -u rvd_user -p ravada
mysql> select * from vms;
mysql> update vms set min_free_memory=2000000 where id=*id*;
```

### 3.36 New documentation

We build documentation and host it in Read the Docs.

All documentation files are stored only in `gh-pages` branch, with the following directory structure:

```
docs
  _config.yml
  devel-docs
  docs
    index.rst
```

Documentation is created using reStructuredText, is an easy-to-read, what-you-see-is-what-you-get plaintext markup syntax and parser system.

#### 3.36.1 Procedure

1. Consider the editing style of existing pages.
2. Edit a doc page or create a new one in `gh-pages` branch.
3. Insert in `index.rst` according to the section.

---

**Note:** Documentation web is updated automatically, thanks to Read the Docs.

---

#### 3.36.2 Sidebar

The organization of the sidebar is configured in the `index.rst`. If you create a new documentation page remember to include in the section more according to the content. Add the directory and the name of rst file, e.g.:

```
new_documentation.rst in docs/ will be docs/new_documentation somewhere in the index.rst
```

#### 3.36.3 Convert POD files

Install `libpod-pom-view-restructured-perl` in your computer.

### 3.37 Configure LDAP Authentication

Ravada can use LDAP as the authentication engine.
3.37.1 Configuration

The configuration file is /etc/ravada.conf. The format is YAML, make sure you edit this file with spaces, no tabs.

Add a section ldap like this:

```yaml
ldap:
  server: 192.168.1.44
  port: 389 # or 636 for secure connections
  secure: 0 # defaults to 1 if port is 636
  base: dc=domain,dc=com
  admin_user:
    dn: cn=admin.user,dc=domain,dc=com
    password: secretpassword
```

The _secure_ setting is optional. It defaults to 0 for port 389 (ldap) and to 1 for port 636 (ldaps). It can be enabled so secure connections can be forced for other ports.

The LDAP admin user can be a low level account with minimal privileges.

Another optional setting can be used to force the authentication method. By default Ravada tries first to bind to the LDAP as the user. If that fails then it tries to match the encrypted password. You can force the method with:

```yaml
auth: all # defaults to all, can be all, bind, match
```

Notice matching authentication may be a security risk so bind is recommended.

3.37.2 Example: All users

All the users in the LDAP can have access to ravada:

```yaml
ldap:
  server: 192.168.1.44
  port: 636
  base: dc=domain,dc=com
  admin_user:
    dn: cn=admin.user,dc=domain,dc=com
    password: secretpassword
```

3.37.3 Example: Group of users

Allow only a group of users to access ravada:

```yaml
ldap:
  server: 192.168.1.44
  port: 636
  base: ou=users,ou=groupname,dc=upc,dc=edu
  admin_user:
    dn: cn=admin.user,dc=domain,dc=com
    password: secretpassword
```

3.37.4 Example: Posix Group

If you have all your users under a main OU (e.g. ou=users, dc=domain, dc=com), you can use Posix Groups (https://ldapwiki.com/wiki/PosixGroup) to create a list of users that can access to your Ravada instance, using their
memberUid attribute. This allows you to grant or remove access to ravada to some users without modifying your LDAP structure.

```
ldap:
  server: 192.168.1.44
  port: 636
  base: ou=users,ou=groupname,dc=upc,dc=edu
  ravada_posix_group: cn=ravada,ou=groups,dc=domain,dc=com
  admin_user:
    dn: cn=admin.user,dc=domain,dc=com
    password: secretpassword
```

In the example, `cn=ravada,ou=groups,dc=domain,dc=com` is a Posix Group in your LDAP server. It should contain the memberUid's of the users allowed to access to Ravada:

```
dn: cn=ravada,ou=groups,dc=domain,dc=com
  objectclass: posixGroup
  memberUid: user1
  memberUid: user2
  memberUid: user3
```

### 3.37.5 Example: Attribute Filter

In this example, only the users that pass a filter can login:

```
ldap:
  server: 192.168.1.44
  port: 636
  base: dc=domain,dc=com
  filter: campus=North
  admin_user:
    dn: cn=admin.user,dc=domain,dc=com
    password: secretpassword
```

### 3.38 Authentication with Active Directory

This feature is experimental and only can be used if you have a development release of Ravada.

#### 3.38.1 Install Modules

```
$ sudo apt-get install libtest-spelling-perl
$ sudo apt-get install cpanminus
$ sudo cpanm Auth::ActiveDirectory
```

#### 3.38.2 Configure Ravada

Add this entries to the file `/etc/ravada.conf`. The tag `ActiveDirectory` must be at first level without indentations, the other tags must be space-indented. The port is optional.

---

**3.38. Authentication with Active Directory**

This feature is experimental and only can be used if you have a development release of Ravada.
3.38.3 Run

Restart the rvd_front service and try to login

3.38.4 Admin Users

Admin users must be set from the management tool once they have logged in.

3.38.5 Todo

- Admin users: how to find if an user is admin? Maybe some kind of LDAP group?
- Create users
- Create groups
- Remove users
- Remove groups

3.39 Set access restrictions to Virtual Machines

When a base is set as public, all the users have access to create clones by default. If you want to set access restrictions to base you can filter by LDAP attributes.

3.39.1 Access restrictions

Restrictions can be defined given a base and the value of an LDAP attribute. For example, you could set that only users who have the attribute “typology” as “teacher” are allowed to clone a virtual machine. Or you could deny “student” users access to another base.

3.39.2 Configuration

Actually there is a form to configure the access restrictions in the works. Until it is done you have to set this directly in the database.

Grant access

To grant access add an entry in the table access_ldap_attribute with the id of the base, the name of the attribute, the value of the attribute. The optional field allowed can be used to deny access to a virtual machine.

To remove a grant delete the row from the table access_ldap_attribute.
3.39.3 Examples

Example 1: grant access

Grant access to a virtual machine, only to those users that have typology = teacher.
First you need to know the id of the base virtual machine.

```sql
mysql> select id, name from domains where name = 'mymachine';
+------+--------------------------+
| id   | name                     |
+------+--------------------------+
| 88   | mymachine                |
```

Then add the restriction:

```sql
mysql> insert into access_ldap_attribute (id_domain, attribute, value) VALUES(88, 'typology', 'teacher');
```

Example 2: deny access

Deny access to a virtual machine, to those users that have typology = student.
First you need to know the id of the base virtual machine.

