Setup your own openQRM Cloud with KVM on Ubuntu Lucid Lynx

Preface

This HowTo is brought to you by openQRM Enterprise [http://www.openqrm-enterprise.com/]
Document Version : 10.05.2010

openQRM Enterprise GmbH
Berrenrather Straße 188c
50937 Köln / Germany
Telefon : +49 (0) 221 995589-10
Fax : +49 (0) 221 995589-20
Mail : info@openqrm-enterprise.com

What it is about

This HowTo guides you step-by-step through an openQRM Cloud setup on Ubuntu 10.04 aka Lucid Lynx using the KVM Virtualization technology. The only hardware required is 1 physical system which has the VT (Virtualization Technology) available in the CPU and enabled in the BIOS. The technical details of this single-system openQRM Cloud setup are described in the openQRM Enterprise Documentation at http://www.openqrm-enterprise.com/news/details/article/in-depth-documentation-of-openqrm-available.html [http://www.openqrm-enterprise.com/news/details/article/in-depth-documentation-of-openqrm-available.html] section “Configuring a Basic Setup”.

1. Start with a fresh Ubuntu Lucid Lynx installation.

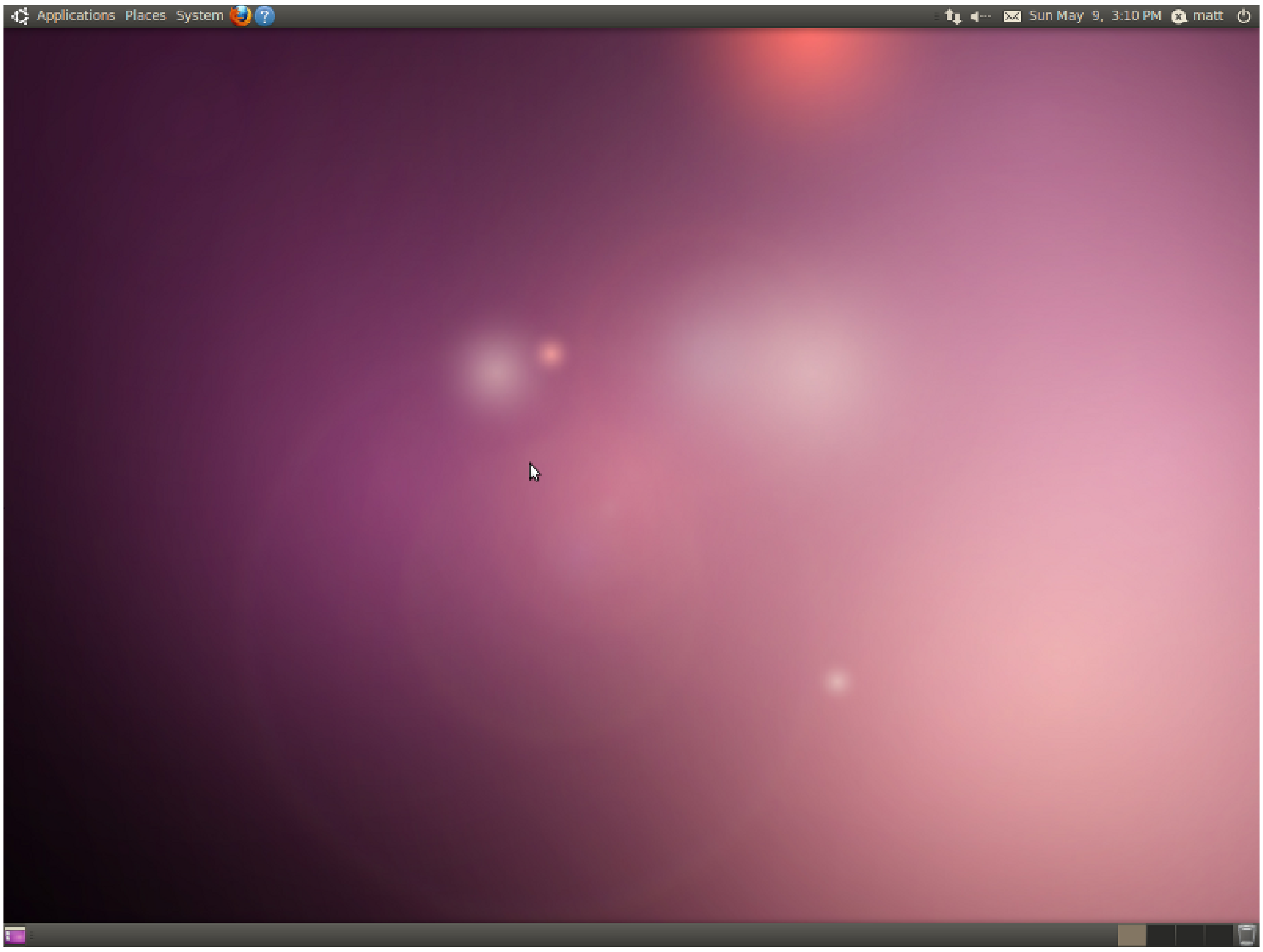
When installing the system with Ubuntu Lucid Lynx select “manual” partitioning. Create 3 Partitions :

1 - primary ext4 mounted at / (the rootfs)
2 - primary swap
3 - primary “do not use” (for the server-image store)

Important is to use a custom partition schema and create a dedicated partition for later storing the server-images. Mark that partition as “do not use” (in this HowTo it will be /dev/sda3). No need to install any extra software packages in the further installation procedure.

When the installation finished reboot and login. If you have selected the Ubuntu-Server version please install the “ubuntu-desktop” package.

```
matt@cloud:~$ sudo apt-get install ubuntu-desktop
```



Ubuntu Lucid Lynx after a fresh installation.

2. Prepare Network

Install “bridge-utils”

```
matt@cloud:~$ sudo apt-get install bridge-utils
Reading package lists... Done
Building dependency tree
...<snip>
Setting up bridge-utils (1.4-5ubuntu2) ...
matt@cloud:~$
```

Now edit /etc/network/interfaces and setup a bridge using a static, private ip-address.

```
matt@cloud:~$ cat /etc/network/interfaces
auto lo
iface lo inet loopback
auto br0
iface br0 inet static
address 192.168.88.3
netmask 255.255.255.0
gateway 192.168.88.1
      bridge_ports eth0
      bridge_fd 0
      bridge_hello 2
      bridge_maxage 12
      bridge_stp off
matt@cloud:~$
```

Then apply the new network configuration by restarting the network.

```
matt@cloud:~$ sudo /etc/init.d/networking restart
* Reconfiguring network interfaces...
Waiting for br0 to get ready (MAXWAIT is 2 seconds).
ssh stop/waiting
ssh start/running, process 2864
matt@cloud:~$
```

Run “brctl show” to check the new network configuration. It should look like below.

```
matt@cloud:~$ brctl show
bridge name    bridge id               STP enabled     interfaces
br0            8000.002215be747a       no              eth0
matt@cloud:~$
```

Now setup the static ip-address (in this HowTo “192.168.88.3”) and hostname in /etc/hosts. Please make sure that the hostname (in this Howto “cloud”) does not appear in the line starting with 127.0.0.1.

```
matt@cloud:~$ cat /etc/hosts
127.0.0.1      localhost
192.168.88.3   cloud.openqrm   cloud
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
matt@cloud:~$
```

3. Prepare Storage for Server-Images

Install lvm2, nfs-kernel-server, iscsi-target and vblade.

```
matt@cloud:~$ sudo apt-get install lvm2 nfs-kernel-server iscsitarget vblade
Reading package lists... Done
Building dependency tree
...<snip>
ldconfig deferred processing now taking place
Processing triggers for initramfs-tools ...
update-initramfs: Generating /boot/initrd.img-2.6.32-21-server
matt@cloud:~$
```

Now prepare the dedicated partition to be used with lvm, then create a logical volume group “vol”.

```
matt@cloud:~$ sudo pvcreate /dev/sda3
Physical volume "/dev/sda3" successfully created
matt@cloud:~$ sudo pvs
PV         VG   Fmt  Attr PSize   PFree
/dev/sda3       lvm2 --   186.23g 186.23g
matt@cloud:~$ sudo vgcreate vol /dev/sda3
Volume group "vol" successfully created
matt@cloud:~$ sudo vgs
VG   #PV #LV #SN Attr   VSize   VFree
vol    1   0   0 wz--n- 186.22g 186.22g
matt@cloud:~$
```

Edit /etc/default/iscsitarget and set the iscsitarget to be started on boot-up.

```
matt@cloud:~$ cat /etc/default/iscsitarget
ISCSITARGET_ENABLE=true
matt@cloud:~$
```

Then start the iscsitarget and nfs-kernel-server services.

```
matt@cloud:~$ sudo /etc/init.d/iscsitarget start
* Starting iSCSI enterprise target service
matt@cloud:~$
```

```
matt@cloud:~$ sudo /etc/init.d/nfs-kernel-server start
* Exporting directories for NFS kernel daemon...
* Starting NFS kernel daemon
matt@cloud:~$
```

4. Prepare the Database

For the openQRM Server Database backend please install “mysql-server”.

```
matt@cloud:~$ sudo apt-get install -y mysql-server
Reading package lists... Done
Building dependency tree
...<snip>
Setting up mysql-server (5.1.41-3ubuntu12) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
matt@cloud:~$
```

For sake of simplicity in this HowTo we have left the myslq-password empty.

5. Prepare KVM

Install the “kvm” package.

```
matt@cloud:~$ sudo apt-get install -y kvm
Reading package lists... Done
Building dependency tree
.....<snip>
Setting up qemu-kvm (0.12.3+noroms-0ubuntu9) ...
qemu-kvm start/running
Setting up kvm (1:84+dfsg-0ubuntu16+0.12.3+noroms+0ubuntu9) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
matt@cloud:~$
```

6. Install openQRM

We build openQRM from the sources which are available in the openQRM Projects subversion repository. The only requirement for that is to have “subversion (client)” and “make” available so please install both components.

```
matt@cloud:~$ sudo apt-get install -y subversion make
Reading package lists... Done
Building dependency tree
...<snip>
Setting up subversion (1.6.6dfsg-2ubuntu1) ...
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
matt@cloud:~$
```

Now checkout the openQRM sources from the svn repository.

```
matt@cloud:~$  svn co https://openqrm.svn.sourceforge.net/svnroot/openqrm openqrm
.... <snip>
matt@cloud:~$
```

Change to the src/ dir.

```
matt@cloud:~$ cd openqrm/trunk/src/
matt@cloud:~/openqrm/trunk/src$
```

And run “make”. Please notice that this step requires a working internet connection. If no internet is available on this system you can download http://sourceforge.net/projects/openqrm/files/openQRM-4.6/source/openqrm-thirdparty-cache.tgz/download [http://sourceforge.net/projects/openqrm/files/openQRM-4.6/source/openqrm-thirdparty-cache.tgz/download] and unzip it in your home directory. This build-cache then will avoid any downloads.

```
matt@cloud:~/openqrm/trunk/src$ make
.... <snip>
```

All compilation results are cached by the openQRM build-system. To ensure all components are build correctly simply run “make” again. The second (and every further “make” run) will just takes a few seconds. Here how the console output looks like for the second “make” run :

```
matt@cloud:~/openqrm/trunk/src$ make
Checking requirements for the compilation phase
openqrm-server requires: make, gcc, portmap, rsync, zlib1g-dev, wget, tar, bzip2, unzip, wget, netbase, patch
found make installed
found gcc installed
found portmap installed
found rsync installed
found zlib1g-dev installed
found wget installed
found tar installed
found bzip2 installed
found unzip installed
found wget installed
found netbase installed
found patch installed
openqrm-plugin-aoe-storage requires:
openqrm-plugin-aws requires:
openqrm-plugin-citrix requires:
openqrm-plugin-cloud requires:
openqrm-plugin-collectd requires:
openqrm-plugin-dhcpd requires:
openqrm-plugin-dns requires:
openqrm-plugin-equallogic-storage requires:
openqrm-plugin-highavailability requires:
openqrm-plugin-image-shelf requires:
openqrm-plugin-iscsi-storage requires:
openqrm-plugin-kvm requires:
openqrm-plugin-kvm-storage requires:
openqrm-plugin-linux-vserver requires:
openqrm-plugin-linuxcoe requires:
openqrm-plugin-local-server requires:
openqrm-plugin-local-storage requires:
openqrm-plugin-lvm-storage requires:
openqrm-plugin-nagios2 requires:
openqrm-plugin-nagios3 requires:
openqrm-plugin-netapp-storage requires:
openqrm-plugin-nfs-storage requires:
openqrm-plugin-puppet requires:
openqrm-plugin-sanboot-storage requires:
openqrm-plugin-solx86 requires:
openqrm-plugin-sshterm requires:
openqrm-plugin-tftpd requires:
```

```
openqrm-plugin-tmpfs-storage requires:
openqrm-plugin-vbox requires:
openqrm-plugin-vmware-esx requires:
openqrm-plugin-vmware-server requires:
openqrm-plugin-vmware-server2 requires:
openqrm-plugin-windows requires:
openqrm-plugin-xen requires:
openqrm-plugin-xen-storage requires:
openqrm-plugin-zabbix requires:
openqrm-plugin-zfs-storage requires:
Checking for required components to compile openQRM finished successfully
if [ -d ./thirdparty ]; then mkdir -p ../buildtmp; cp -aR ./thirdparty/* ../buildtmp/; fi
-> found component kvm-nic-bios (kvm-nic-bios-1.1.tgz) already downloaded
-> found component gpxe (undionly.kpxe.0.9.9.tgz) already downloaded
-> found component sshterm-component (openqrm-plugin-sshterm-components-1.0.tgz) already downloaded
-> found component openqrm-client.windows (openQRM-Client-4.6.1-setup.exe) already downloaded
Creating the default initrd-template
-> found component busybox (busybox-1.14.2.tar.bz2) already downloaded
-> Found busybox-1.14.2/_install/bin/busybox already in the build-cache
-> Skipping compilation, taking the ready built component from the cache
-> found component pciutils (pciutils-3.1.4.tar.gz) already downloaded
-> Found pciutils-3.1.4/pcimodules already in the build-cache
-> Skipping compilation, taking the ready built component from the cache
-> found component dropbear (dropbear-0.52.tar.gz) already downloaded
-> Found dropbear-0.52/dropbear already in the build-cache
-> Skipping compilation, taking the ready built component from the cache
/lib64/ld-2.11.1.so  /lib64/ld-linux-x86-64.so.2
Adding /sbin/portmap to default initrd-template
Adding /sbin/rpc.statd to default initrd-template
Adding /bin/bash to default initrd-template
Adding /usr/bin/rsync to default initrd-template
Adding /usr/bin/wget to default initrd-template
Adding /sbin/modprobe to default initrd-template
Adding /sbin/depmod to default initrd-template
Adding /sbin/insmod to default initrd-template
Adding /sbin/lsmod to default initrd-template
Adding /sbin/mke2fs to default initrd-template
Adding /sbin/sfdisk to default initrd-template
Adding /sbin/udevd to default initrd-template
Adding /sbin/blkid to default initrd-template
/lib64/libnss_files-2.11.1.so  /lib64/libnss_files.so.2
-> found component jquery (jquery-1.3.2.tgz) already downloaded
-> found component js-interface (interface_1.2.zip) already downloaded
-> found component openqrm-client.centos.i386 (openqrm-client.4.6.1.centos.i386.tgz) already downloaded
-> found component openqrm-client.centos.x86_64 (openqrm-client.4.6.1.centos.x86_64.tgz) already downloaded
-> found component openqrm-client.debian.i386 (openqrm-client.4.6.1.debian.i386.tgz) already downloaded
-> found component openqrm-client.debian.x86_64 (openqrm-client.4.6.1.debian.x86_64.tgz) already downloaded
-> found component openqrm-client.ubuntu.i386 (openqrm-client.4.6.1.ubuntu.i386.tgz) already downloaded
-> found component openqrm-client.ubuntu.x86_64 (openqrm-client.4.6.1.ubuntu.x86_64.tgz) already downloaded
-> found component openqrm-initrd-template.centos.i386 (openqrm-initrd-template.4.6.1.centos.i386.tgz) already downloaded
-> found component openqrm-initrd-template.centos.x86_64 (openqrm-initrd-template.4.6.1.centos.x86_64.tgz) already downlo
-> found component openqrm-initrd-template.debian.i386 (openqrm-initrd-template.4.6.1.debian.i386.tgz) already downloaded
-> found component openqrm-initrd-template.debian.x86_64 (openqrm-initrd-template.4.6.1.debian.x86_64.tgz) already downlo
-> found component openqrm-initrd-template.ubuntu.i386 (openqrm-initrd-template.4.6.1.ubuntu.i386.tgz) already downloaded
-> found component openqrm-initrd-template.ubuntu.x86_64 (openqrm-initrd-template.4.6.1.ubuntu.x86_64.tgz) already downlo
-> found component kvm-nic-bios (kvm-nic-bios-1.1.tgz) already downloaded
-> found component gpxe (undionly.kpxe.0.9.9.tgz) already downloaded
-> found component sshterm-component (openqrm-plugin-sshterm-components-1.0.tgz) already downloaded
-> found component openqrm-client.windows (openQRM-Client-4.6.1-setup.exe) already downloaded
matt@cloud:~/openqrm/trunk/src$
```

Then run “sudo make install”.

```
matt@cloud:~/openqrm/trunk/src$ sudo make install
Creating the openqrm-client boot-service package
include/
include/openqrm-plugin-kvm-functions
.... further install output
sbin/
sbin/openqrm-kvm-storage-monitord
matt@cloud:~/openqrm/trunk/src$
```

And finally initialize and start openQRM by “sudo make start”.

```
matt@cloud:~/openqrm/trunk/src$ sudo make start
.... runtime dependency check, automatic install additional requirements
...<snip>
openqrm-plugin-xen requires: , screen
-> found screen installed
openqrm-plugin-xen-storage requires: , screen
-> found screen installed
openqrm-plugin-zabbix requires:
openqrm-plugin-zfs-storage requires: , open-iscsi
-> found open-iscsi installed
Checking for required components finished successfully
First startup detected. Running initialization.

