ravada Documentation

Release latest

Oct 06, 2017

User Documentation

The chances are you're here because you've been searching for free Virtual Desktop Infraestructure (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 an open-source project that allows users to connect to a virtual desktop. So it is a VDI broker.

CHAPTER 1

Ravada delivers

By following the documentation and editing some configuration files, you'll be able to deploy a VM within minutes.

CHAPTER 2

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.

CHAPTER 3

Ravada VDI documentation

The main documentation for the site is divided into three main sections:

- *User Documentation*
- *Feature Documentation*
- *About Ravada*

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

- *Developer Documentation*

Install Ravada

Requirements

OS

Ravada has been successfully tested only on Ubuntu 16.10 and 17.04. It should also work in recent RedHat based systems. Follow this guide if you prefer Debian Jessie.

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.

Install Ravada

Ubuntu

We provide *deb* Ubuntu packages. Download it from the UPC ETSETB repository. There we have also development releases. Try them if you want to help us testing new features.

```
$ wget http://infoteleco.upc.edu/img/debian/libmojolicious-plugin-renderfile-perl_0.
↪10-1_all.deb
$ wget http://infoteleco.upc.edu/img/debian/ravada_0.2.8.1_all.deb
$ sudo dpkg -i libmojolicious-plugin-renderfile-perl_0.10-1_all.deb
$ sudo dpkg -i ravada_0.2.8.1_all.deb
```

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

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

Development Release

Read Development Release if you want to develop Ravada or install a bleeding edge, non-packaged, release.

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.

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

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.

Create the database:

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

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

```
$ mysql -u root -p ravada -e "grant all on ravada.* to rvd_user@'localhost'␣
↪identified by 'CHOOSE A PASSWORD'"
```

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: THE PASSWORD CHOSEN BEFORE
```

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
```

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 domnains. 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
```

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.

Run

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

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.

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

Configuration (Optional)

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' ]
```

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
```

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.

```
# apt-get install apache2
```

In order to make ravada use apache, you must follow the steps explained on here.

Firewall

Ravada uses `iptables` to restrict the access to the virtual machines. These iptables rules grants acess 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
```

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.

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.

Warning: Don't do this if you install a packaged release.

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

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.

Warning: Do not consider VirtualBox in this situation, because it doesn't pass VT-X / AMD-V to the guest operating system.

Ubuntu required packages

These are the Ubuntu required packages. It is is only necessary for the development release.

```
$ sudo apt-get install libmojolicious-perl  mysql-server libauthen-passphrase-perl
↪libdbd-mysql-perl libdbi-perl libdbix-connector-perl libipc-run3-perl libnet-ldap-
↪perl libproc-pid-file-perl libvirt-bin libsys-virt-perl libxml-libxml-perl
↪libconfig-yaml-perl libmoose-perl libjson-xs-perl qemu-utils perlmagick libmoosex-
↪types-netaddr-ip-perl libsys-statistics-linux-perl libio-interface-perl libiptables-
↪chainmgr-perl libnet-dns-perl wget liblocale-maketext-lexicon-perl libmojolicious-
↪plugin-i18n-perl libdbd-sqlite3-perl net-tools
```

- libmojolicious-perl
- mysql-server
- libauthen-passphrase-perl
- libdbd-mysql-perl
- libdbi-perl
- libdbix-connector-perl
- libipc-run3-perl
- libnet-ldap-perl
- libproc-pid-file-perl
- libvirt-bin
- libsys-virt-perl
- libxml-libxml-perl
- libconfig-yaml-perl
- libmoose-perl
- libjson-xs-perl
- qemu-utils
- perlmagick
- libmoosex-types-netaddr-ip-perl
- libsys-statistics-linux-perl
- libio-interface-perl
- libiptables-chainmgr-perl
- libnet-dns-perl

- wget
- liblocale-maketext-lexicon-perl
- libmojolicious-plugin-i18n-perl
- net-tools

Config file

When developping 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
```

Mysql Database

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 'figure a password
↪';
exit
```

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: *****
```

Ravada web user

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

```
$ cd ravada
$ sudo /usr/sbin/rvd_back --add-user user.name
```

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. 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

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.

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

Application directory

The ravada application should be installed in `/var/www/ravada`

Ravada system user

The frontend daemon must run as a non-privileged user.

```
# useradd ravada
```

Allow it to write to some diretories inside `/var/www/ravada/`

```
# mkdir /var/www/ravada/log
# chown ravada /var/www/ravada/log
# chgrp ravada /etc/ravada.conf
# chmod g+r /etc/ravada.conf
# mkdir -p /var/www/img/screenshots/
# chown ravada /var/www/img/screenshots
```

Apache

It is advised to run an apache server or similar before the frontend.