```sql
mysql> select id, name from domains where name = 'mymachine2';
+------+--------------------------+
| id   | name                     |
+------+--------------------------+
| 89   | mymachine2               |
```

Then add the restriction:

```sql
mysql> insert into access_ldap_attribute (id_domain, attribute, value, allowed) VALUES(89, 'typology', 'student', 0);
```

Example 3: remove an access restriction

We have some a restriction to deny access to students we want to remove because we want everybody access to that virtual machine:

```sql
mysql> select * from access_ldap_attribute;
+----+-----------+-----------+---------+--------+
| id | id_domain | attribute | value   | allowed|
+----+-----------+-----------+---------+--------+
| 2  | 88        | typology  | teacher | 1      |
| 3  | 89        | typology  | student | 0      |
mysql> delete from access_ldap_attribute where id=3;
```

3.40 Test Active Directory

This document is a guide to test Active Directory features from Ravada source.
If you want to test the active directory you must create 2 files: one for the ravada configuration parameters, and another one with AD user and password to test.

### 3.40.1 Ravada Conf

Create a file: `/etc/ravada_ad.conf` with these contents:

```yaml
ActiveDirectory:
  host: thehost
  port: 389
  domain: thedomain
  principal: theprincipal whatever it is
```

### 3.40.2 Auth data

It is required a valid user and password to test AD. Put them in the file `/etc/test_ad_data.conf`

```yaml
name: theusername
password: thepassword
```

### 3.40.3 Run the tests

From the source root directory run:

```
$ perl Makefile.PL
$ make && prove -b t/67_user_ad.t
```

### 3.41 Tuning Ravada

Some additional settings to improve your Ravada installation.

#### 3.41.1 IP Conflict

When there are machines that have been hibernated long ago, their ip address may conflict with other that have just been created. That is because the IP addresses have a short life span by default.

**Warning**: TODO Make IP addresses last forever changing the settings of dnsmasq.

### 3.42 Server Monitoring

From VM settings tab you can see the VM system overview:
And for admin role you can see the server:

Server monitoring is disabled for default. In order to support Ravada server monitoring, you have to do a few steps:

- install netdata on a ravada server
- and enable monitoring in `/etc/rvd_front.conf`
3.42.1 Install my-netdata.io

Follow this steps from my-netdata.io
or execute this on a terminal:

```bash
# bash <(curl -Ss https://my-netdata.io/kickstart-static64.sh)
```

Linux 64bit, pre-built static binary installation for any Linux distro, any kernel version - for Intel/AMD 64bit hosts.

3.42.2 Apache config for netdata with SSL

Enable SSL and proxy in apache:

```bash
# a2enmod proxy_http proxy ssl
# a2ensite default-ssl

In /opt/netdata/etc/netdata/netdata.conf add:

```bash
[web]
default port = 19998
```

In /etc/apache2/ports.conf add:

```bash
<IfModule ssl_module>
   Listen 443
   Listen 19999
</IfModule>
```

and adding a new virtualhost for port 19999 in /etc/apache2/sites-available/default-ssl.conf

```bash
<VirtualHost *:19999>
   ProxyRequests Off
   ProxyPreserveHost On
   ProxyPass / http://localhost:19998/ keepalive=On
   ProxyPassReverse / http://localhost:19998/

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

   SSLEngine on
   SSLCertificateFile /etc/ssl/certs/server.crt
   SSLCertificateKeyFile /etc/ssl/private/server.key
   SSLCertificateChainFile /etc/ssl/certs/ca.crt
</VirtualHost>
```

**Warning:** Be careful with self-signed certificates. The browser needs to accept the certificate. We recommend the use of Let’s Encrypt or your trusted SSL provider.

Then restart netdata and apache:

```bash
# systemctl restart apache2
# systemctl restart netdata
```

Thanks to @jlopezramos for this contribution.
3.42.3 Enable monitoring

Monitoring is disabled by default. Add in `/etc/rvd_front.conf` file:

```
monitoring => 1
```

Restart `rvd_front` service:

```
# systemctl restart rvd_front.service
```

3.42.4 Tuning netdata

Disable mail alarms

Edit the file `/opt/netdata/etc/netdata/health_alarm_notify.conf`, `/usr/lib/netdata/conf.d/health_alarm_notify.conf` or `/etc/netdata/health_alarm_notify.conf` and set

```plaintext
SEND_MAIL="NO"
```

3.42.5 Monitoring Architecture

All servers have a local My-netdata installation, this lets us know what happens now. Metrics in real time, small story. The file of metrics is stored in the Graphite server, this allows us to obtain historical and archived data. And Grafana lets us know what happened in the past. Low detail metric, long history. Follow this link to known more about it.

3.42.6 Graphite backend

Edit the file `/opt/netdata/etc/netdata/netdata.conf`:

```plaintext
[backend]
    host tags =
    enabled = yes
    data source = average
    type = graphite
    destination = <GraphiteServer>
    prefix = netdata
    hostname = <hostname>
    update every = 10
    buffer on failures = 10
    timeout ms = 20000
    send names instead of ids = yes
    send charts matching = *
    send hosts matching = localhost *
```

3.42.7 Graphana

This dashboard makes our lives easier.
3.43 Guide button step-by-step

The guide button offers step-by-step help. You can see a taste in the following screenshots:

Guide button is disabled for default. In order to support you have to enable guide in `/etc/rvd_front.conf`

3.43.1 Enable guide

In `/etc/rvd_front.conf`:

```
guide => 1
```
Restart rvd_front service:

```
systemctl restart rvd_front.service
```

### 3.43.2 Custom guide

Add path in `/etc/rvd_front.conf`:

```
guide_custom => '/js/custom/custom_intro.js'
```

### 3.44 Local JS and CSS files instead CDN

Some users behind firewalls may experience rendering problems. By default we use CDN versions of the JavaScript and CSS libraries. In some specific cases it may be useful to have those libraries in the same server as the Ravada web frontend runs.

Since Ravada 0.5 release we package the required javascript and CSS files. You can enable the local copy setting the file `/etc/rvd_front.conf` in your host:

```
fallback => 1
```

and restart the rvd_front.service to apply changes.

```
# systemctl restart rvd_front.service
```

Refresh your browser cache and now Ravada use JS and CSS downloaded from your own server.