```

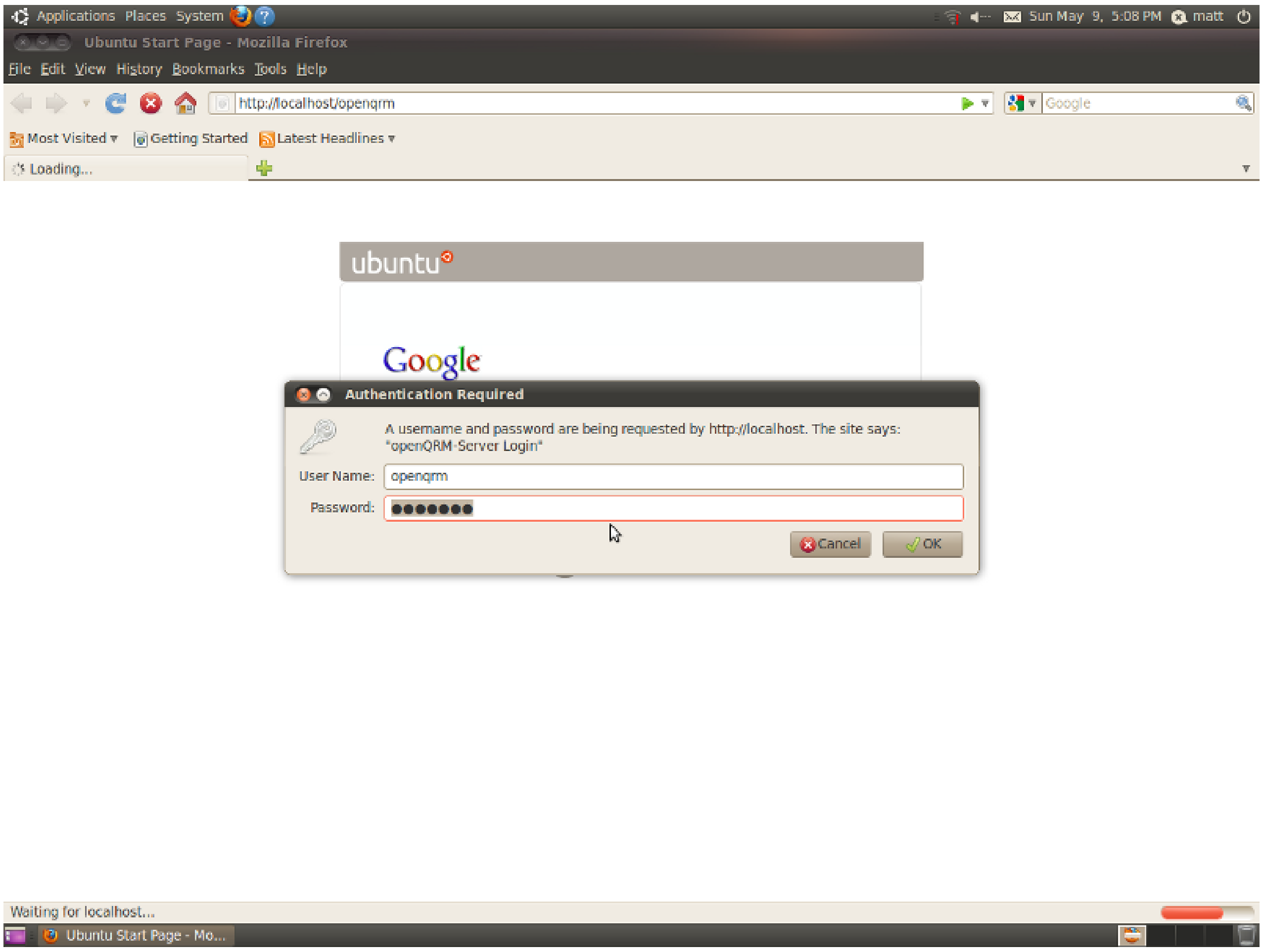
```
Adding system startup for /etc/init.d/openqrm ...
 /etc/rc0.d/K24openqrm -> ../init.d/openqrm
 /etc/rc1.d/K24openqrm -> ../init.d/openqrm
 /etc/rc6.d/K24openqrm -> ../init.d/openqrm
 /etc/rc2.d/S98openqrm -> ../init.d/openqrm
 /etc/rc3.d/S98openqrm -> ../init.d/openqrm
 /etc/rc4.d/S98openqrm -> ../init.d/openqrm
 /etc/rc5.d/S98openqrm -> ../init.d/openqrm
Looking for syslinux/pxelinux.0...found: /usr/lib/syslinux/pxelinux.0
Creating custom apache config.../etc/apache2/conf.d/openqrm-httpd.conf
Checking /usr/share/openqrm/etc/openqrm-server.conf for OPENQRM_WEB_PROTOCOL=https.. * Reloading web server config apache
Adding password for user openqrm
Initializing dropbear...
Will output 1024 bit rsa secret key to '/usr/share/openqrm/etc/dropbear/dropbear_rsa_host_key'
Generating key, this may take a while...
Public key portion is:
ssh-rsa xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxroot@cloud
Fingerprint: md5 28:46:66:b4:b7:59:b6:28:70:ec:2b:6f:16:4a:dd:70
Adding public key to /root/.ssh/authorized_keys...
Starting the openQRM-server ver. 4.6.
Initialization complete.  Please configure your openQRM Server at: http://[server-ip-address]/openqrm/
-> User: openqrm  -> Password: openqrm
matt@cloud:~/openqrm/trunk/src$
```

“make start” triggers a check for the openQRM runtime-dependencies which will install all additional required packages automatically. At first start the openQRM Server is initialized.

We are now ready to configure the openQRM Server via the Web-UI.

7. Configure openQRM

Login to your openQRM Server at http://localhost/openqrm [http://localhost/openqrm]. User and password is “openqrm”. Please make sure to change this default credentials after the configuration phase.

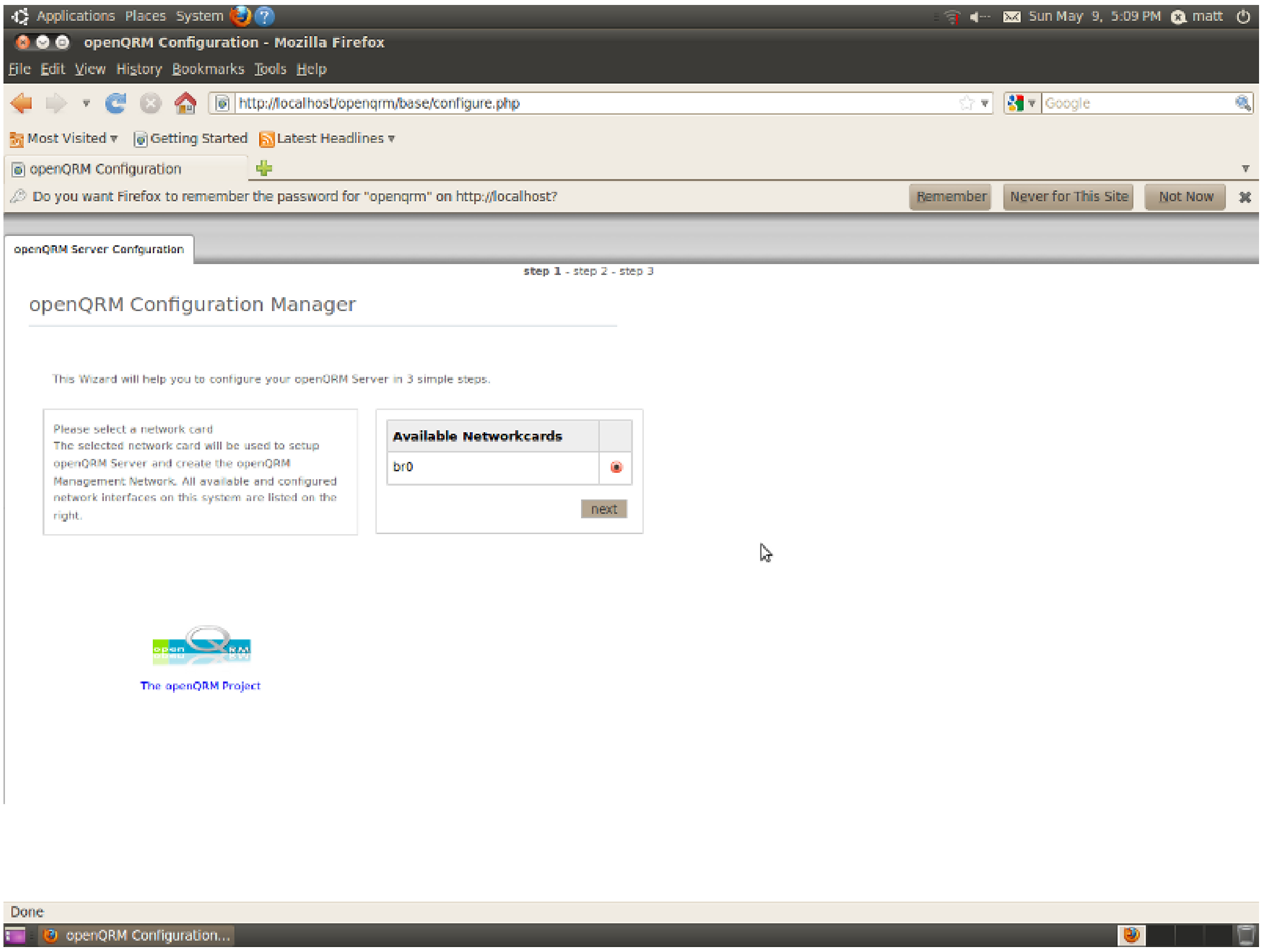


A username and password are being requested by http://localhost. The site says: "openQRM-Server Login"

User Name: openqrm

Password: ●●●●●●●

The step-by-step Web-based configuration guides you through the setup phase. First select the bridge-interface as the openQRM management network-device.


Applications Places System
Sun May 9, 5:09 PM matt
openQRM Configuration - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/openqrm/base/configure.php
Most Visited Getting Started Latest Headlines
openQRM Configuration
Do you want Firefox to remember the password for "openqrm" on http://localhost?
Remember
Never for This Site
Not Now
openQRM Server Configuration
step 1 - step 2 - step 3
openQRM Configuration Manager

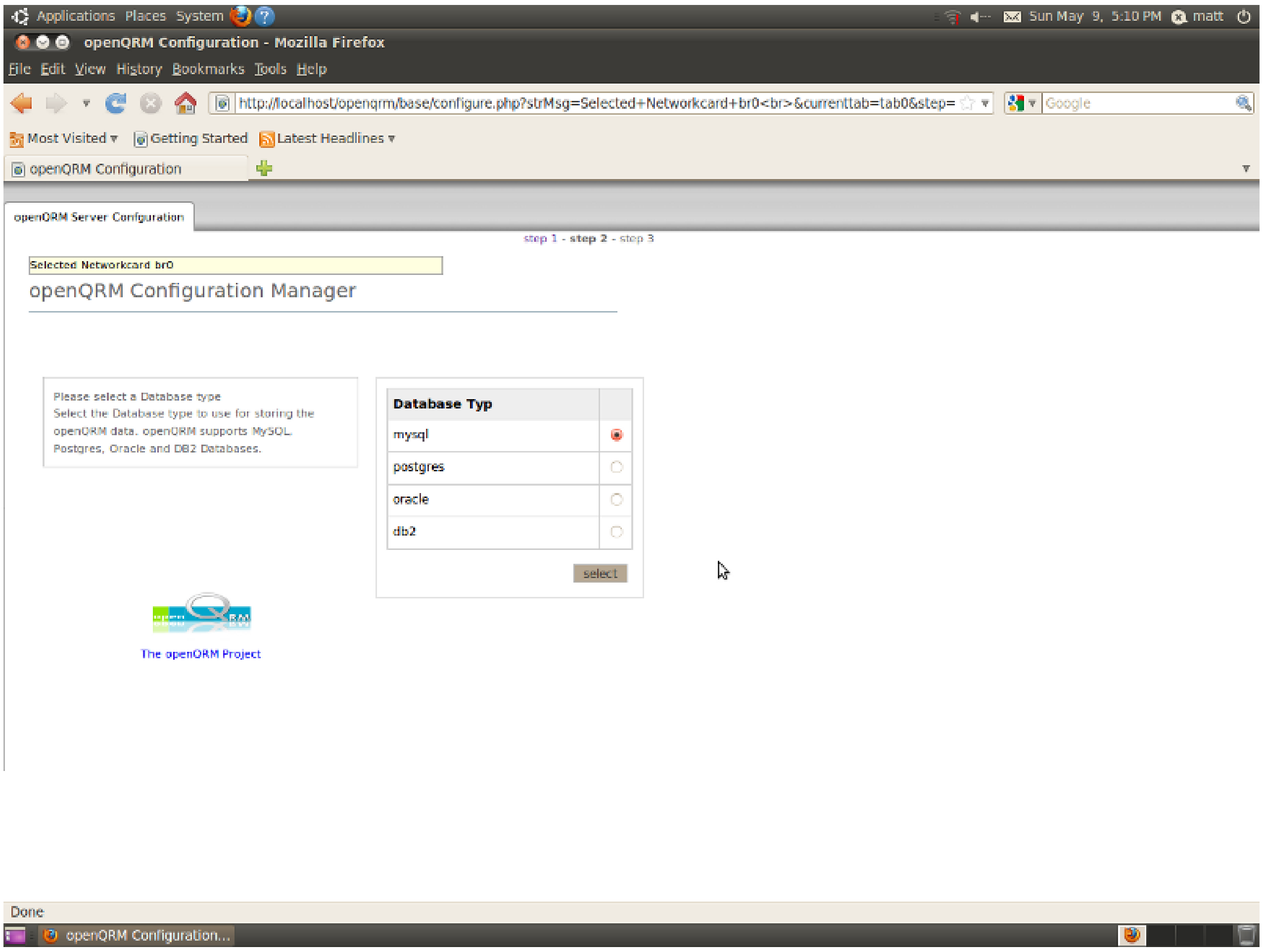
This Wizard will help you to configure your openQRM Server in 3 simple steps.

Please select a network card
The selected network card will be used to setup openQRM Server and create the openQRM Management Network. All available and configured network interfaces on this system are listed on the right.

Available Networkcards	
br0	●

next
The openQRM Project
Done
openQRM Configuration...

Then select “myslq” as the Database to use as the openQRM backend.