```
# apt-get install apache2
```

Systemd

Configuration for boot start

First you have to copy the service scripts to the systemd directory:

```
$ cd ravada/etc/systemd/
$ sudo cp *service /lib/systemd/system/
```

Edit `/lib/systemd/system/rvd_front.service` and change `User=****` to the `ravada` user just created.

Then enable the services to run at startup

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

Start or stop

```
$ sudo systemctl stop rvd_back
$ sudo systemctl stop rvd_front
```

Other systems

For production mode you must run the front end with a high perfomance server like hypnotoad:

```
$ hypnotoad ./rvd_front.pl
```

And the backend must run from root

```
# ./bin/rvd_back.pl &
```

Firewall

Ravada uses *iptables* to restrict the access to the virtual machines. Thes iptables rules grants acess 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
```

Read *Developer Documentation* to learn how to start it.

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.

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.

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.

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
```

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
```

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
```

And that's it, now Ravada will use the pool that has more empty space the next time it needs to create a volume.

Run Hypnotoad service and Apache as a proxy for it.

Enable apache modules

```
# a2enmod ssl proxy proxy_http proxy_connect
```

Apache Proxy Configuration

Link the https configuration and add the proxy lines.

```
# a2ensite default-ssl
```

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

```
<IfModule mod_ssl.c>
    <VirtualHost _default_:443>
        ProxyRequests Off
        ProxyPreserveHost On
        ProxyPass / http://localhost:8081/ keepalive=On
        ProxyPassReverse / http://localhost:8081/
```

Apache redirect to https

Redirect all the connections to https.

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

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

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

How to import a Virtualbox image

Note: In this example we have VirtualBox machine called *EXAMPLE*.

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`,

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

Remove the `swap`, `cdrom` and other disk volumes.

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

```
<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.

```
</disk>
```

Remove also the SWAP image file:

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

Convert the image file

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

DIRECTLY VDI TO QCOW2

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

OR IN TWO STEPS

1. Convert to raw

```
$ 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:

```
$ sudo qemu-img convert -p -f raw EXAMPLE.vdi -O qcow2 /var/lib/libvirt/images/
↪EXAMPLE-vda-id8Q.img
```

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.

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

How to dump a hard drive to Ravada

Introduction

Tools

- VMWare Converter. First you must register in https://my.vmware.com/en/web/vmware/evalcenter?p=converter#tab_register
- External hard disk or something similar, you will need to save about 60GB or 100GB.

Procedure

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

How to extend a Ravada Windows guest's disk space

More info: http://libguestfs.org/virt-resize.1.html#expanding-a-virtual-machine-disk

> **Warning:** Use truncate only for raw image files. For qcow2 files, use qemu-img

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:

```
virsh dumpxml Windows10Slim
```

Here is our image file:

```
<source 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

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

The output will look like this:

```
Name       Type       MBR  Size  Parent
/dev/sda1  partition  -    500M  /dev/sda
/dev/sda2  partition  07   20G   /dev/sda
/dev/sda   device     -    20G   -
```

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

```
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

```
virt-resize --expand /dev/sda2 /var/lib/libvirt/images-celerra1/Windows10Slim-vda-
↪UrQ2.img /var/lib/libvirt/images.2/Windows10Slim-vda-UrQ3.img
```

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

```
virsh edit Windows10Slim
```

Finally, fix permissions

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

How to Install a local LDAP

Install and configure 389-ds

```
$ sudo apt-get install 389-ds-base
$ sudo setup-ds
```

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: thepasswordyouusedwhensetup-ds
    base: 'dc=telecom,dc=bcn'
```

Insert one test user

The ravada backend script allows creating users in the LDAP

```
$ sudo ./bin/ldap_admin.pl --add-user-ldap jimmy.mcnulty
```

How to add a KVM template

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

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, device)
        VALUES ('name','the description', 'i386', 'name.xml' ,'name-vol.xml',
↪'bbblamd5sumjustgenerated','/var/lib/libvirt/images/file.iso');
```

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 down dump the xml with

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

XML Volume file

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

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', 'xenial64-amd64.xml' ,'xenial64-
↪volume.xml','http://ftp.cixug.es/mint/linuxmint.com/testing/linuxmint-18.2-cinnamon-
↪64bit-beta.iso', 'https://ftp.heanet.ie/mirrors/linuxmint.com/testing/sha256sum.txt
↪');
```

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).

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
```

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

Ravada may also detect ISOs from ISO directories from your directory /home.

If you want to include a KVM instead of an ISO, use this guide instead.

Operation

Note: Must run from the command line interface (CLI)

Create users

```
sudo ./usr/sbin/rvd_back --add-user=username

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

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.