### 3.45 Exposing a Virtual Machine

By default the virtual machines are created inside a private internal network. This allows the user to reach internet but no connections from outside are allowed.

Some times we may want to install a server in the Virtual Machine and grant access to it. There are many ways to expose a Virtual Machine and allow access to it from outside. Here we describe a few: set a public IP address, redirect with IPTables and HTTP forwarding.

#### 3.45.1 Setting a public IP

One way to expose the virtual machine is use a public IP instead the private used by default. To do so you have to manually edit the machine definition before creating the base. Change the network settings to `bridge`.

This setting gives the more exposure to the virtual machine, so firewalls and other security measures must be configured.

#### 3.45.2 IPTables redirection

You can redirect a port from the host or a virtual machine acting as gateway to the internal address of the machine you want to expose.

This technique restricts which ports from the internal machine are exposed from outside.
3.45.3 HTTP Forwarding

HTTP forwarding can be configured in a web server in the host to access internal web services from outside.

3.45.4 Expose Ports

Ports from the virtual machine can be exposed to outside with this new feature introduced in release 0.5.

3.46 Exposing Ports from a Virtual Machine

Since release 0.5 ports from the internal virtual machine can be exposed to outside.

3.46.1 Requirements

Ports exposing works with Linux Kernel `iptables` so it requires the host to have it installed in properly configured.

The virtual machine must have a way to tell the host what is the internal IP to the host, so the `qemu guest agent <qemu_ga.html>` must be installed.

3.46.2 Configure Expose

From the virtual machine settings there is a new menu item **Ports**. There you can add new ports to be exposed.

There you must add the number of the internal port to be exposed and an optional name. If you set this port to `restricted` it will only be allowed to the remote client IP.

**Fields**

- **Port**: internal port in the virtual machine
- **Name**: optional short description of the service to expose
- **Restricted**: restrict the port to the remote client if set
3.46.3 Run

When the virtual machine gets started Ravada searches for a free port in the host to expose the port through.

In this example the user can connect to the IP 10.1.36.68 port 5954 to access the SSH server inside the virtual machine.
Ports redirection requires the internal IP of the virtual machine to be active. So it retries for a while trying to redirect until an IP is detected or it fails.

### 3.47 Install Alpine Linux

Alpine is a really small linux distribution. It can be installed in a really tight disk drive with barely no RAM. It is good for testing purposes but it can also be used in production servers.

These are guidelines to install Alpine Linux inside a Ravada KVM Guest.

#### 3.47.1 Base Guest

The guest should have more than 256 MB of RAM. If you are planning to run many services you should create the virtual machine with more memory. You can increase it later if you want to keep it slim.

At least 1GB disk drive is required. A swap partition should also be added when creating the virtual machine.

![Create Alpine Virtual Machine](image)

When the machine is created start it from Admin Tools menu, click on Virtual Machines to see a list. At the right there is a bunch of buttons. Click on view to start and access the virtual machine console.

#### 3.47.2 Login

Once it boots login as root with no password.
3.47.3 Setup

Alpine Linux has a setup script that eases the installation process. These are the default settings we use in a Ravada-KVM Virtual Machine.

Start the setup typing:

```
# setup-alpine
```

**Keyboard**

First choose a keyboard layout and variant.

**Hostname**

Type a hostname for the virtual machine. If you call it with the same name when you created the machine before it would be easier for you:

**Networking**

The setup will detect a network interface, probably eth0. Accept this value. Also you should probably accept using dhcp and answer no to manual network configuration, unless you know what you are doing.

**Password**

Type a new password for the root user.

**Timezone**

Select a timezone, the default UTC won’t do any harm when you are testing, but if you know the actual timezone, type it.
Proxy

If you are downloading packages through a proxy tell it now, you probably should keep it to none.

Mirror

Choose a mirror for Alpine to download packages from. Type f will check your connection and choose the best one, it may take a while.

SSH Server

Answer openssh when asked about an SSH server.

Partitioning

The setup script will ask the disk

```
Available disks are:
- vda
- vdb
Which disk(s) would you like to use? [none] vda
How would you like to use them? [?] sys
WARNING: Erase the above disk(s) and continue [y/N]: y
```

Now the setup process will install minimal applications to start the server.

Installing more software

You should at least install these applications:

```
# apk add qemu-guest-agent acpi
```

3.47.4 Shutdown and restart

Type these commands from the console to restart and shutdown the server:

```
# reboot
# poweroff
```

3.47.5 Advanced Settings

Add a swap partition

If you added a swap volume in the Ravada installation form you must define it later after the Alpine setup script.
Define de partition

The swap device will be probably in /dev/vdb, check first with df it is not mounted already. If it is not shown it is ok.

Using fdisk you should find an empty disk and you must create the partitions like this:

```
localhost:~# fdisk /dev/vdb
Command (m for help): n
Command action
  e extended
  p primary partition (1-4)

Partition number (1-4): 1
First cylinder (1-2080, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-2080, default 2080):
Using default value 2080
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table
```

Create the swap

```
# mkswap /dev/vdb1
```

Test it

Type this to start using the swap space:

```
# swapon -a
```

This command will check the memory utilization, at the bottom there should be now a swap entry:

```
# free

             total    used    free   shared  buffers  cached
Mem:         1031924  48896  983028     100    2884  10964
-/+ buffers/cache:    35048  996876
Swap:        1262140     0   1262140
```

Add it to the system

```
# echo "dev/vdb1 swap swap defaults 0 0" >> /dev/fstab
```

Now the swap space is configured, after rebooting the system it should show typing the `free` command.

3.48 Install Windows 10

These are guidelines to install Windows 10 inside a Ravada KVM Guest.
3.48.1 Base Guest

The guest should have more than 3 GB of RAM. If you are planning to run many services you should create the virtual machine with more memory. You can increase it later if you want to keep it slim.

At least 30GB disk drive is required. A swap partition should also be added when creating the virtual machine.

![Image of virtual machine configuration]

When the machine is created start it from Admin Tools menu, click on Virtual Machines to see a list. At the right there is a bunch of buttons. Click on view to start and access the virtual machine console.

3.48.2 Setup

Follow the usual procedure to install Windows10.