Applications Places System
Sun May 9, 5:10 PM matt
openQRM Configuration - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/openqrm/base/configure.php?strMsg=Selected+Networkcard+br0
&currenttab=tab0&step=
Most Visited Getting Started Latest Headlines
openQRM Configuration
openQRM Server Configuration
step 1 - step 2 - step 3

Selected Networkcard br0

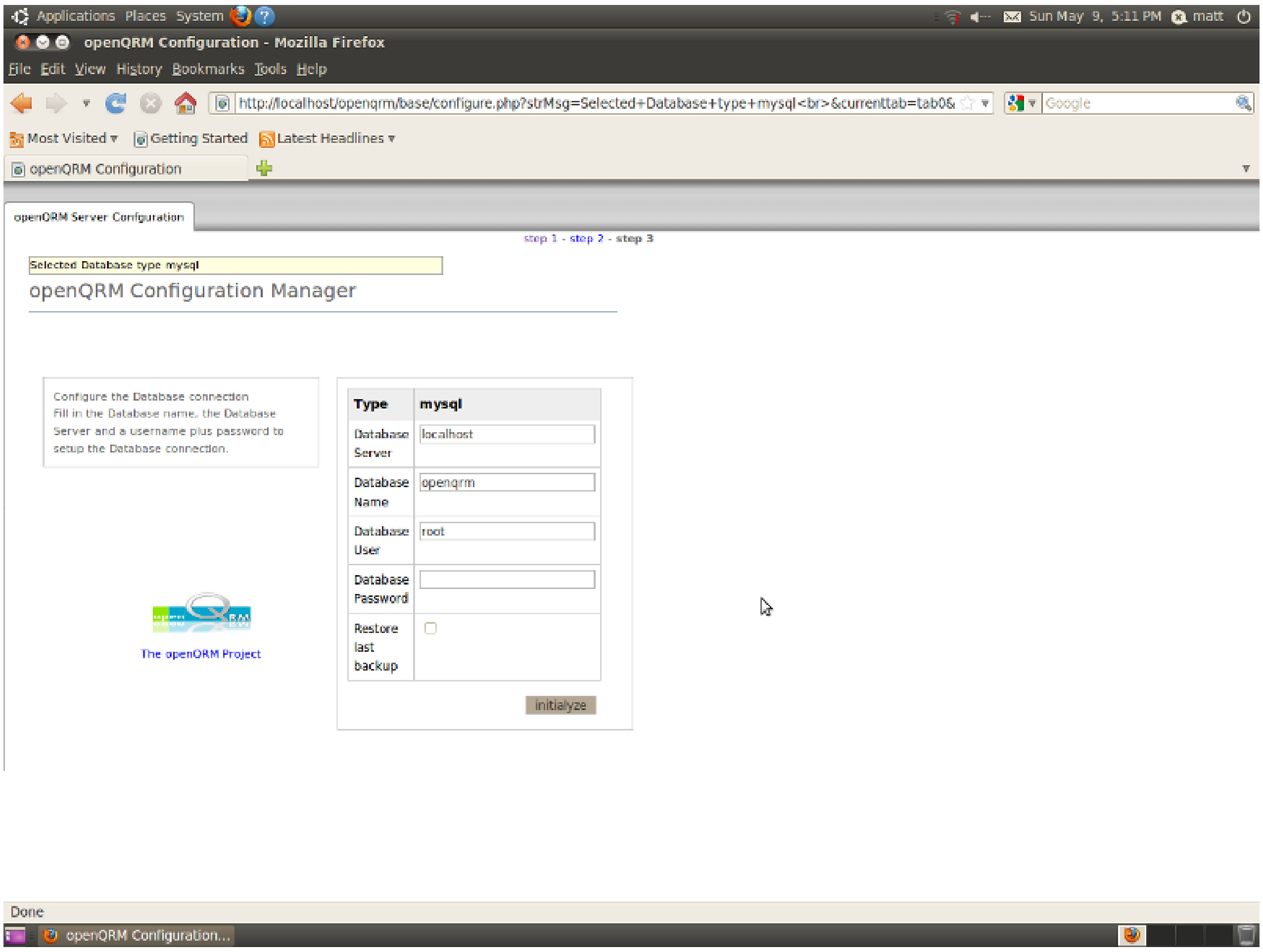
openQRM Configuration Manager

Please select a Database type
Select the Database type to use for storing the openQRM data. openQRM supports MySQL, Postgres, Oracle and DB2 Databases.

Database Typ	
mysql	●
postgres	○
oracle	○
db2	○

select
The openQRM Project
Done
openQRM Configuration...

And then configure the Database connection credentials.



Applications Places System
Sun May 9, 5:11 PM matt
openQRM Configuration - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/openqrm/base/configure.php?strMsg=Selected+Database+type+mysql
&currenttab=tab0&
Most Visited Getting Started Latest Headlines
openQRM Configuration
openQRM Server Configuration
step 1 - step 2 - step 3
The openQRM Project
initialyze
Done
openQRM Configuration...

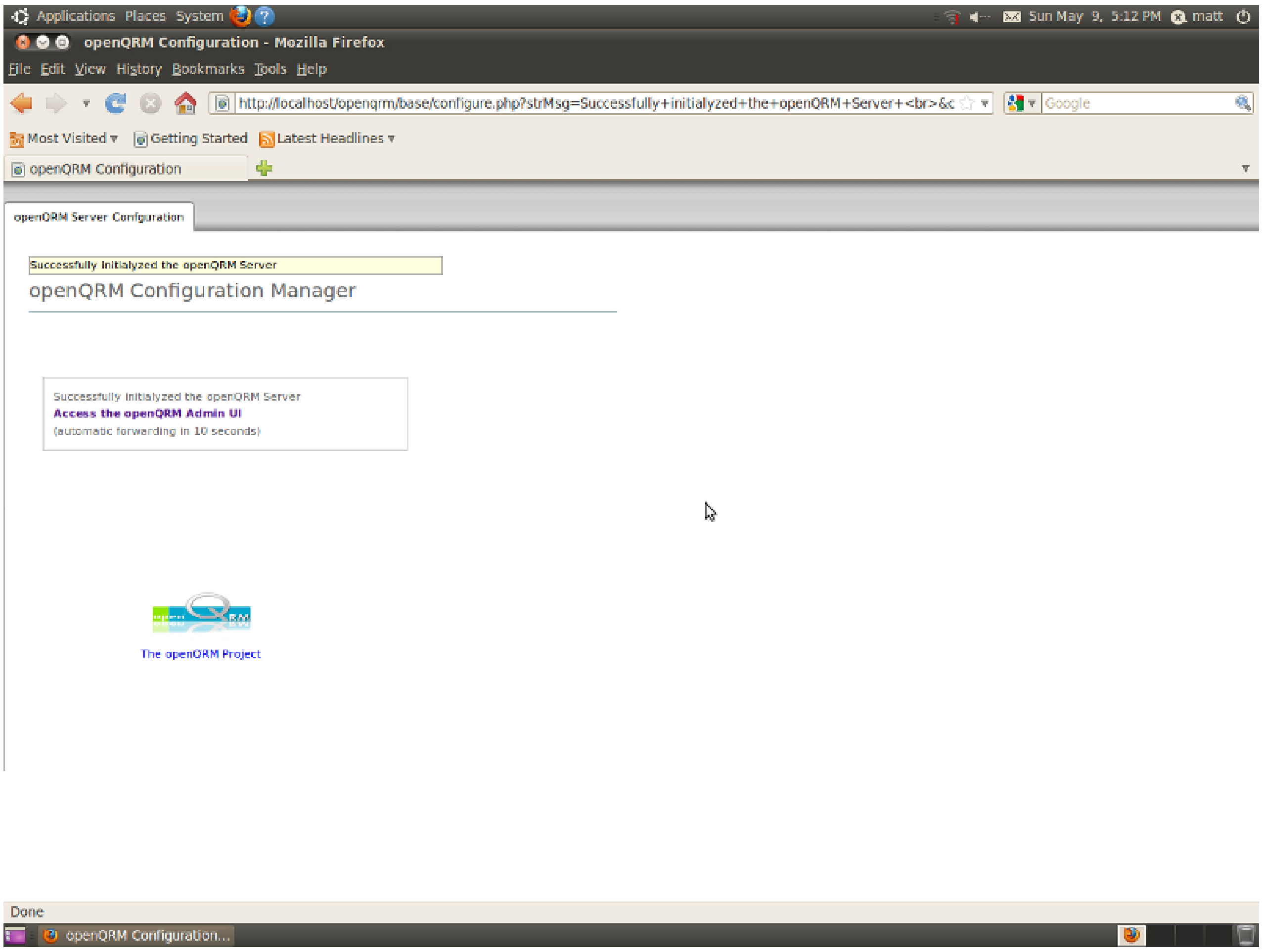
Selected Database type mysql

openQRM Configuration Manager

Configure the Database connection
Fill in the Database name, the Database Server and a username plus password to setup the Database connection.

Type	mysql
Database Server	localhost
Database Name	openqrm
Database User	root
Database Password	
Restore last backup	☐

openQRM is now fully configured and will forward to the Datacenter Dashboard.



Applications Places System
Sun May 9, 5:12 PM matt
openQRM Configuration - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/openqrm/base/configure.php?strMsg=Successfully+initialyzed+the+openQRM+Server+
&c
Google
Most Visited Getting Started Latest Headlines
openQRM Configuration
openQRM Server Configuration

Successfully initialyzed the openQRM Server

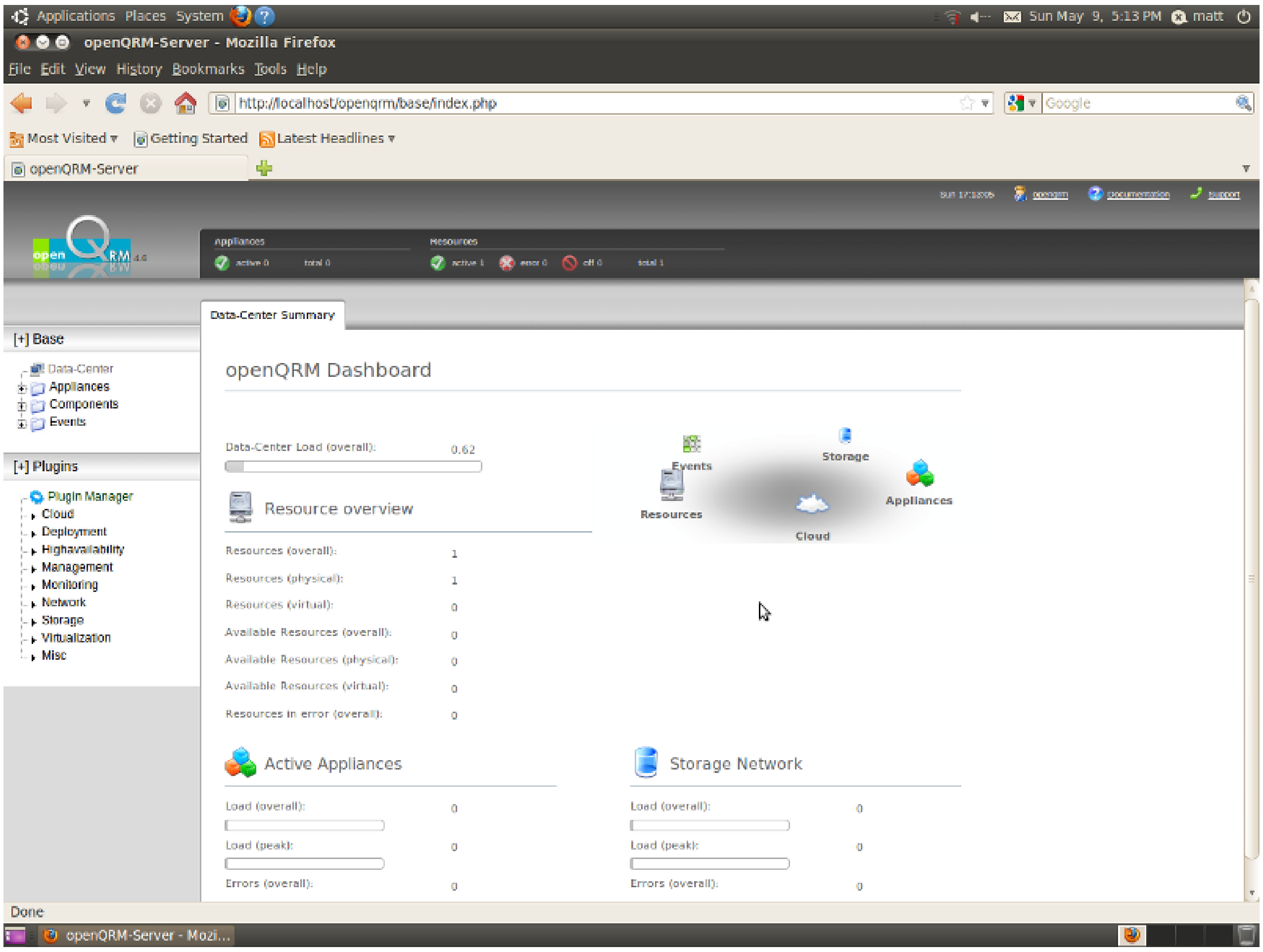
openQRM Configuration Manager

Successfully initialyzed the openQRM Server
Access the openQRM Admin UI
(automatic forwarding in 10 seconds)

The openQRM Project

Done

The openQRM Datacenter Dashboard



openQRM Dashboard

Data-Center Load (overall): 0.62

Resource overview

Resources (overall): 1
Resources (physical): 1
Resources (virtual): 0
Available Resources (overall): 0
Available Resources (physical): 0
Available Resources (virtual): 0
Resources in error (overall): 0

Active Appliances

Load (overall): 0
Load (peak): 0
Errors (overall): 0

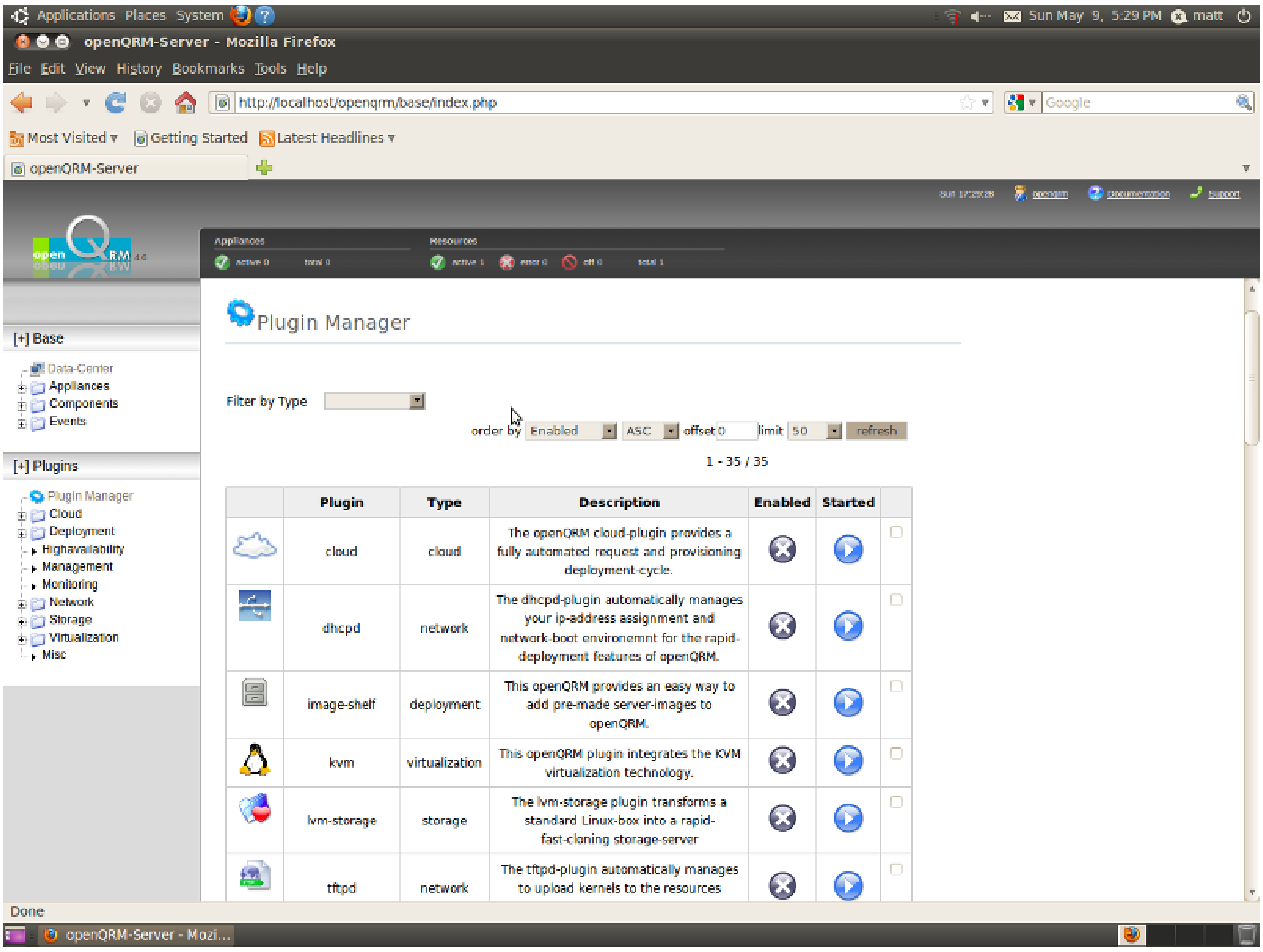
Storage Network

Load (overall): 0
Load (peak): 0
Errors (overall): 0

8. Prepare Server-Images

First step is to enable and start the following plug-ins :