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
```

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 missmatch

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.

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.

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.

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.

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*.

Linux

It is reccommended keep the swapping 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 minimun this way:

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

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

Troubleshooting frequent problems

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
```

Dealing with permissions

The system may deny access to some directories.

On Screnshots (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,
```

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(),  >>>>>>  date_send timestamp default now(),
```

More information about.

Spice-Warning Error in certificate chain verification

(/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%spicecspice_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.

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 Ravad 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.

Update your Ravada

In order to update your ravada, you have to do a few steps from the install and production documents that we will show here.

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>
```

Step 3

Reconfigurate the systemd.

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

Step 4

Restart the services.

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

Windows SPICE Clients

Download Virt Viewer

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

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"=""

[HKEY_CLASSES_ROOT\spice\DefaultIcon]
@="C:\\Program Files\\VirtViewer v5.0-256\\bin\\remote-viewer.exe,1"

[HKEY_CLASSES_ROOT\spice\Extensions]

[HKEY_CLASSES_ROOT\spice\shell]
@="open"

[HKEY_CLASSES_ROOT\spice\shell\open]


[HKEY_CLASSES_ROOT\spice\shell\open\command]
@="\"C:\\Program Files\\VirtViewer v5.0-256\\bin\\remote-viewer.exe\" \"%1\""
```

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/

How to enable KVM virsh console access

From Debian / Ubuntu guest

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

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.

Adding Custom Messages

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

Configuration

Configure the file rvd_front.conf like this:

```
{
      login_header => 'Login'
    ,login_message => 'Login to Start a Machine'
}
```

Create a custom login template

If you need custom login template create one and save it in `templates/main/custom`, e.g. `custom\_login.html.ep`

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'
```

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.

Restart frontend

Finally restart rvd_front:

```
$ sudo systemctl restart rvd_front
```

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.

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.

Sidebar

The organization of the sidebar is configured in the `index.rst`.

Convert POD files

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

Configure LDAP Authentication

Ravada can use LDAP as the authencation engine.

Example: All users

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

```
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
```

Example: Group of users

Allow only a group of users to access ravada:

```
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
```

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. 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.

```
$ msguniq --repeated en.po
```

If you need to work directly in .po file, it's a good option use an application like Poedit.

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 29 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.

Commit Rules

Main Branches

The main branches are *master* and *develop* as described here:

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

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
```

Commit Message

All the commits come from an issue, so add it at the very beggining of the message with brackets , a dash, and the number of the issue. Example:

```
[#44] Fixed flux capacitor leak
```

More guidelines for commit messages here: http://chris.beams.io/posts/git-commit/

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`

```
#!/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
```

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*.

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

If you want to run only one test:

```
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/frankiejol/Test-SQL-Data

Contribution Guide

Check our contribution guide for more information about this topic.

https://github.com/UPC/ravada/blob/master/CONTRIBUTING.md

Database Changes

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

SQL tables

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

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

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

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:

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

Editor configuration rules

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

- expand tabs to spaces
- one tab are 4 spaces

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

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".

Vim Example

Set those options in your .vimrc to match ours

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

Local ISO server

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
```

Apache

Install Apache

Install apache web server:

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

Config apache

Configure it so ISOs are donwloaded 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:

```
<Location /iso>

    Options FollowSymLinks
    AllowOverride None

    Allow from localhost
    Allow from 192.168.122.0/24
    Deny from all

    Require all granted
    Options +Indexes

</Location>
```

Restart apache

```
$ sudo systemctl restart apache2
```

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:

```
$ 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

```
$ ifconfig virbr0


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

Try it

Remove the ISO from the storage and from the table

Remove from the VM storage pool

```
$ 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.

```
$ 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.

```
$ sudo ./bin/rvd_back.pl --debug
```

Steps to release

Create a branch

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

Draft

Draft the release

At code -> releases draft a new release

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

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.

Create issues

Assign issues to the milestone

Close

Close the milestone

Check the milestone has no open issues and close it.

Update the authors

```
$ git checkout master
$ cd templates/bootstrap/
$ ./get_authors.sh
```

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

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

Update the release number

In Ravada.pm

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

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

Modify the Changelog

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

```
$ git checkout master
$ gvim Changelog.md
$ git commit -a
```

Close the release

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

- Close the Milestone
- Close the Release

Release binary

Debian

Create the *debian* package.

```
$ fakeroot ./deb/debianize.pl
$ lintian ravaa_0.2.2_all.deb
```

Upload the file to our repo and change the number at:

```
https://github.com/UPC/ravada/blob/master/docs/INSTALL.md

$ git checkout master
$ git merge v0.2.5
$ gvim docs/INSTALL.md
$ git commit -a
$ git push
```

Publish

- Tweet it
- Change the release in gh-pages

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
```

Testing environment

Previously install TEST::SQL::DATA module.

In project root run:

```
$ 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

Run a single test

```
$ make && sudo prove -b t/lxc/*t
```

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.