When the installations it’s finished, you need to install:

- qemu-guest agent, see the instructions here: [https://pve.proxmox.com/wiki/Qemu-guest-agent#Windows](https://pve.proxmox.com/wiki/Qemu-guest-agent#Windows)
- make sure that acpi service it’s activated.

3.48.3 Use a swap partition for pagefiles

By default the Windows pagefiles will go to the C: disk drive. That will make the clones partitions grow up too much and fill your host disk drive. It should go to the SWAP volume, probably in your D: or E: drive.

In this link you can see how to move pagefiles to another disk:


3.49 How to extend a Ravada Windows guest’s disk space

### 3.49.1 Expanding a Windows 10 guest

Here we will show how to expand the system partition of a Windows 10 host by 10 GB.

First, retrieve the path to the hard drive file that you want to resize. For a VM named Windows10Slim, we would do the following:

```bash
# virsh dumpxml Windows10Slim
```

Here is our image file:

```xml
<src file='/var/lib/libvirt/images-celerra1/Windows10Slim-vda-UrQ2.img'/>
```

As we want to expand a certain partition, the system one, we must find it first:

```bash
# virt-filesystems --long --parts --blkdevs -h -a /var/lib/libvirt/images-celerra1/Windows10Slim-vda-UrQ2.img
```

The output will look like this:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>MBR</th>
<th>Size</th>
<th>Parent</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/sda1</td>
<td>partition</td>
<td>-</td>
<td>500M</td>
<td>/dev/sda</td>
</tr>
<tr>
<td>/dev/sda2</td>
<td>partition</td>
<td>07</td>
<td>20G</td>
<td>/dev/sda</td>
</tr>
<tr>
<td>/dev/sda</td>
<td>device</td>
<td>-</td>
<td>20G</td>
<td>-</td>
</tr>
</tbody>
</table>

And that means we are going to resize `/dev/sda2` in this example.

Use qemu-img to create a new qcow2 hard drive file. As we want to add 10 GB, the resulting disk will be a 30 GB file:

```bash
# qemu-img create -f qcow2 -o preallocation=metadata /var/lib/libvirt/images.2/Windows10Slim-vda-UrQ3.img 30G
```
Now virt-resize will expand the image into the new file

```bash
```

With virsh we can point the VM to use the newly created image

```bash
# virsh edit Windows10Slim
```

Finally, fix permissions

```bash
# chown libvirt-qemu:kvm /var/lib/libvirt/images.2/Windows10Slim-vda-UrQ3.img
# chmod 600 /var/lib/libvirt/images.2/Windows10Slim-vda-UrQ3.img
```

### 3.50 How to extend a Ravada Linux guest’s disk space

Here we will show how to extend the system partition of a Linux host by 10 GB.

1. Shutdown the virtual machine
2. Consult the hard drive name of the Virtual Machine you want resize:
   ```bash
   # virsh edit VirtualMachineName
   ```
3. Use qemu-resize to increase the image size by 10GB:
   ```bash
   # qemu-img resize /path_to_img_file/VirtualDiskImageName.img +10G
   ```
4. IMPORTANT. Do a backup before continue.
   ```bash
   # cp VirtualDiskImageName.img ./VirtualDiskImageName.img.backup
   ```
5. Now start the Virtual Machine. Open a terminal and type:
   ```bash
   $ sudo fdisk /dev/vda
   ```
   Delete the partition
   :: d
   Create a new partition
   :: n
   Accept all by default and exit saving
   :: w
   7. When it starts in a terminal:
   ```bash
   $ sudo resize2fs /dev/vda1
   ```
   You can check if the disk was increased with the ‘df’ command.
3.51 How to enable KVM virsh console access

3.51.1 From Debian / Ubuntu guest

$ sudo systemctl enable serial-getty@ttyS0.service
$ sudo systemctl start serial-getty@ttyS0.service

3.51.2 From KVM server

```
$ virsh list
Id  Name    State
----------------------------------------------------
1   freebsd running
2   ubuntu-box1 running
3   ubuntu-box2 running
```

Type the following command from KVM host to login to the guest named ubuntu-box1

$ virsh console ubuntu-box1

OR

$ virsh console 2

Use CTRL + 5 to exit the console.

3.52 Reduce the image size after cloning a physical PC

Things to keep in mind when we have a cloned image of Windows from a physical PC.

**Note:** During these tasks, be aware that it affects the performance of the server. Avoid making them on a Ravada server in production.

3.52.1 Check the image format

In the following case you can see that it’s RAW format. Although the extension of the file is qcow2 this obviously does not affect.

```sh
qemu-img info Win7.qcow2
image: Win7.qcow2
file format: raw
virtual size: 90G (96636764160 bytes)
disk size: 90G
```

3.52.2 STEPS TO FOLLOW

1. Convert from RAW (binary) to QCOW2:

```
# qemu-img convert -p -f raw Win7.qcow2 -O qcow2 Win7-QCOW2.qcow2
```

Now verify that the image format is QCOW2, and it’s 26GB smaller.
2. The virt-sparsify command-line tool can be used to make a virtual machine disk (or any disk image) sparse. This is also known as thin-provisioning. Free disk space on the disk image is converted to free space on the host.

```
# virt-sparsify -v Win7-QCOW2.qcow2 Win7-QCOW2-sparsi.qcow2
```

**Note:** The virtual machine must be shutdown before using virt-sparsify. In a worst case scenario, virt-sparsify may require up to twice the virtual size of the source disk image. One for the temporary copy and one for the destination image. If you use the –in-place option, large amounts of temporary space are not needed.

Disk size now is 60G, below you can see that reduce image size in 30GB.

```
qemu-img info Win7-QCOW2-sparsi.qcow2
image: Win7-QCOW2-sparsi.qcow2
file format: qcow2
virtual size: 90G (96636764160 bytes)
disk size: 60G
cluster_size: 65536
```

Format specific information:
- compat: 1.1
- lazy refcounts: false
- refcount bits: 16
- corrupt: false

Now it is advisable to let Windows do a CHKDSK, do not interrupt it. Finally, you need to install the Spice guest-tools. This improves features of the VM, such as the screen settings, it adjusts automatically, etc.