- cloud
- dhcpd
- image-shelf
- kvm
- lvm-storage
- tftpd



Plugin Manager

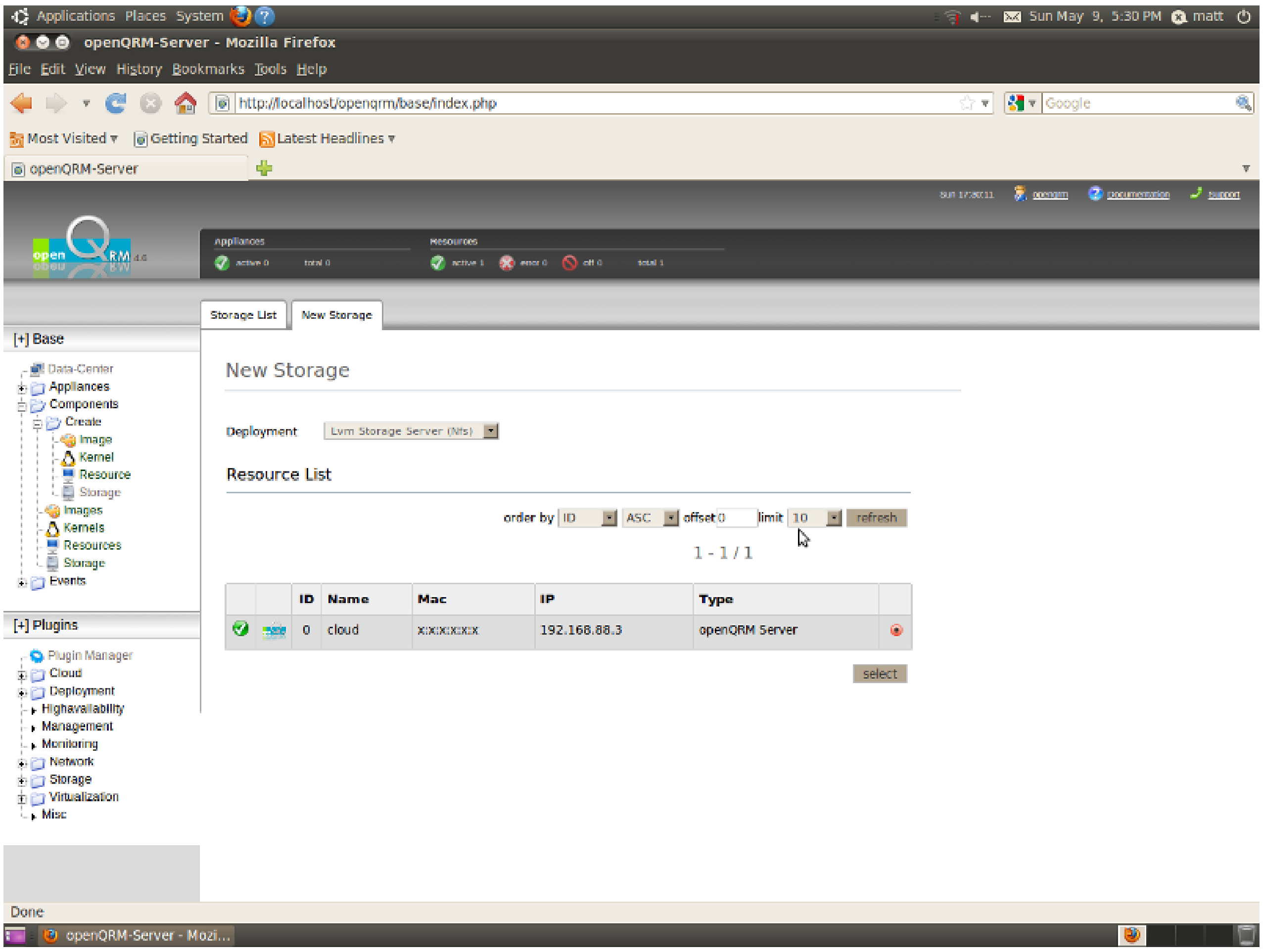
Filter by Type

order by Enabled ASC offset 0 limit 50 refresh

1 - 35 / 35

	Plugin	Type	Description	Enabled	Started	
	cloud	cloud	The openQRM cloud-plugin provides a fully automated request and provisioning deployment-cycle.	⊗	▶	☐
	dhcpd	network	The dhcpd-plugin automatically manages your ip-address assignment and network-boot environemnt for the rapid-deployment features of openQRM.	⊗	▶	☐
	image-shelf	deployment	This openQRM provides an easy way to add pre-made server-images to openQRM.	⊗	▶	☐
	kvm	virtualization	This openQRM plugin integrates the KVM virtualization technology.	⊗	▶	☐
	lvm-storage	storage	The lvm-storage plugin transforms a standard Linux-box into a rapid-fast-cloning storage-server	⊗	▶	☐
	tftpd	network	The tftpd-plugin automatically manages to upload kernels to the resources	⊗	▶	☐

After that go to Base → Components → Create → Storage and create a new Storage from the type “Lvm Storage Server (NFS)”. Select the openQRM Server “resource”



New Storage

Deployment: Lvm Storage Server (Nfs)

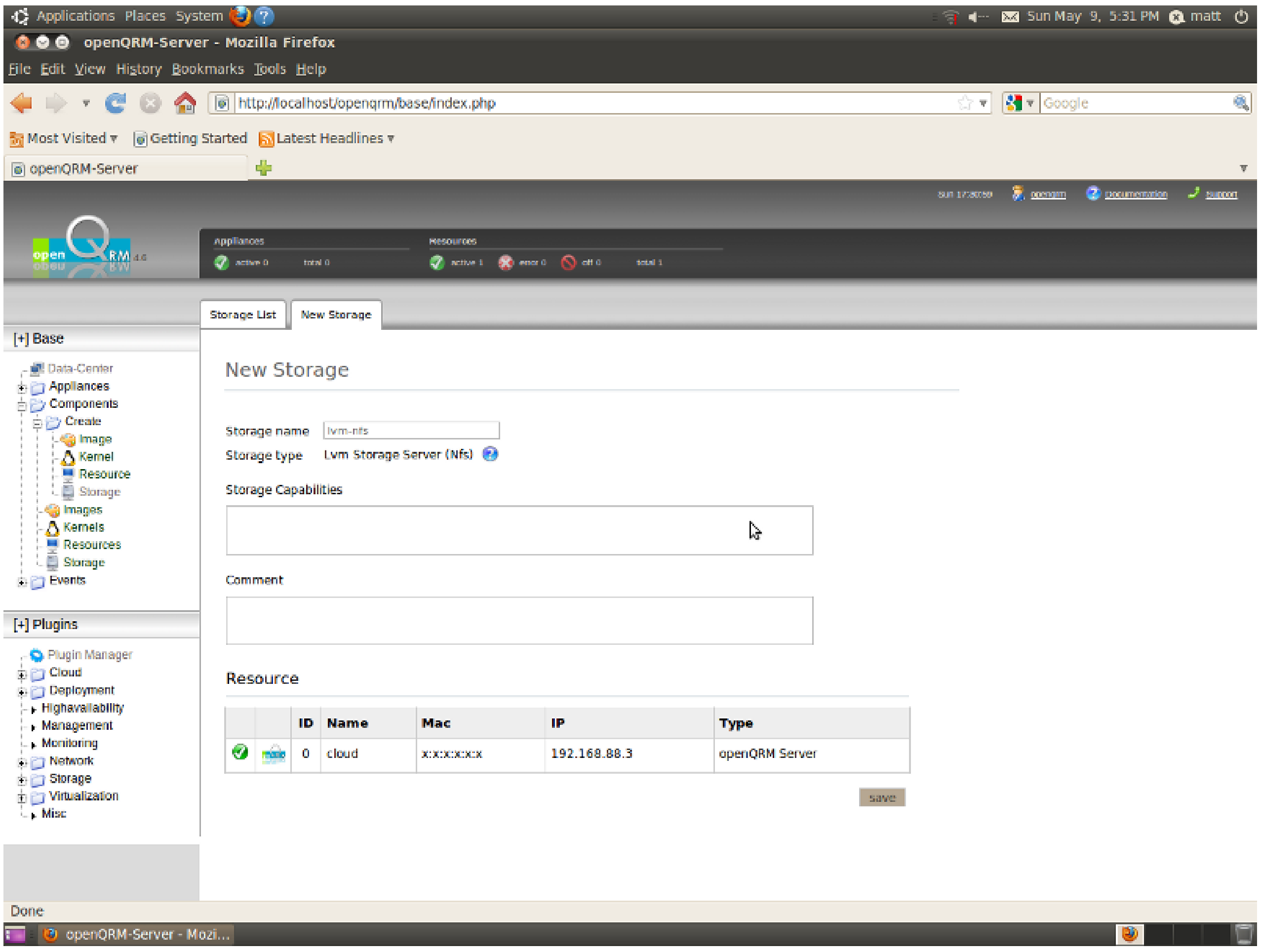
Resource List

order by ID ASC offset 0 limit 10

1 - 1 / 1

		ID	Name	Mac	IP	Type	
✓		0	cloud	x:x:x:x:x:x	192.168.88.3	openQRM Server	●

Here just give the Storage Server a name and save.


openQRM-Server - Mozilla Firefox
http://localhost/openqrm/base/index.php
Storage List
New Storage
[+] Base
[+] Plugins
save

New Storage

Storage name	lvm-nfs
Storage type	Lvm Storage Server (Nfs)

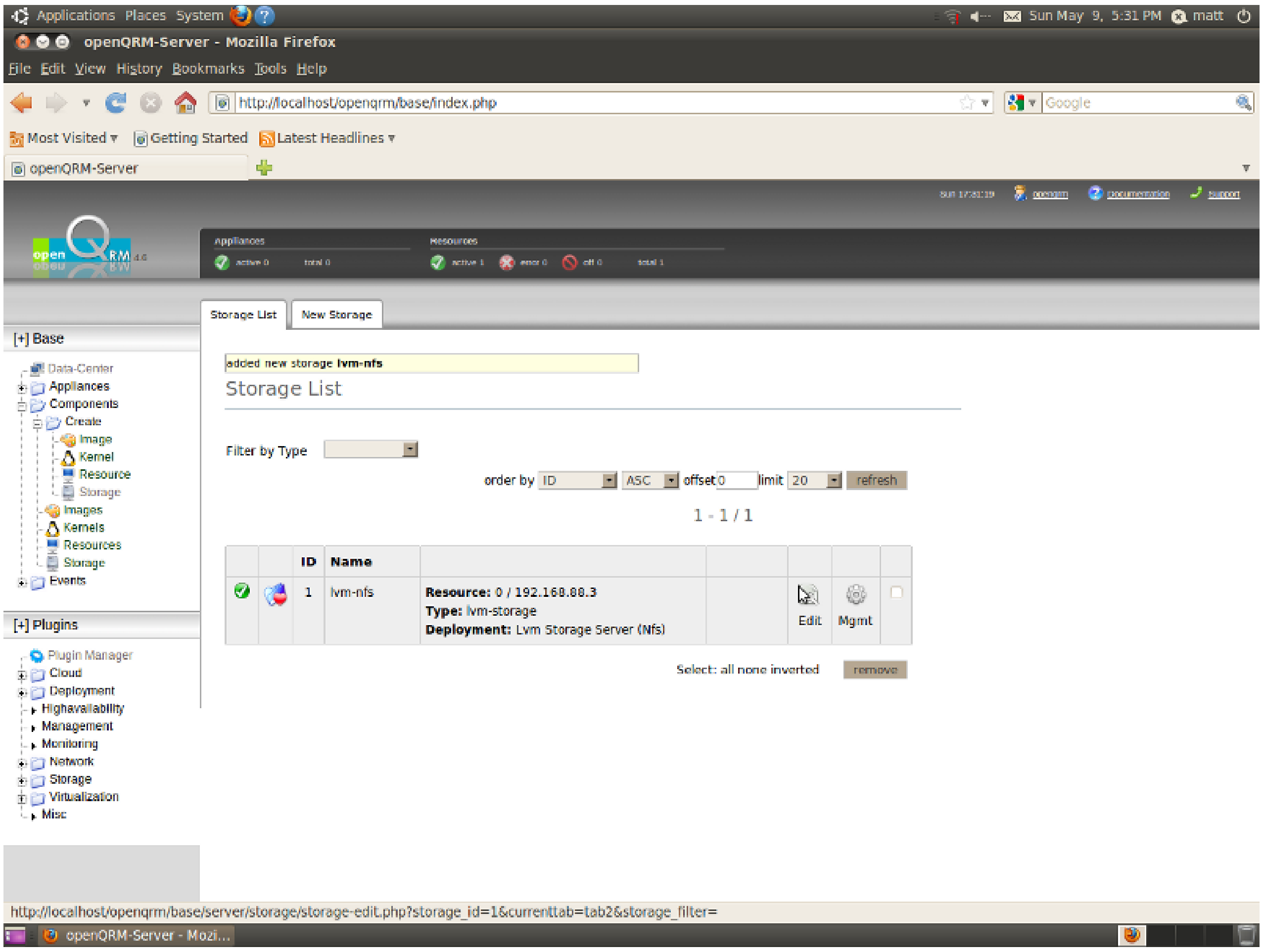
Storage Capabilities

Comment

Resource

		ID	Name	Mac	IP	Type
✓		0	cloud	x:x:x:x:x:x	192.168.88.3	openQRM Server

Here how the Storage List looks now.



added new storage **lvm-nfs**

Storage List

Filter by Type

order by ID ASC offset 0 limit 20 refresh

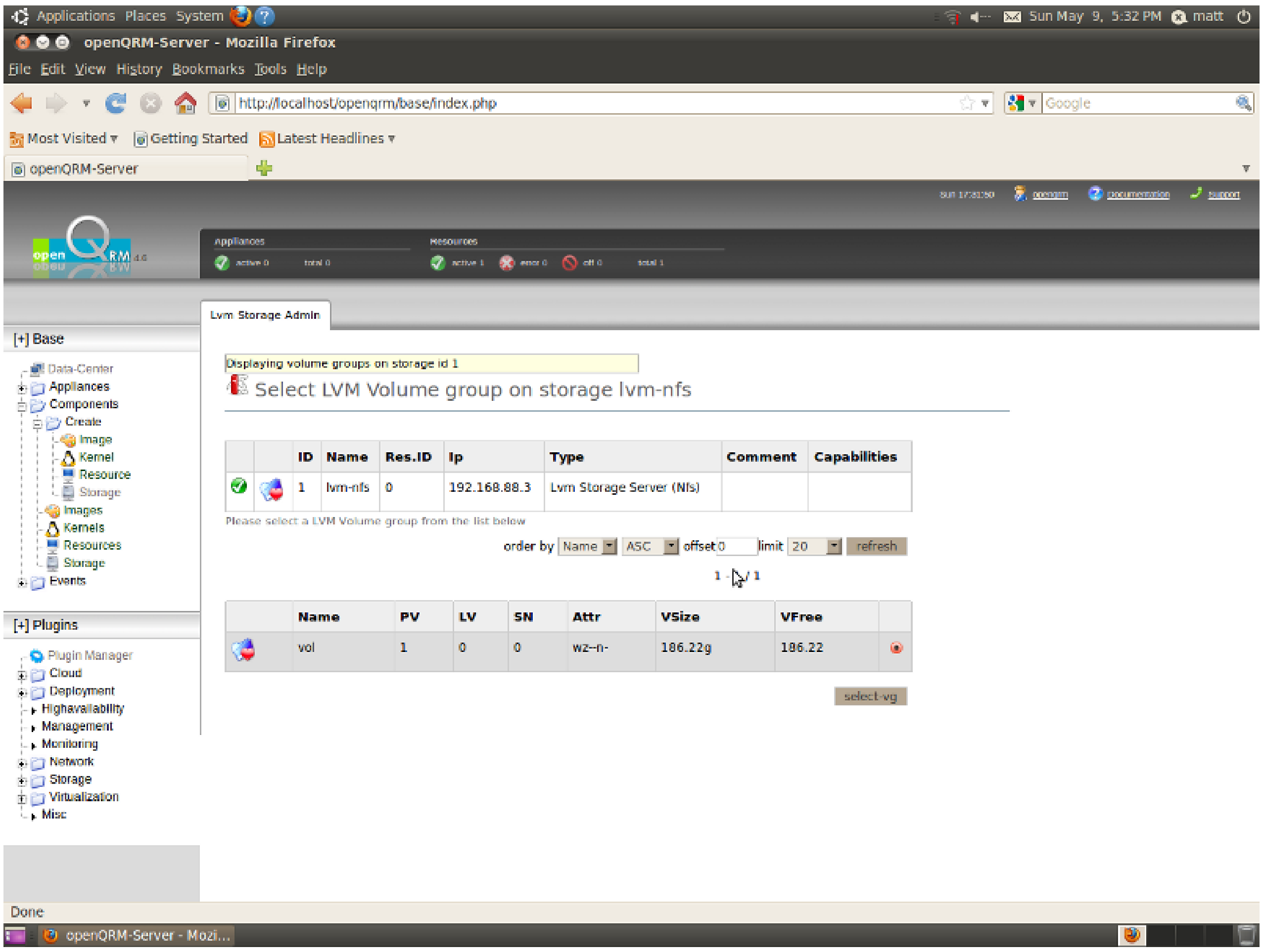
1 - 1 / 1

		ID	Name					
✓		1	lvm-nfs	**Resource:** 0 / 192.168.88.3 **Type:** lvm-storage **Deployment:** Lvm Storage Server (Nfs)		Edit	Mgmt	☐

Select: all none inverted remove

http://localhost/openqrm/base/server/storage/storage-edit.php?storage_id=1&currenttab=tab2&storage_filter=

Click on the “Mgmt” button of the new created “lvm-nfs” Storage Server.