Change libvirtd configuration

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

Uncomment the lines: *spice_tls = 1* and *spice_tls_x509_cert_dir = "/etc/pki/libvirt-spice"*

```
# 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
# 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"
```

Add path in Apparmor

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

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

Note: Remmember restart the services: `systemctl restart apparmor.service` & `systemctl restart libvirtd.service`

Generate certificates

Perform the following script, to generate the cert files for ssl , and then copy `*.pem` file into `/etc/pkil/libvirt-spice` directory: (source)

```
#!/bin/bash

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 "/C=IL/
↪L=Raanana/O=Red Hat/CN=my CA"
fi
# 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 "/C=IL/L=Raanana/
↪O=Red Hat/CN=my server"
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

# show the results (no other effect)
openssl rsa -noout -text -in $SERVER_KEY
openssl rsa -noout -text -in ca-key.pem
openssl req -noout -text -in server-key.csr
openssl x509 -noout -text -in server-cert.pem
openssl x509 -noout -text -in ca-cert.pem

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

# 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.

Configuration in XML

For example in this VM with id 1, the connection is possible both through TLS and without any encryption:

```
<graphics type='spice' autoport='yes' listen='172.17.0.1' keymap='es'>
```

```
$ virsh domdisplay 1
spice://172.17.0.1:5901?tls-port=5902
```

For example in VM with id 2, you can edit the libvirt graphics node if you want to change that behaviour and only allow connections through TLS:

```
<graphics type='spice' autoport='yes' listen='171.17.0.1' defaultMode='secure'>
```

```
$ virsh domdisplay 2
spice://171.17.0.1?tls-port=5900
```

From command line

With self-signed certificates, it's necessary pass to the client the certificate of the authority which signed the host certificate.

```
SUBJECT=`openssl x509 -noout -text -in /etc/pki/libvirt-spice/server-cert.pem | grep␣
↪Subject: | cut -f 10- -d " "`

remote-viewer --spice-ca-file=/etc/pki/libvirt-spice/ca-cert.pem spice://172.17.0.1?
↪tls-port=5902 "--spice-host-subject=$SUBJECT"
```

Configuration in .vv file

Tip: Use the following command `openssl x509 -noout -text -in server-cert.pem | grep Subject: | cut -f 10- -d " "` to copy in `host-subject=`.

Tip: Use the following command `awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' server-cert.pem` to convert `server-cert.pem` file to a value that can copy in `ca=`.

See this .vv file as an example reproduced below:

```
[virt-viewer]
type=spice
host=172.17.0.1
tls-port=5902
fullscreen=1
title=Acme - Press SHIFT+F12 to exit
enable-usbredir=1
enable-smartcard=0
enable-usb-autoshare=1
delete-this-file=0
usb-filter=-1,-1,-1,-1,0
tls-ciphers=DEFAULT
host-subject=C=XX, L=XXX, O=XXXX, CN=my server
ca=-----BEGIN CERTIFICATE-----
↪\nMIICUDCCAbmgAwIBAgIJAOgNQo8MIorJMA0GSGSIb3DQEBCwUAMEExCzAJBgNV\nBAYTAklMMRAwDgYDVQQHDAdSYWFuYW5hN
↪fLE+rliV/
↪wFMtwYD+7TtDEFDrafQC8Y7Zd1B\nrdBT9VC+orAc9PqpImXJ3pN152P9rvyZvI3OxKkVTkGFQi+9z3M1AmxTp5nmKA\nrazPM
↪YzV3vraynBXp4x65qLdc2yF2A0cCAwEAAaNQME4wHQYDVR0OBBYEFFGm\nvI6T/
↪86+cpQZ7ob3xd0PgCMB8GA1UdIwQYMBaAFFGmvI6T/
↪86+cpQZ7zohb3xd\n0PgCMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQELBQADgYEALG0TBhPTQwXNpUGi\nia/
↪zxdOh0r7mJWeYcRgZ2lZtesCozYyZz9P2CDb5OnZlu75qs6Ws/fjztRLG/
↪0j\n4r51Og212Up+mQ8eaq2Lox7S/7Ao0P8QWgHZNviltSBb3l9eaYpHENZjW9mMB/
↪JH\nYmIRDdTW1bYuXIsinDPBk0OS20=\n-----END CERTIFICATE-----
toggle-fullscreen=shift+f11
release-cursor=shift+f12
secure-attention=ctrl+alt+end
disable-effects=all
;secure-channels=main;inputs;cursor;playback;record;display;usbredir;smartcard
```

More information about