### 3.53 Qemu Guest Agent

#### 3.53.1 Host Qemu Agent Prerequisites

Execute the following commands on your host:

```
$ sudo mkdir -p /var/lib/libvirt/qemu/channel/target
$ sudo chown -R libvirt-qemu:kvm /var/lib/libvirt/qemu/channel
```

And edit the file `/etc/apparmor.d/abstractions/libvirt-qemu` adding the following in the end:

```
$ /var/lib/libvirt/qemu/channel/target/* rw,
```

#### 3.53.2 Guest Agent Installation (VM)

This installation must be done in your guest VM if you want to keep the correct time after hibernate.
Ubuntu and Debian

$ sudo apt install qemu-guest-agent

Fedora

$ dnf install qemu-guest-agent

RedHat and CentOS

$ yum install qemu-guest-agent

Windows

Follow the instructions provided by Linux KVM

3.53.3 For VM’s older than this functionality

If you try to use this function on VM’s created before this function was implemented you must do one thing to make it work, first open the machine xml:

$ virsh edit <name-or-id-of-your-machine>

And add the following inside the ‘devices’ section:

```
<channel type="unix">
  <source mode="bind"/>
  <target type="virtio" name="org.qemu.guest_agent.0"/>
</channel>
```

That’s it, enjoy.

3.54 Set Hostname

The hostname of a virtual machine can be changed on startup. The name of the virtual domain is passed in a smbios string and can be used to rename.

3.54.1 Requirements

This feature is available from release 0.3.4 and requires libvirt-4.6.

Packages

- Ravada: 0.3.4
- libvirt: 4.6
Distributions

This feature has been reported to work with these Linux distributions. Any other distribution with libvirt 4.6 or bigger will work too. Please report if you successfully tested it.

Supported distributions:
- Ubuntu 18.10

3.54.2 Linux

The virtual machine name can be read with dmidecode

```
$ dmidecode | grep hostname | awk -F: ' print $3'
```

To set the hostname you must create a script that runs on startup, this one line should be enough for most cases:

```
$ hostname `dmidecode | grep hostname | awk -F: ' print $3'`
```

Some tools may read the hostname from the config file, set it like this:

```
$ dmidecode | grep hostname | awk -F: ' print $3' | sed -e 's/^ //' > /etc/hostname
```

systemd

If your system supports systemd this script will set the virtual machine name as the hostname on startup. Put the service file in `/lib/systemd/system/sethostname.service`

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

[Unit]
Description=Set Hostname
After=systemd-hostnamed.target syslog.target

[Service]
Type=oneshot
ExecStart=/usr/local/bin/set_hostname.sh
```

This is the script that is launched by the service, it should be in `/usr/local/bin/set_hostname.sh` as specified in the previous file.

```
#!/bin/sh
date >> /var/log/set_hostname.log
hostname=`/usr/sbin/dmidecode | grep hostname | awk '{ print $4}'`
if [ ! -z "$hostname" ]; then
  /bin/hostname $hostname
  /bin/hostname > /etc/hostname
  echo "Found hostname $hostname in dmidecode " >> /var/log/set_hostname.log
else
  echo "Not found hostname in dmidecode " >> /var/log/set_hostname.log
  /usr/sbin/dmidecode >> /var/log/set_hostname.log
fi
```

Type this so the script is executed on startup:

```
$ sudo chmod +x /usr/local/bin/set_hostname.sh
$ sudo systemctl enable sethostname
$ sudo systemctl start sethostname
$ sudo systemctl status sethostname
```

Reboot and check if the hostname is applied. You should find a log file at `/var/log/set_hostname.log`. 
If you Linux system supports `rc.local` just add this lines to it and the hostname will be updated on boot:

```
hostname `dmidecode | grep hostname | awk -F: '{ print $3}'`
hostname > /etc/hostname
```

### 3.54.3 Windows

SMBios information is available in Windows too. The data is stored in the registry and also can be shown with a tool called WMI.

Contributed information would be appreciated.

### 3.55 Development Tools

We’re proud to program it in Perl based on TDD. Perl 5 is a highly capable, feature-rich programming language with over 30 years of development. More about why we love Perl….

We use Mojolicious, a real-time web framework. We use MySQL. It’s the world’s most popular open source database. With its proven performance, reliability, and ease-of-use.

We use a lot of powerful free source like GNU/Linux Ubuntu, KVM or Spice, among others. Responsive web made with Bootstrap and AngularJS.

We use Transifex to provide a cleaner and easy to use interface for translators. It’s meant for adapting applications and text to enable their usability in a particular cultural or linguistic market.

Then we build documentation and host it in Read the Docs for you.

### 3.56 Commit Rules

#### 3.56.1 Main Branches

The main branches are master and develop as described here:

http://nvie.com/posts/a-successful-git-branching-model/

#### 3.56.2 Issues

Please create a new branch for each issue. Also it would be a good idea to call the branch with the number of the issue and a short text, ie:

```
$ git checkout -b 77_crashonstart
```

#### 3.56.3 Commit Message

We use conventional commits guideline as specified in https://conventionalcommits.org/ Each commit must be for a reason, and we should have an issue for that, so we decided to add the issue number in the footer.

Definition:
Example:

**fix:** active virtual machines can **not** be started

When a virtual machine **is** already active, do **not try** to start it **and return**

#77

### 3.56.4 Show the branch in the message

Add the file `prepare-commit-msg` to the directory `.git/hooks/` with this content:

**Note:** Remember to give permission to execute, `chmod a+x prepare-commit-msg`  

```bash
#!/bin/sh

# Automatically adds branch name and branch description to every commit message.

NAME=$(git branch | grep '*' | sed 's/* //')
DESCRIPTION=$(git config branch."$NAME".description)
TEXT=$(cat "$1" | sed '/^#.*/d')

if [ -n "$TEXT" ]
    then
        echo "$NAME": "$(cat "$1" | sed '/^#.*/d')" > "$1"
        if [ -n "$DESCRIPTION" ]
            then
                echo "" >> "$1"
                echo "$DESCRIPTION" >> "$1"
        fi
    else
        echo "Aborting commit due to empty commit message."
        exit 1
    fi
```

### 3.56.5 Testing

Before committing, make sure it passes all the tests. This should be run specially when changing the `master` and `develop` branches. Notice some tests require `root` access, so it must run with `sudo`.