Displaying volume groups on storage id 1

Select LVM Volume group on storage lvm-nfs

		ID	Name	Res.ID	Ip	Type	Comment	Capabilities
✓		1	lvm-nfs	0	192.168.88.3	Lvm Storage Server (Nfs)		

Please select a LVM Volume group from the list below

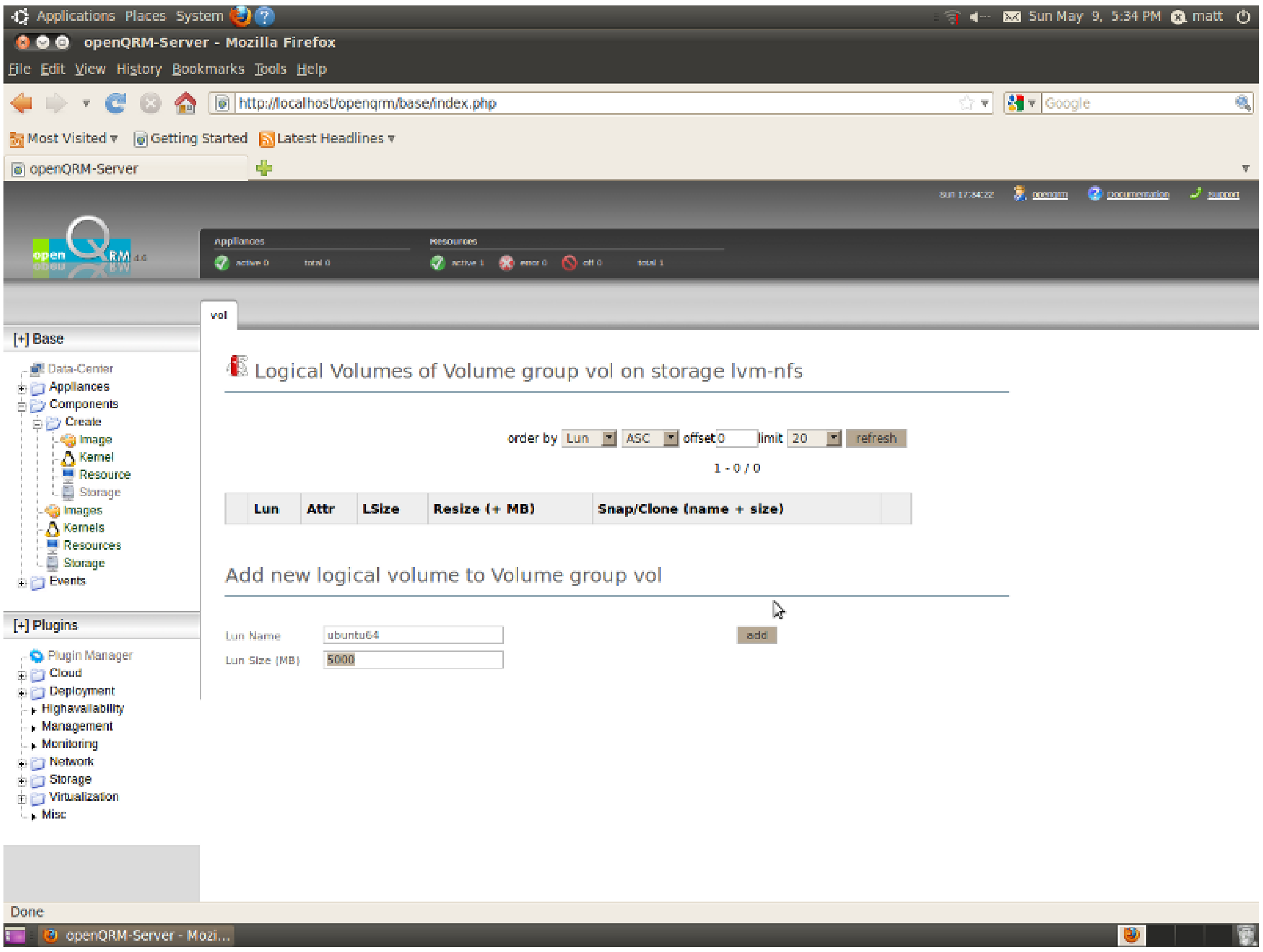
order by Name ASC offset 0 limit 20 refresh

1 / 1

	Name	PV	LV	SN	Attr	VSize	VFree	
	vol	1	0	0	wz--n-	186.22g	186.22	●

select-vg

Please select the “vol” volume group.



Logical Volumes of Volume group vol on storage lvm-nfs

order by Lun ASC offset 0 limit 20 refresh

1 - 0 / 0

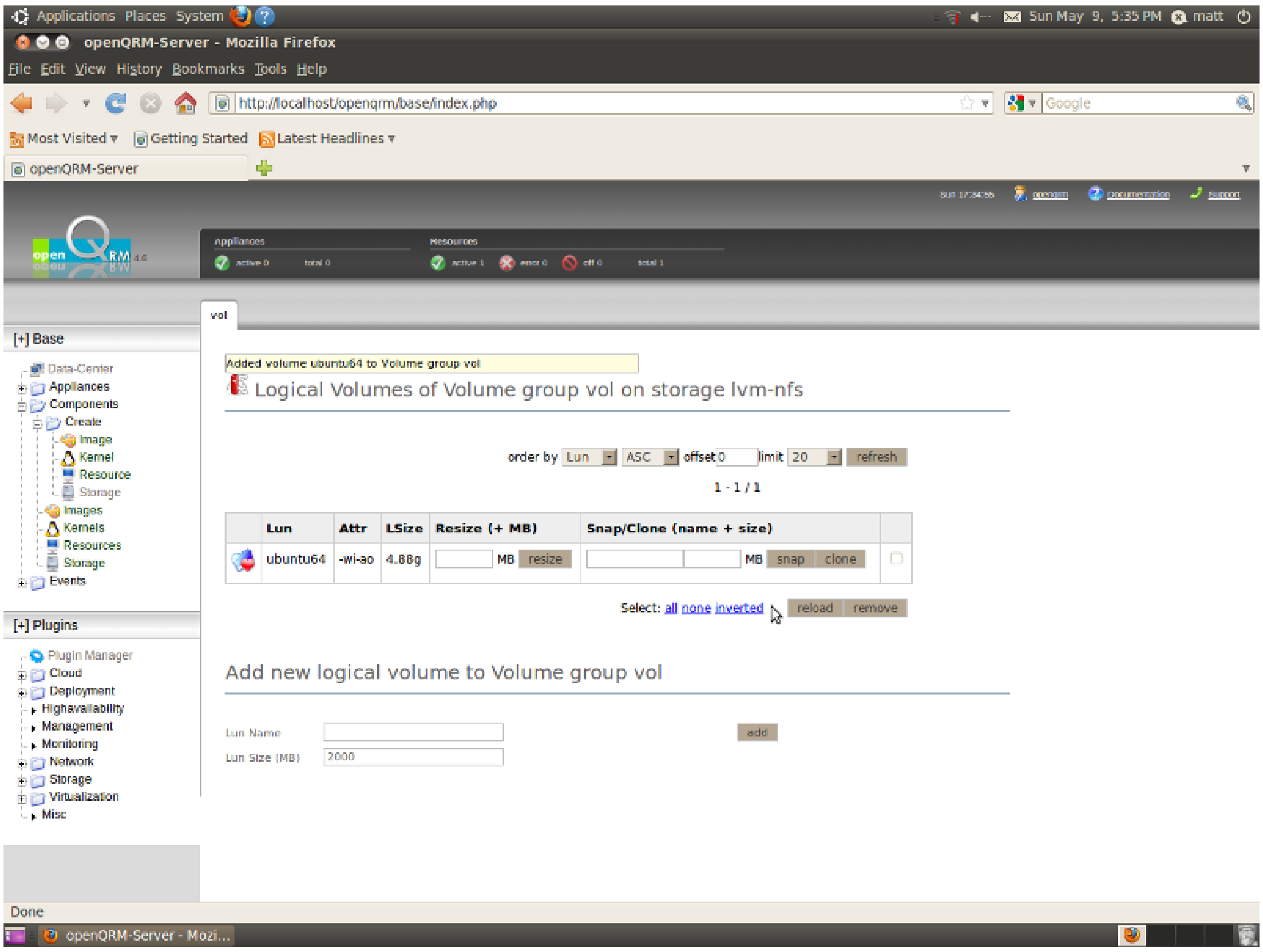
	Lun	Attr	LSize	Resize (+ MB)	Snap/Clone (name + size)	

Add new logical volume to Volume group vol

Lun Name	ubuntu64
Lun Size (MB)	5000

add

Now create a new volume with the name “ubuntu64” and the size 5000MB.



Added volume ubuntu64 to Volume group vol

Logical Volumes of Volume group vol on storage lvm-nfs

order by Lun ASC offset 0 limit 20 refresh

1 - 1 / 1

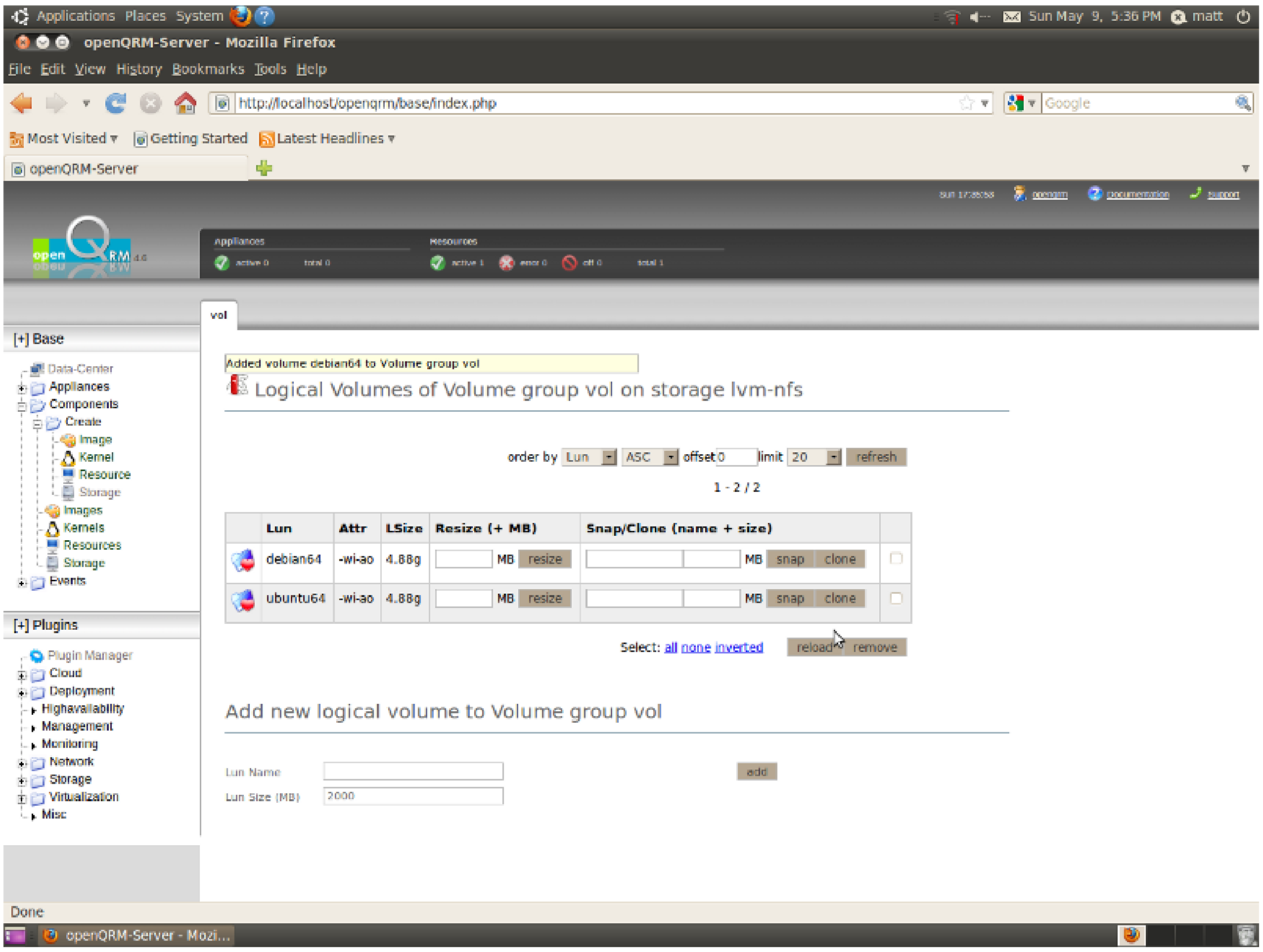
	Lun	Attr	LSize	Resize (+ MB)	Snap/Clone (name + size)	
	ubuntu64	-wi-ao	4.88g	MB resize	MB snap clone	☐

Select: all none inverted reload remove

Add new logical volume to Volume group vol

Lun Name		add
Lun Size (MB)	2000	

Also create another volume with the name “debian64” and the size 5000MB.



Added volume debian64 to Volume group vol

Logical Volumes of Volume group vol on storage lvm-nfs

order by Lun ASC offset 0 limit 20 refresh

1 - 2 / 2

	Lun	Attr	LSize	Resize (+ MB)	Snap/Clone (name + size)	
	debian64	-wi-ao	4.88g	MB resize	MB snap clone	☐
	ubuntu64	-wi-ao	4.88g	MB resize	MB snap clone	☐

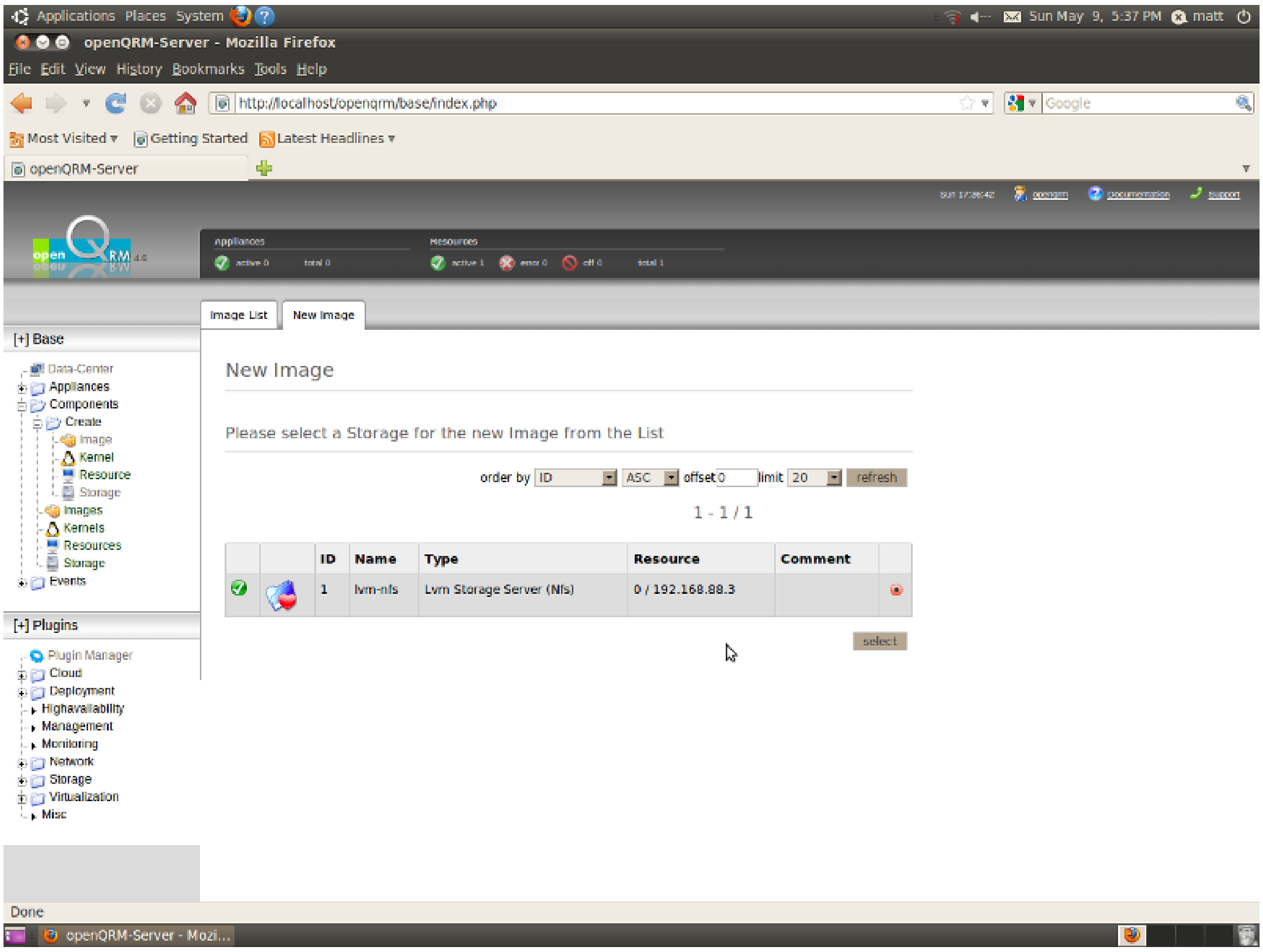
Select: all none inverted reload remove

Add new logical volume to Volume group vol

Lun Name	
Lun Size (MB)	2000

add

Go to Base → Components → Create → Image and create “images” from the just created “volumes”. First step is to select the Storage Server on which the “image” is physically located.



New Image

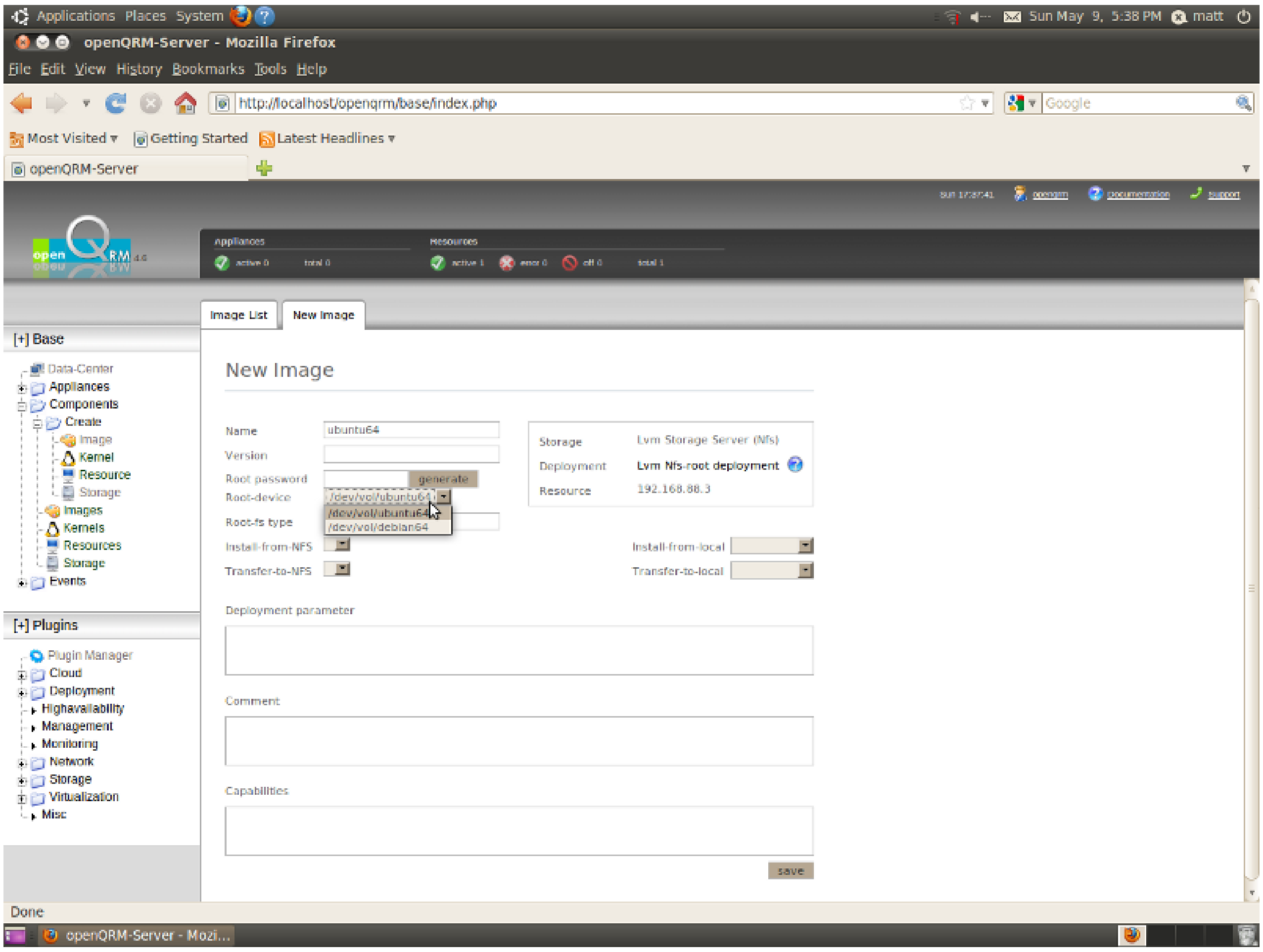
Please select a Storage for the new Image from the List

order by ID ASC offset 0 limit 20

1 - 1 / 1

		ID	Name	Type	Resource	Comment	
✓		1	lvm-nfs	Lvm Storage Server (Nfs)	0 / 192.168.88.3		●

In the next form give the “image” a name (here “ubuntu64”) and select the “ubuntu64” volume as the root-device identifier.

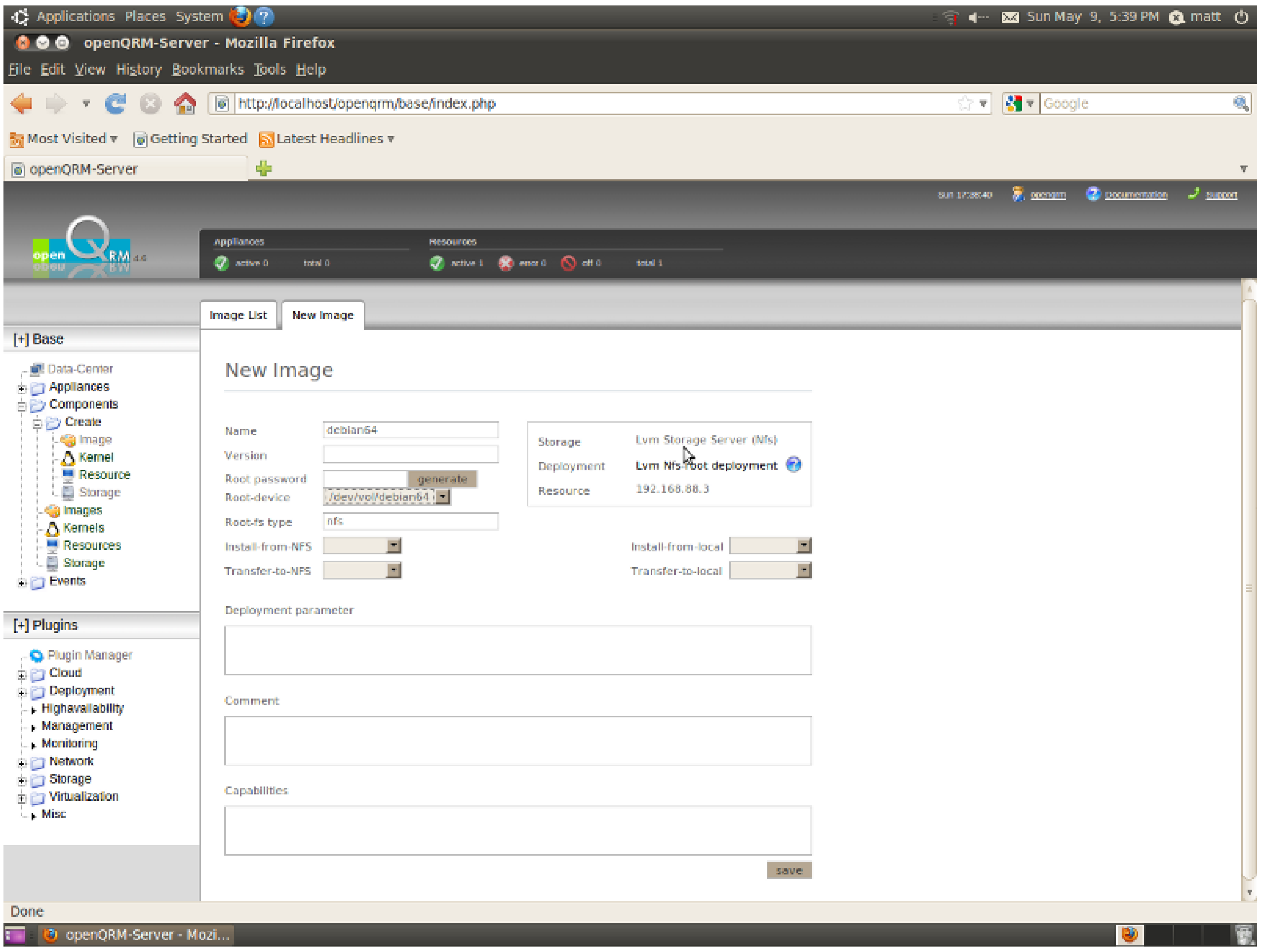

Applications Places System
openQRM-Server - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/openqrm/base/index.php
Image List
New Image
[+] Base
[+] Plugins
save

New Image

Name	ubuntu64
Version	
Root password	
Root-device	/dev/vol/ubuntu64
Root-fs type	

Storage	Lvm Storage Server (Nfs)
Deployment	Lvm Nfs-root deployment
Resource	192.168.88.3

Repeat the previous 2 steps to create another “image” named “debian64” using the “debian64” volume as its root-device.



New Image

Name	debian64
Version	
Root password	
Root-device	/dev/vol/debian64
Root-fs type	nfs
Install-from-NFS	
Transfer-to-NFS	
Install-from-local	
Transfer-to-local	

Storage	Lvm Storage Server (Nfs)
Deployment	Lvm Nfs-root deployment
Resource	192.168.88.3

Deployment parameter

Comment

Capabilities

Here how the Image List looks like now.

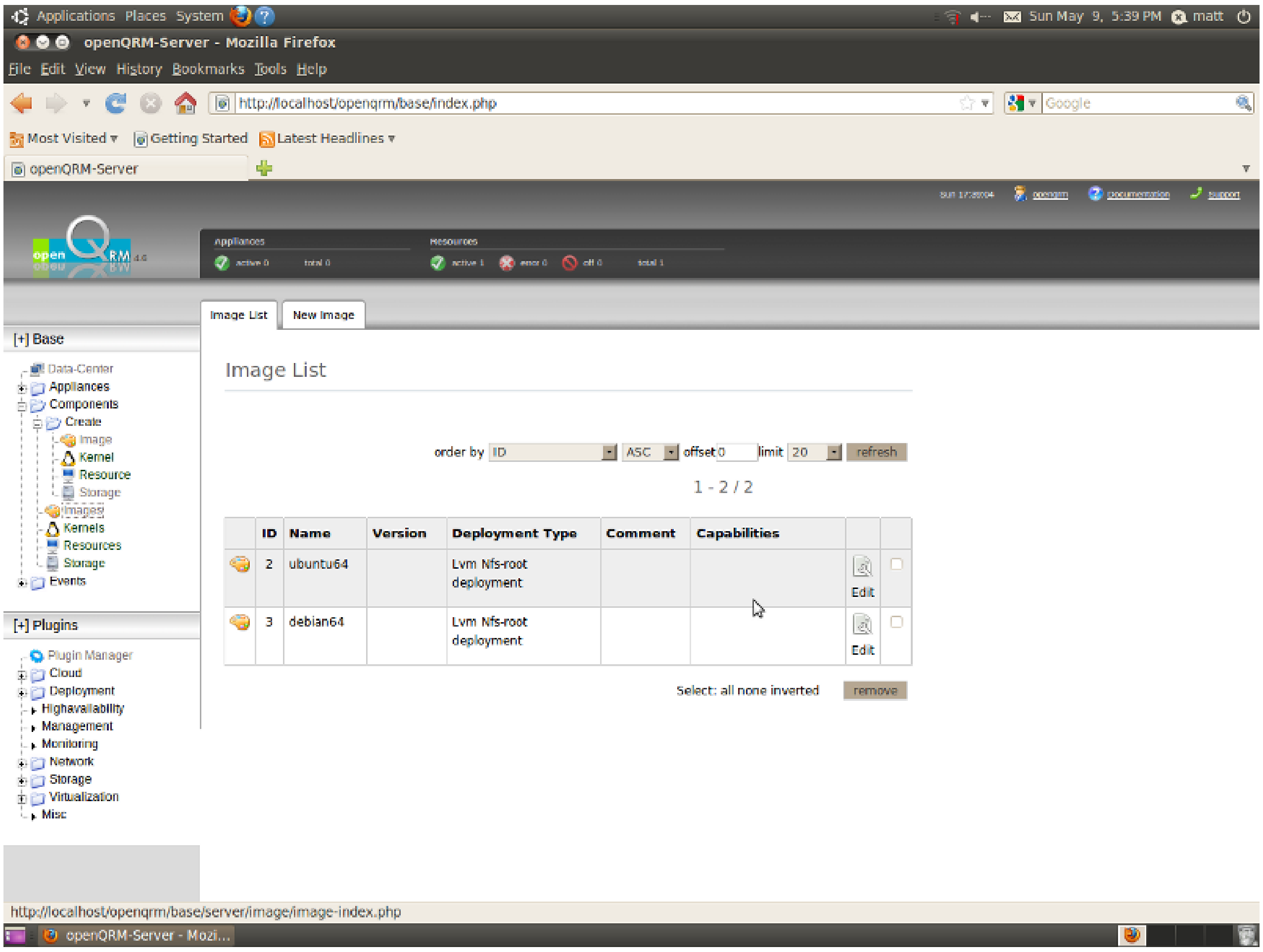


Image List

order by ID ASC offset 0 limit 20 refresh

1 - 2 / 2

ID	Name	Version	Deployment Type	Comment	Capabilities		
2	ubuntu64		Lvm Nfs-root deployment			Edit	☐
3	debian64		Lvm Nfs-root deployment			Edit	☐