```bash
$ perl Makefile.PL && make && sudo make test
```

If you want to run only one test:

```bash
$ perl Makefile.PL && make && sudo prove -b t/dir/file.t
```

Proper testing requires the Perl Module Test::SQL::Data, available here: https://github.com/frankiejo/Test-SQL-Data
3.56.6 Contribution Guide

Check our contribution guide for more information about this topic.
https://github.com/UPC/ravada/blob/master/CONTRIBUTING.md

3.57 Database Changes

When changing code for this project you may add, remove or modify columns in the SQL database. Those changes must be updated too in the sql directory.

3.57.1 SQL tables

Modify the SQL table definitions in the directory sql/mysql/.

3.57.2 Data

Some tables may require data, place them in the directory sql/data. The files must be called with the insert prefix to the table name. So if you create the new table domaindrivers you have to:

- Create a file at sql/mysql/domaindrivers.sql
- Optionally create a file at sql/data/insert_domaindrivers.sql with the insertions

3.57.3 SQLite

SQLite definitions are used for testing and are created from the MySQL files. Once the mysql file is created, add the new table name to the sql/mysql/Makefile and run make. It requires https://github.com/dumblob/mysql2sqlite

3.57.4 Runtime upgrade

When ravada runs, it can check if the table defition is accurate. Place the code following the examples at the function upgrade_tables in Ravada.pm

Example: To check if the table vms has the field vm_type:

```perl
$self->_upgrade_table('vms','vm_type','char(20) NOT NULL DEFAULT 'KVM');
```

3.58 Editor configuration rules

If you work with Ravada code it will be easier for all of us if you follow a minimum configuration rules.

- expand tabs to spaces
- one tab are 4 spaces
3.58.1 Highlight unwanted spaces

Please, don’t remove unwanted spaces if aren’t yours. Highlight them with these tips: [http://vim.wikia.com/wiki/Highlight_unwanted_spaces](http://vim.wikia.com/wiki/Highlight_unwanted_spaces)

To disable autoremove of trailing spaces in **Atom**: In Atom Preferences->Packages, select the whitespace package. In the whitespace package settings, disable “Remove Trailing Whitespace”.

3.58.2 Vim Example

Set those options in your .vimrc to match ours

```
set tabstop=4
set expandtab
"highlight unwanted spaces
highlight ExtraWhitespace ctermbg=red guibg=red
match ExtraWhitespace /\s\+$/
```

3.59 Local ISO server

3.59.1 ISO Web Server

It is pointless and resource consuming download each time the ISO files from the Internet. Set up a webserver in the main host and let the development virtual ravadas download them from there.

Copy the ISO files

Copy the `.iso` files to the directory `/var/www/html/iso`.

```
$ sudo mkdir /var/www/html/iso
$ sudo cp /var/lib/libvirt/images/*iso /var/www/html/iso
```

3.59.2 Apache

Install Apache

Install apache web server:

```
$ sudo apt-get install apache2
```

Config apache

Configure it so ISOs are downloaded from the storage pool, and only the local virtual network is able to access to it. Edit `/etc/apache2/sites-enabled/000-default.conf` and add:
3.59.3 Change the ISO locations

In the table iso_images there is an entry that states where are located original ISO files, change it.

**From localhost**

If you want to access to the ISO files from localhost change the URL field to this:

```bash
$ mysql -u root -p ravada
mysql> update iso_images set url = 'http://127.0.0.1/iso/';
```

**From Virtual Machines**

If you install ravada in a virtual machine inside the host you have to change the URLs to the virtual address, it will probably be 192.168.1.1, check it is doing

```bash
$ ifconfig virbr0

$ mysql -u root -p ravada
mysql> update iso_images set url = 'http://192.168.122.1/iso/';
```

3.59.4 Try it

Remove the ISO from the storage and from the table

**Remove from the VM storage pool**

```bash
$ sudo rm /var/lib/libvirt/images/*iso
```
Remove the device name from the table

First find out the id of the iso image, then remove it.

```bash
$ mysql -u root -p ravada
mysql> select id,name FROM iso_images;
mysql> update iso_images set device = null where id=9;
```

Restart rvd_back and reload the admin page and verify Ravada won’t download them from Internet the next time you try to install a new machine.

```bash
$ sudo ./bin/rdv_back.pl --debug
```

3.60 Steps to release

3.60.1 Create a branch

Name the branch following the guidelines of semantic versioning [http://semver.org/]:

MAJOR.MINOR.PATCH, increment the:

- MAJOR version when you make incompatible API changes,
- MINOR version when you add functionality in a backwards-compatible manner, and
- PATCH version when you make backwards-compatible bug fixes.

```bash
$ git checkout master
$ git checkout -b release/0.2.2
$ git push --set-upstream origin release/0.2.2
```

3.60.2 Draft

Draft the release

This step should be done at the very beginning of planning. If you already did it, skip it now.

At code -> releases draft a new release

- tag version : v0.2.2
- release title : v0.2.2

3.60.3 Create the milestone

At the issues section, create a milestone. Name it like the tag version: 0.2.2. There must be a way to link it to the tag, I just don’t know how.

3.60.4 Create issues

Assign issues to the milestone
3.60.5 Close

Close the milestone

Check the milestone has no open issues and close it.

3.60.6 Update the authors

```bash
$ git checkout 0.2.2
$ cd templates/bootstrap/
$ ./get_authors.sh
```

It will create a file `authors.html.ep`, review it and commit it.

```bash
$ git commit authors.html.ep
$ cd ../..
```

3.60.7 Update the release number

In Ravada.pm

Modify `lib/Ravada.pm` around line 5:

```perl
our $VERSION = '0.2.5';
```

3.60.8 Modify the Changelog

Check the last issues closed for this milestone and add them to the Changelog file:

```bash
$ git checkout release/0.2.2
$ gvim Changelog.md
$ git commit -a
```

3.60.9 Close the release

Make sure the target is the same as the branch, not the master. Close the release at:

- Close the Milestone
- Publish the Release

3.60.10 Release binary

Debian

Create the `debian` package.

```bash
$ fakeroot ./deb/debianize.pl
$ lintian ravada_0.2.2_all.deb
```
Upload the file to our repo and change the number at:

```
```

```
$ git checkout gh-pages
$ gvim docs/docs/INSTALL.md
$ gvim docs/docs/res/ravada_installer.sh
$ gvim index.html
$ git commit -a
$ git push
```

### 3.60.11 Install it

In a test machine, upgrade ravada following:

```
```

In a fresh machine, install it following the whole process:

```
```

### 3.60.12 Publish

- Tweet it
- Mail it in google group ravada@groups.google.com
- Change the release in branch master README.md

### 3.61 Run Ravada in development mode

Once it is installed, you have to run the two ravada daemons. One is the web frontend and the other one runs as root and manage the virtual machines. It is a good practice run each one in a different terminal:

The web frontend runs with the morbo tool that comes with mojolicious. It auto reloads itself if it detects any change in the source code:

```
~/$ src/ravada$ morbo -v ./rvd_front.pl
```

The backend runs as root because it has to deal with the VM processes. It won’t reload automatically when there is a change, so it has to be restarted manually when the code is modified:

```
~/$ src/ravada$ sudo ./bin/rvd_back.pl --debug
```

### 3.61.1 Stop system Ravada

You may have another copy of Ravada if you installed the package release. rvd_back will complain if it finds there is another daemon running. Stop it with:

```
$ sudo systemctl stop rvd_back; sudo systemctl stop rvd_front
```
3.61.2 Keep the library up to date

If you change of branch you may have old libraries running, clean it up from the ravada source directory with:

```bash
~/src/ravada$ sudo make clean
```

3.62 Testing environment

Previously install TEST::SQL::DATA module.

In project root run:

```bash
$ perl Makefile.PL
$ sudo make test
```

At the end, in “Test Summary Report” you can check the result.

If something goes wrong you see: Result: FAIL.

3.62.1 Run a single test

```bash
$ make; sudo prove -b t/lxc/*t
```

3.63 How to create tests

We take great care on crafting a stable product. One of the main keys is creating automated tests to check everything works as expected.

As soon as a problem is found, the very first thing is to be able to reproduce it and then create a test case. Make this test fail, so when the code is fixed it should succeed.

3.63.1 Test Requirements

Tests run on an blank sqlite database that is an exact replica of the real mysql database used in production. The fields and data is the same but the data is empty. So you can run the tests in the same host when a real Ravada service is running.

To ease the process of creating this mock database it is required to install the module Test-SQL-Data.

3.63.2 Test Directory

Create a file in the directory t with the .t extension. There are subdirs there, try to put the file in one of them.

3.63.3 Test File Template

This is an empty tests that does nothing. It just loads a test environment with a blank sqlite database. Notice there is a cleaning of the test environment at the begin and end of the tests. This removes possible old leftovers from failed tests.

Let’s create a simple test that checks if a virtual machine is removed. So we edit a new file called 30_remove.t in the directory ravada/t.
#!/perl

use strict;
use warnings;
use Test::More;
use Test::SQL::Data;

use lib 't/lib';
use Test::Ravada;

# create the mock database
my $test = Test::SQL::Data->new(config => 't/etc/sql.conf');

# init ravada for testing
init($test->connector);

# clean();
use_ok('Ravada');
clean();
done_testing();

3.63.4 Run

The prove command runs a test file, before you have to prepare the libraries and environment. Tests create remove and manage virtual machines. That requires root access to the system, so sudo should be used to run the tests.

The best thing is to run it this way:

```
$ cd ravada
$ perl Makefile.PL && make && sudo prove -b t/30_remove.t
t/user/30_grant_remove.t .. ok
All tests successful.
Files=1, Tests=1, 2 wallclock secs ( 0.02 usr 0.00 sys + 0.57 cusr 0.10 csys = 0.69 CPU)
Result: PASS
```

3.63.5 Trying the Virtual Managers

It is advisable to run the tests on all the virtual managers known to Ravada. To do so we add a loop that tries to load each of them.

Now the test file is like this:

```perl
#!perl

use strict;
use warnings;
use Test::More;
use Test::SQL::Data;
```
use lib 't/lib';
use Test::Ravada;

my $test = Test::SQL::Data->new(config => 't/etc/sql.conf');
init($test->connector);

##############################################################################
clean();
use_ok('Ravada');

for my $vm_name ( vm_names() ) {
  my $vm;
  eval { $vm = rvd_back->search_vm($vm_name) };
  SKIP: {
    my $msg = "SKIPPED test: No $vm_name VM found ";
    if ($vm && $vm_name =~ /kvm/i && $>) {
      $msg = "SKIPPED: Test must run as root";
      $vm = undef;
    }
    diag($msg) if !$vm;
    skip $msg if !$vm;
    diag("Testing remove on $vm_name");
  }
  clean();
done_testing();
}

We also have a mock virtual manager that does nothing but it is used to test generic virtual machines. It is called the Void VM and it only should be used for testing. So the output of running the test should be like this:

$ perl Makefile.PL && make && sudo prove -b t/30_remove.t
t/user/30_grant_remove.t .. 1/?
# Testing remove on KVM
# Testing remove on Void
t/user/30_grant_remove.t .. ok
All tests successful.

### 3.63.6 Test Example: check machine removal

Now the test is there, let’s make it check something, like if a virtual machine has been removed.

#!/perl
use strict;
use warnings;

(continues on next page)
use Test::More;
use Test::SQL::Data;

use lib 't/lib';
use Test::Ravada;

my $test = Test::SQL::Data->new(config => 't/etc/sql.conf');
init($test->connector);

##############################################################################
sub test_remove {
    my $vm = shift;
    my $domain = create_domain($vm->type);
    # $domain->remove( user_admin );
    my $domain2 = $vm->search_domain( $domain->name );
    ok(!$domain2, "[".$domain->type."] expecting domain already removed");
}
##############################################################################

clean();
use_ok('Ravada');
for my $vm_name ( vm_names() ) {
    my $vm;
    eval { $vm = rvd_back->search_vm($vm_name) }
    SKIP: {
        my $msg = "SKIPPED test: No $vm_name VM found ";
        if ($vm && $vm_name =~ /kvm/i && $>) {
            $msg = "SKIPPED: Test must run as root";
            $vm = undef;
        }
        diag($msg) if !$vm;
        skip $msg if !$vm;
        diag("Testing remove on $vm_name");
        test_remove($vm);
    }
}

clean();
done_testing();

Now let’s run the test:

$ perl Makefile.PL && make && sudo prove -b t/30_remove.t
t/user/30_grant_remove.t .. 1/?

(continues on next page)
# Texting remove on KVM
# Failed test ' [KVM] expecting domain already removed'
# at t/user/30_grant_remove.t line 22.
# Texting remove on Void

# Failed test ' [Void] expecting domain already removed'
# at t/user/30_grant_remove.t line 22.
# Looks like you failed 2 tests of 7.
t/user/30_grant_remove.t .. Dubious, test returned 2 (wstat 512, 0x200)
Failed 2/7 subtests

Test Summary Report
-------------------------------
t/user/30_grant_remove.t (Wstat: 512 Tests: 7 Failed: 2)
   Failed tests: 4, 7

Whoah there! It looks like the test failed, of course, someone commented the line 19 that actually removes the machine. Uncomment it and run the tests again. It should return OK.

## 3.64 Translations

We use Transifex to provide a cleaner and easy to use interface for translators.

New entries must be added in the *en.po* file. Because it is the basis of the other language files.

**Warning:** Please don’t add new entries in other .po files directly. Use Transifex instead. If you want to collaborate, create an Issue and give you access as a translator of the language you want.

The language files are stored here in lib/Ravada/I18N.

Before uploading the changes check if there are repeated msgid. The `msguniq` command should not display any output lines.
3.65 Ravada Documentation

We have 2 kinds of documents for Ravada: MarkDown and reStructuredText.

3.65.1 MarkDown

MarkDown (md) files are the default for github documents. It is used for files stored in the code branches of Ravada. This is a guide for the github markdown flavour.

You can view it before commit with the MarkDown Preview Google Chrome extension.

3.65.2 reStructuredText

reStructuredText (rst) files are used for the advanced documentation of Ravada that you can find in our readthedocs web site.

You can read more about rst files at the docutils rst.

The Google Chrome MarkDown Preview extension is supposed to render rst files too but it lacks some features like remote links. It is better to render those files with RestView.

3.66 Hardening Spice security with TLS

TLS support allows to encrypt all/some of the channels Spice uses for its communication. A separate port is used for the encrypted channels.

3.66.1 Change libvirtd configuration

The certificate must be specified in libvirtd configuration file in /etc/libvirt/qemu.conf.

Uncomment the lines: `spice_listen=“0.0.0.0”, spice_tls=1` and `spice_tls_x509_cert_dir="/etc/pki/libvirt-spice"