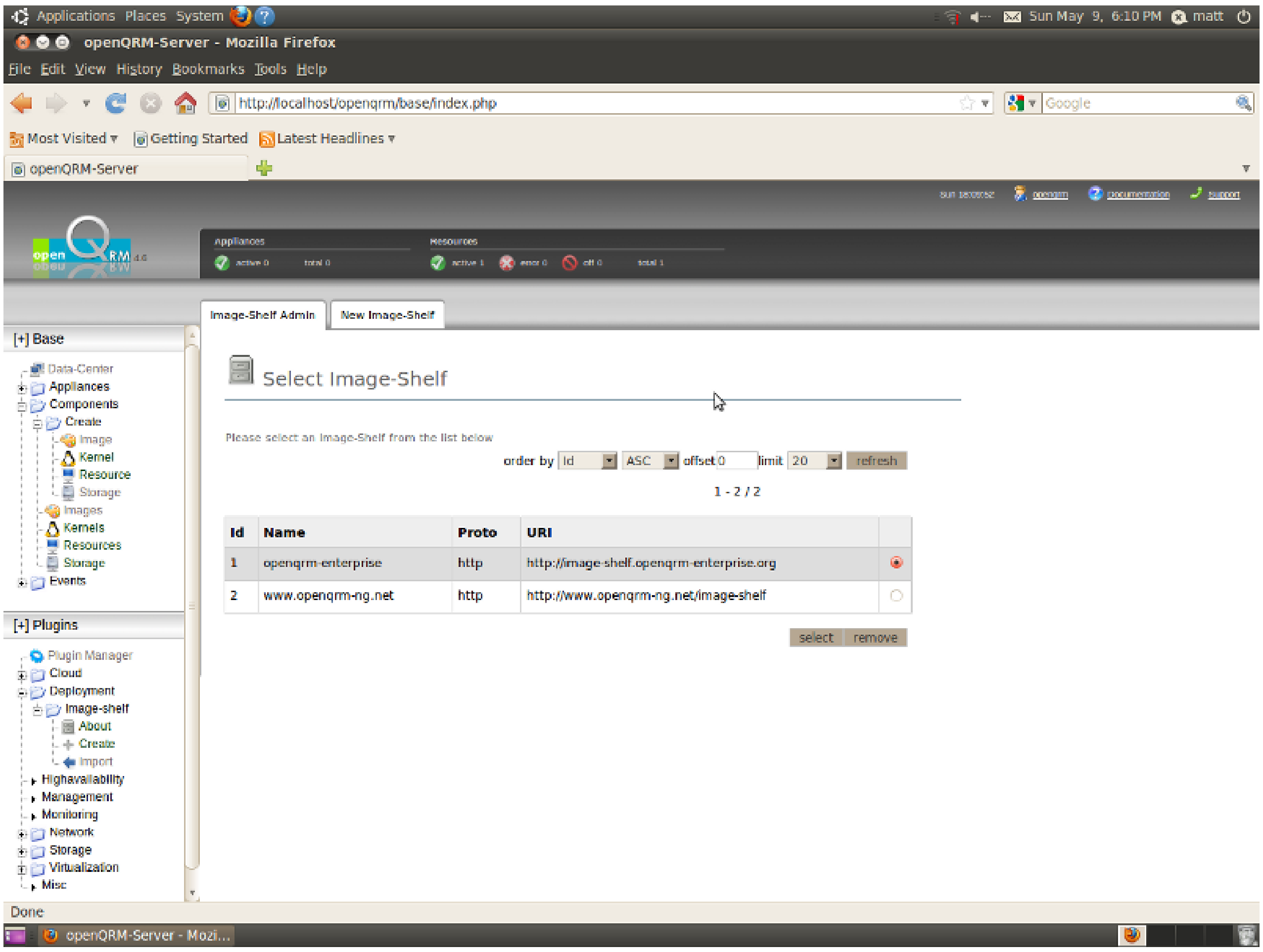
Select: all none inverted remove

On the console it will now look like this :

```
matt@cloud:~$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1             38543848   3470824  33115088  10% /
none                   1523376       324   1523052   1% /dev
none                   1528204       200   1528004   1% /dev/shm
none                   1528204       164   1528040   1% /var/run
none                   1528204         0   1528204   0% /var/lock
none                   1528204         0   1528204   0% /lib/init/rw
none                  38543848   3470824  33115088  10% /var/lib/ureadahead/debugfs
/dev/mapper/vol-ubuntu64
                       5039616    141212   4642404   3% /vol/ubuntu64
/dev/mapper/vol-debian64
                       5039616    141212   4642404   3% /vol/debian64
matt@cloud:~$ sudo exportfs
/vol/ubuntu64  192.168.88.3
/vol/debian64  192.168.88.3
matt@cloud:~$ ls /vol/ubuntu64/
lost+found
matt@cloud:~$ ls /vol/debian64/
lost+found
matt@cloud:~$
```

Now we are going to populate the still empty “images” with root-filesystem content via the “image-shelf” plug-in.

Go to Plugins → Deployment → Image-Shelf → Import and select the “openqrm-enterprise” Image-Shelf.



Select Image-Shelf

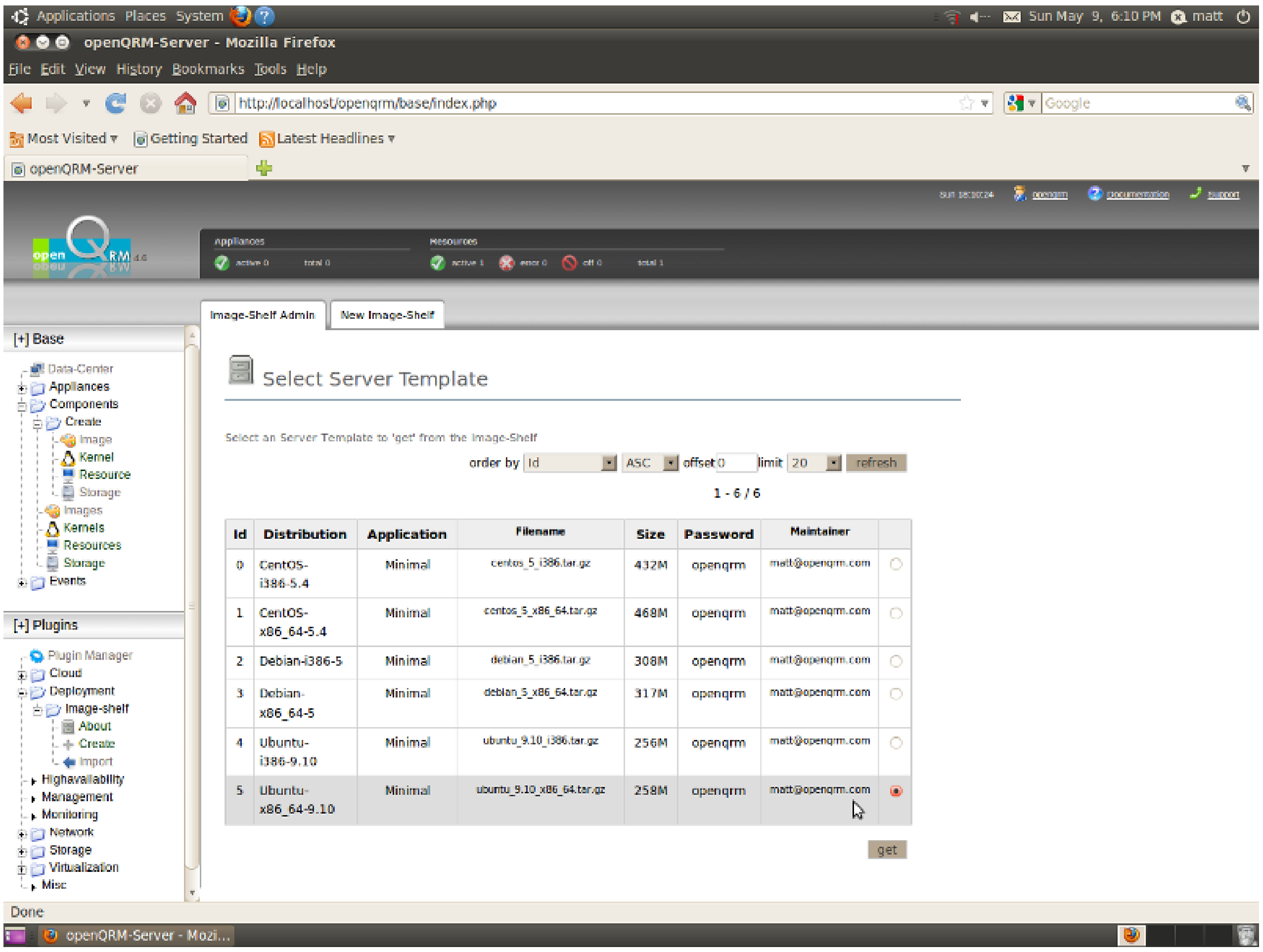
Please select an Image-Shelf from the list below

order by Id ASC offset 0 limit 20

1 - 2 / 2

Id	Name	Proto	URI	
1	openqrm-enterprise	http	http://image-shelf.openqrm-enterprise.org	●
2	www.openqrm-ng.net	http	http://www.openqrm-ng.net/image-shelf	○

On this screen it will provide a list of available Server-Templates. Please select the “Ubuntu x86_64” Template and click on “get”.



Select Server Template

Select an Server Template to 'get' from the Image-Shelf

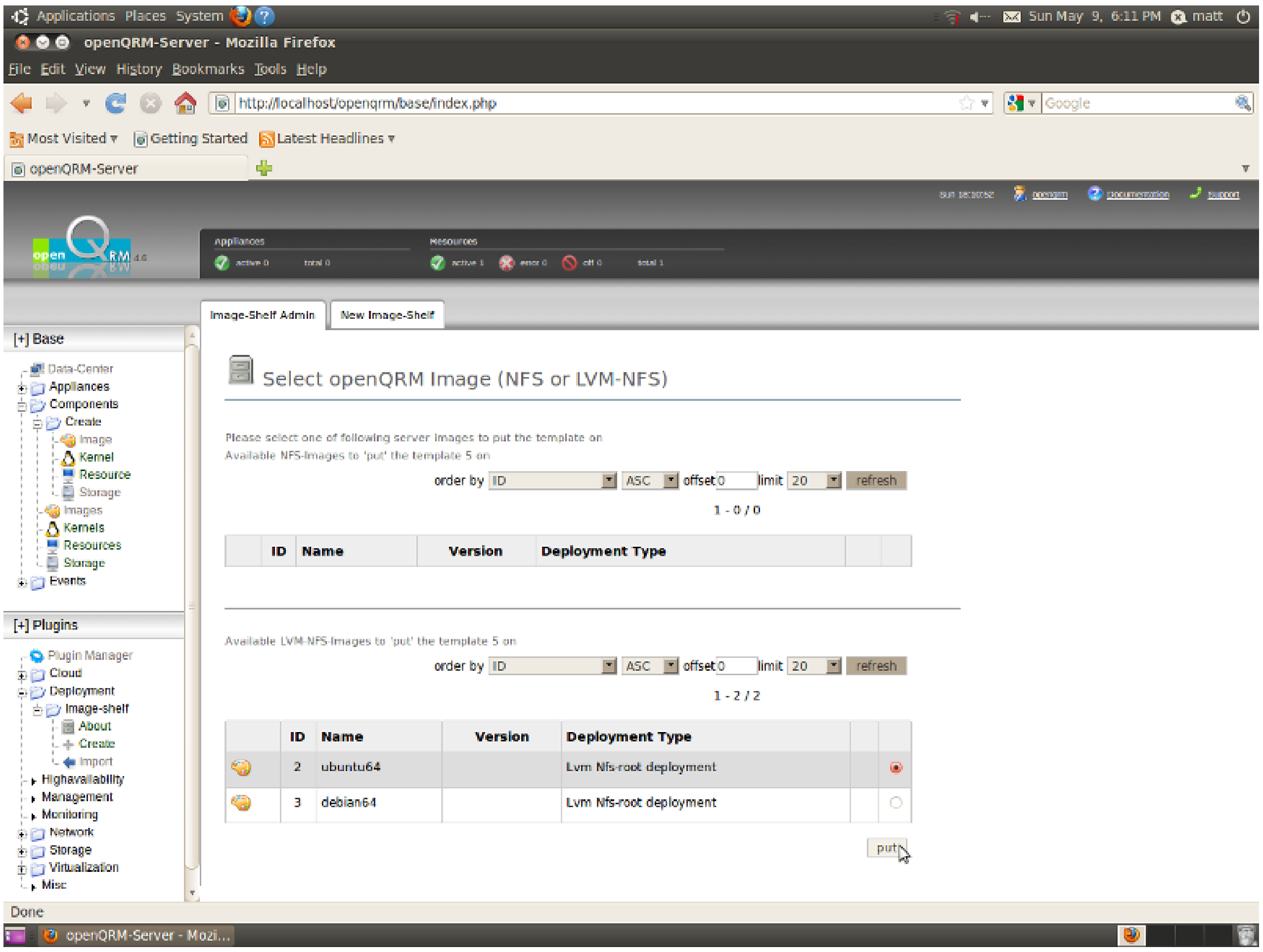
order by Id ASC offset 0 limit 20 refresh

1 - 6 / 6

Id	Distribution	Application	Filename	Size	Password	Maintainer	
0	CentOS-i386-5.4	Minimal	centos_5_i386.tar.gz	432M	openqrm	matt@openqrm.com	○
1	CentOS-x86_64-5.4	Minimal	centos_5_x86_64.tar.gz	468M	openqrm	matt@openqrm.com	○
2	Debian-i386-5	Minimal	debian_5_i386.tar.gz	308M	openqrm	matt@openqrm.com	○
3	Debian-x86_64-5	Minimal	debian_5_x86_64.tar.gz	317M	openqrm	matt@openqrm.com	○
4	Ubuntu-i386-9.10	Minimal	ubuntu_9.10_i386.tar.gz	256M	openqrm	matt@openqrm.com	○
5	Ubuntu-x86_64-9.10	Minimal	ubuntu_9.10_x86_64.tar.gz	258M	openqrm	matt@openqrm.com	●

get

Here select the “image” where to “put” the Server-Template to. Select the “ubuntu64” image created before and click on “put”.



Select openQRM Image (NFS or LVM-NFS)

Please select one of following server images to put the template on
Available NFS-Images to 'put' the template 5 on

order by ID ASC offset 0 limit 20 refresh

1 - 0 / 0

ID	Name	Version	Deployment Type

Available LVM-NFS-Images to 'put' the template 5 on

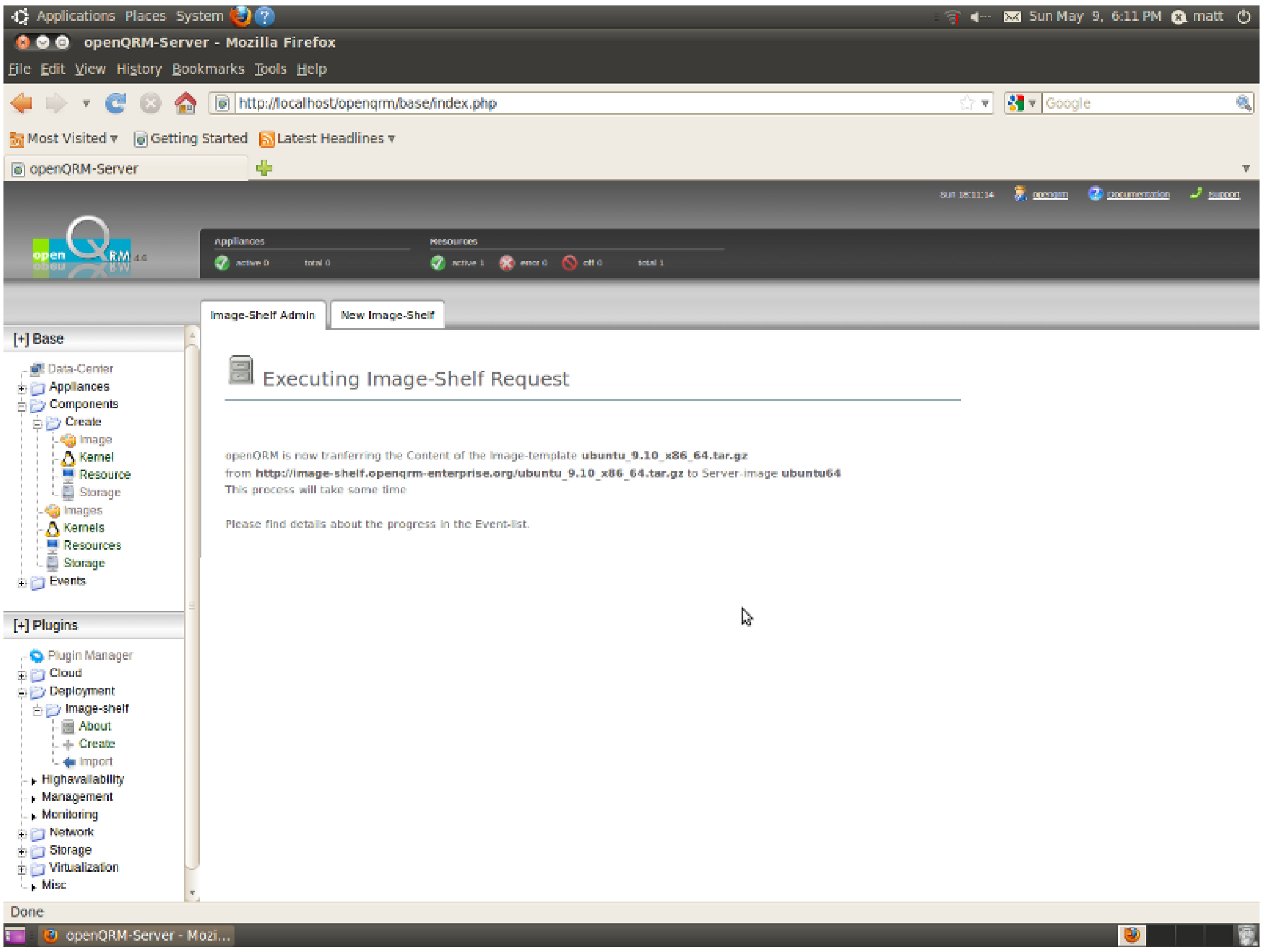
order by ID ASC offset 0 limit 20 refresh

1 - 2 / 2

	ID	Name	Version	Deployment Type	
	2	ubuntu64		Lvm Nfs-root deployment	●
	3	debian64		Lvm Nfs-root deployment	○

put

The Image-Shelf is now progressing the request in the background. It will download the selected Server-Template and unpack it on the Storage Servers "image-location". **Please notice that this process will take some time ! You can check the progress in the Event List.**



Executing Image-Shelf Request

openQRM is now tranferring the Content of the Image-template **ubuntu_9.10_x86_64.tar.gz**
from **http://image-shelf.openqrm-enterprise.org/ubuntu_9.10_x86_64.tar.gz** to Server-image **ubuntu64**
This process will take some time

Please find details about the progress in the Event-list.