```
# SPICE is configured to listen on 127.0.0.1 by default.
# To make it listen on all public interfaces, uncomment
# this next option.
#
# NB, strong recommendation to enable TLS + x509 certificate
# verification when allowing public access
#
spice_listen = "0.0.0.0"
# Enable use of TLS encryption on the SPICE server.
#
# It is necessary to setup CA and issue a server certificate
# before enabling this.
#
spice_tls = 1
# Use of TLS requires that x509 certificates be issued. The
```

(continues on next page)
# default it to keep them in /etc/pki/libvirt-spice. This directory
# must contain

# ca-cert.pem - the CA master certificate
# server-cert.pem - the server certificate signed with ca-cert.pem
# server-key.pem - the server private key
#
# This option allows the certificate directory to be changed.
#
spice_tls_x509_cert_dir = "/etc/pki/libvirt-spice"

## 3.66.2 Add path in Apparmor

You may want to add this path to Apparmor, in some Linux distributions it is not necessary, ie Ubuntu from 18.04.

Add `/etc/pki/libvirt-spice/** r, in` /etc/apparmor.d/abstractions/libvirt-qemu

```bash
# access PKI infrastructure
/etc/pki/libvirt-vnc/** r,
/etc/pki/libvirt-spice/** r,
```

**Note:** Remember restart the services: systemctl restart apparmor.service & systemctl restart libvirtd.service

## 3.66.3 Create self signed certificate

Download and run the `create_cert.sh` script.

```bash
#!/bin/bash
SERVER_IP="$1"
if [ -z "$SERVER_IP" ]; then
  echo "Error, server ip required."
  echo " Usage: $0 ip"
  exit -1
fi

# change the next line
SUBJECT="/C=IL/L=Raanana/O=Red Hat"

SERVER_KEY=server-key.pem

# creating a key for our ca
if [ ! -e ca-key.pem ]; then
  openssl genrsa -des3 -out ca-key.pem 1024
fi

# creating a ca
if [ ! -e ca-cert.pem ]; then
  openssl req -new -x509 -days 1095 -key ca-key.pem -out ca-cert.pem
  -subj "$SUBJECT/CN=my CA"
fi
```

### 3.66. Hardening Spice security with TLS
# create server key
if [ ! -e $SERVER_KEY ]; then
    openssl genrsa -out $SERVER_KEY 1024
fi

# create a certificate signing request (csr)
if [ ! -e server-key.csr ]; then
    openssl req -new -key $SERVER_KEY -out server-key.csr -subj "$SUBJECT/CN=$SERVER_IP";
fi

# signing our server certificate with this ca
if [ ! -e server-cert.pem ]; then
    openssl x509 -req -days 1095 -in server-key.csr -CA ca-cert.pem -CAkey ca-key.pem -set_serial 01 -out server-cert.pem
fi

# now create a key that doesn't require a passphrase
openssl rsa -in $SERVER_KEY -out $SERVER_KEY.insecure
mv $SERVER_KEY $SERVER_KEY.secure
mv $SERVER_KEY.insecure $SERVER_KEY

# copy *.pem file to /etc/pki/libvirt-spice
if [ ! -d "/etc/pki/libvirt-spice" ]
then
    mkdir -p /etc/pki/libvirt-spice
fi

`cp ./*.pem /etc/pki/libvirt-spice
chown :kvm /etc/pki/libvirt-spice/*pem
chmod g+rx /etc/pki/libvirt-spice/*pem`

# echo --host-subject
`echo "your --host-subject is" ` "openssl x509 -noout -text -in server-cert.*pem | grep Subject: | cut -f 10- -d " " `

**Warning:** Whatever method you use to generate the certificate and key files, the Common Name value used for the server and client certificates/keys must each differ from the Common Name value used for the CA certificate. Otherwise, the certificate and key files will not work for servers compiled using OpenSSL.

### 3.66.4 Disable Spice Password

More information about removing SPICE password for all the networks.