Please repeat the same Image-Shelf steps for the "debian64" "image".

After the Image-shelf finished the download and unpack phase it will now look like this on the console :

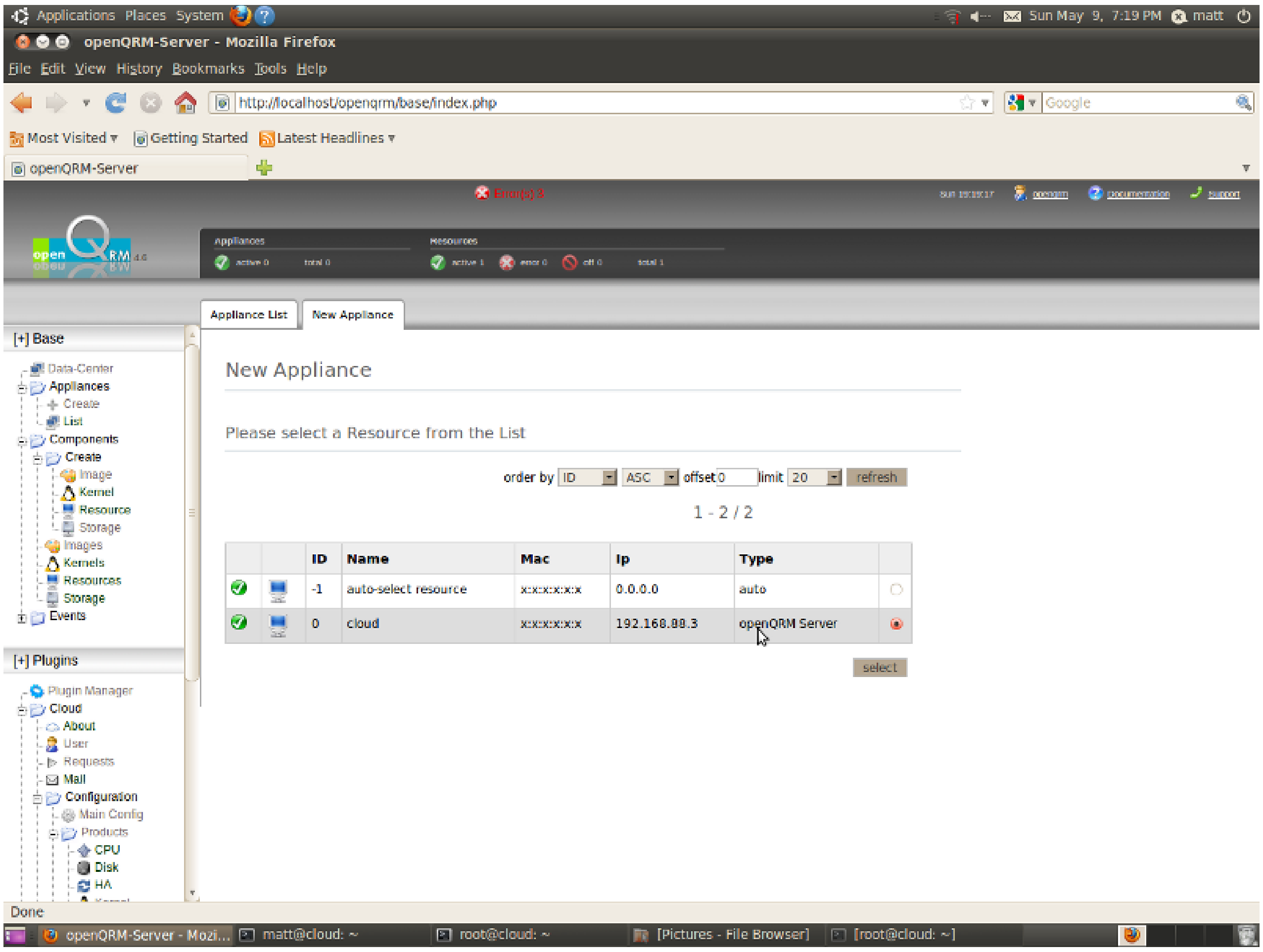
```
matt@cloud:~$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1             38543848   3552184  33033728  10% /
none                   1523376       324   1523052   1% /dev
none                   1528204       200   1528004   1% /dev/shm
none                   1528204       164   1528040   1% /var/run
none                   1528204         0   1528204   0% /var/lock
none                   1528204         0   1528204   0% /lib/init/rw
none                  38543848   3552184  33033728  10% /var/lib/ureadahead/debugfs
/dev/mapper/vol-ubuntu64
                       5039616   1144532   3639084  24% /vol/ubuntu64
/dev/mapper/vol-debian64
                       5039616   1084104   3699512  23% /vol/debian64
matt@cloud:~$ sudo exportfs
/vol/ubuntu64  192.168.88.3
/vol/debian64  192.168.88.3
matt@cloud:~$ ls /vol/ubuntu64/
bin   cdrom  etc   initrd.img  lib64       media  opt   root  selinux  sys  usr  vmlinuz
boot  dev    home  lib         lost+found  mnt    proc  sbin  srv      tmp  var
matt@cloud:~$ ls /vol/debian64/
bin   cdrom  emul  home        lib   lib64       media  opt   root  selinux  sys  usr  vmlinuz
boot  dev    etc   initrd.img  lib32  lost+found  mnt   proc  sbin  srv      tmp  var
matt@cloud:~$
```

Both "images" are now filled with a valid root-filesystem and we can now continue to configure the openQRM Cloud.

9. Create a KVM Host

Now we have to tell openQRM which systems are “Virtualization Hosts”.

Please go to Base → Appliances → Create and select the openQRM Server “resource”.



New Appliance

Please select a Resource from the List

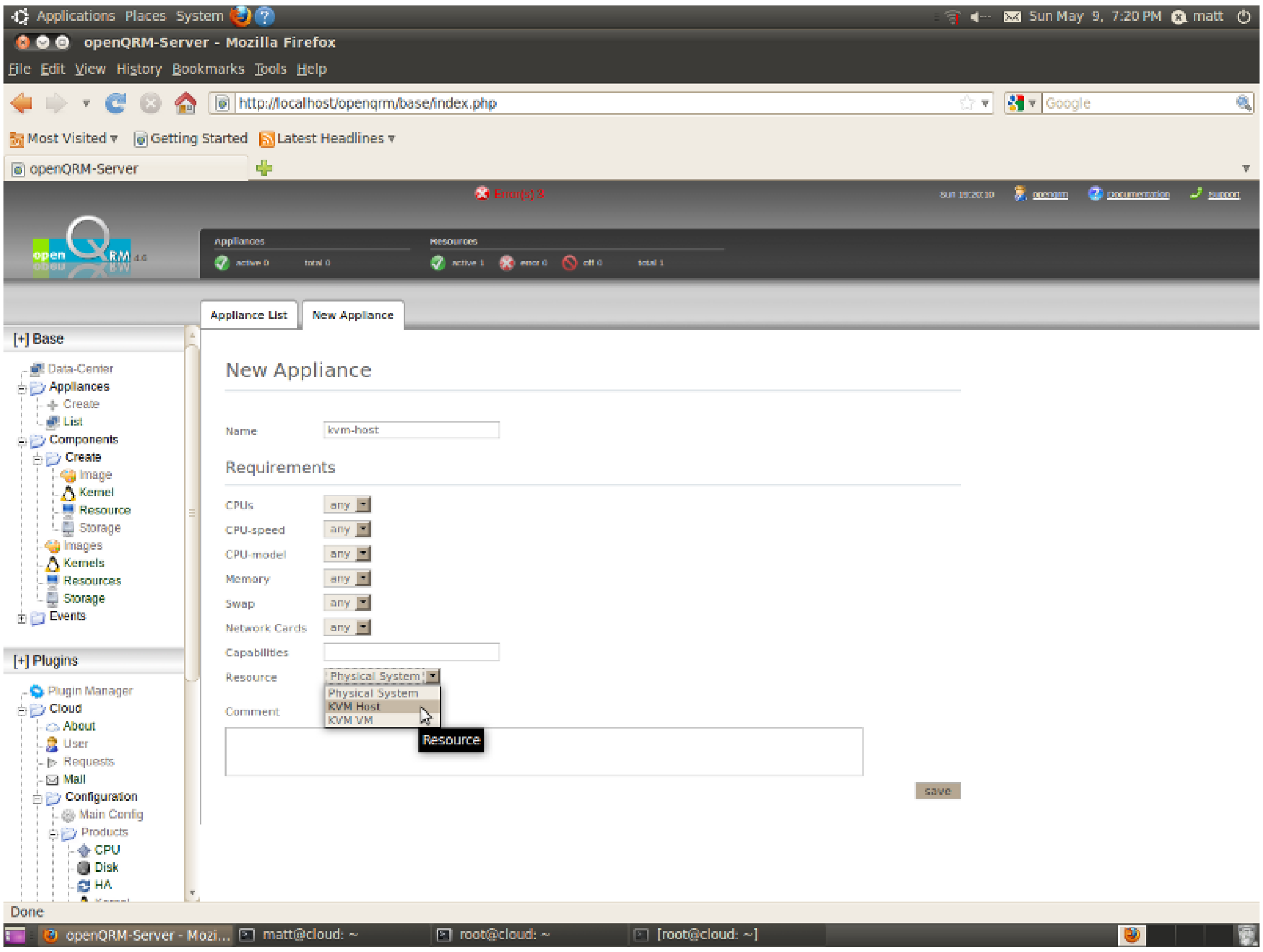
order by ID ASC offset 0 limit 20 refresh

1 - 2 / 2

		ID	Name	Mac	Ip	Type	
✓		-1	auto-select resource	x:x:x:x:x:x	0.0.0.0	auto	○
✓		0	cloud	x:x:x:x:x:x	192.168.88.3	openQRM Server	●

select

Now provide a name for the new appliance (here we have used “kvm-host”) and set the “resource-type” to “KVM Host”. Click on save.



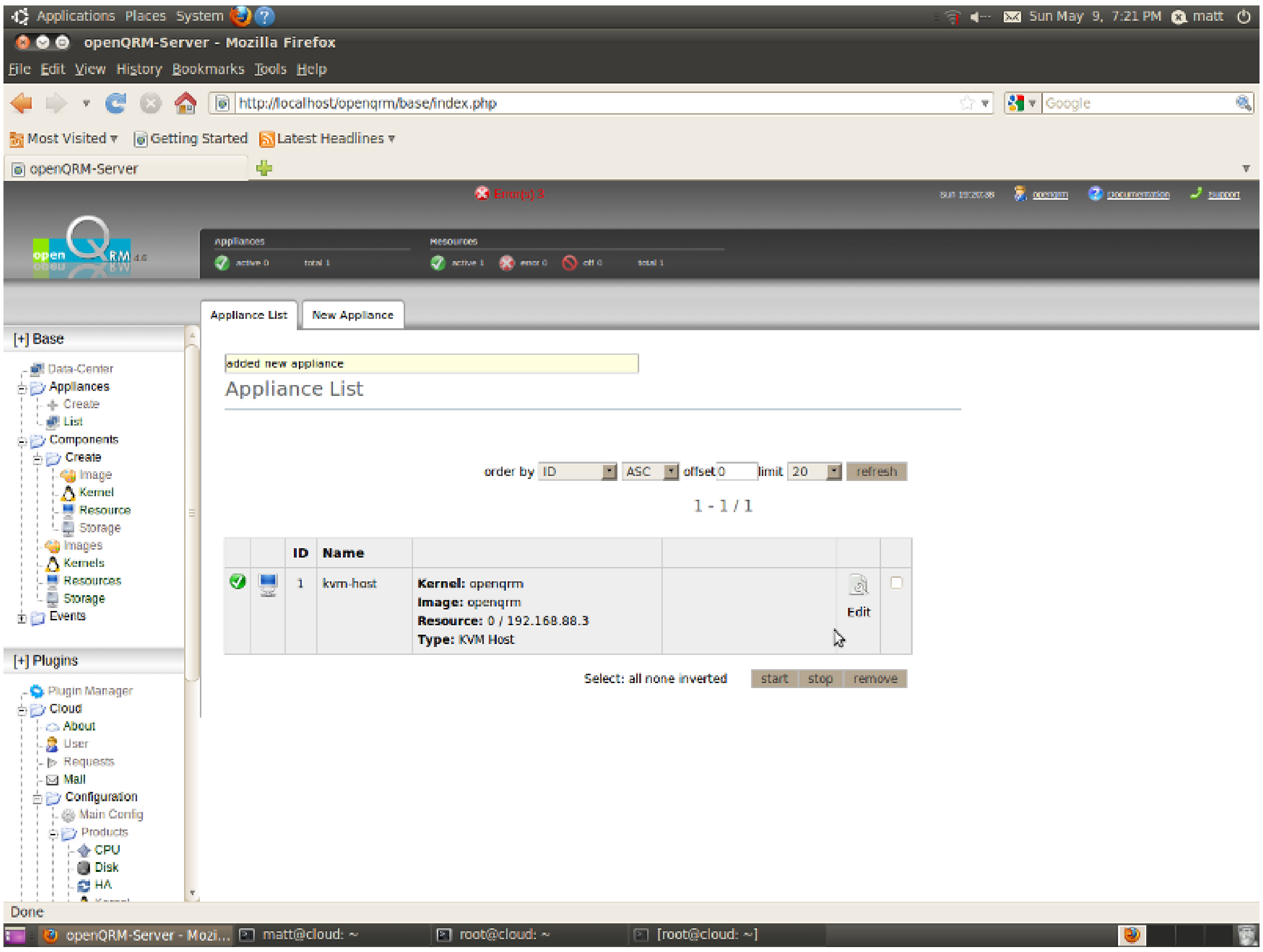
New Appliance

Name	kvm-host

Requirements

CPUs	any
CPU-speed	any
CPU-model	any
Memory	any
Swap	any
Network Cards	any
Capabilities	
Resource	Physical System
Comment	

Here the Appliance List after we have created the “kvm-host” appliance.



added new appliance

Appliance List

order by ID ASC offset 0 limit 20 refresh

1 - 1 / 1

		ID	Name				
✓		1	kvm-host	**Kernel:** openqrm **Image:** openqrm **Resource:** 0 / 192.168.88.3 **Type:** KVM Host		Edit	☐

Select: all none inverted start stop remove

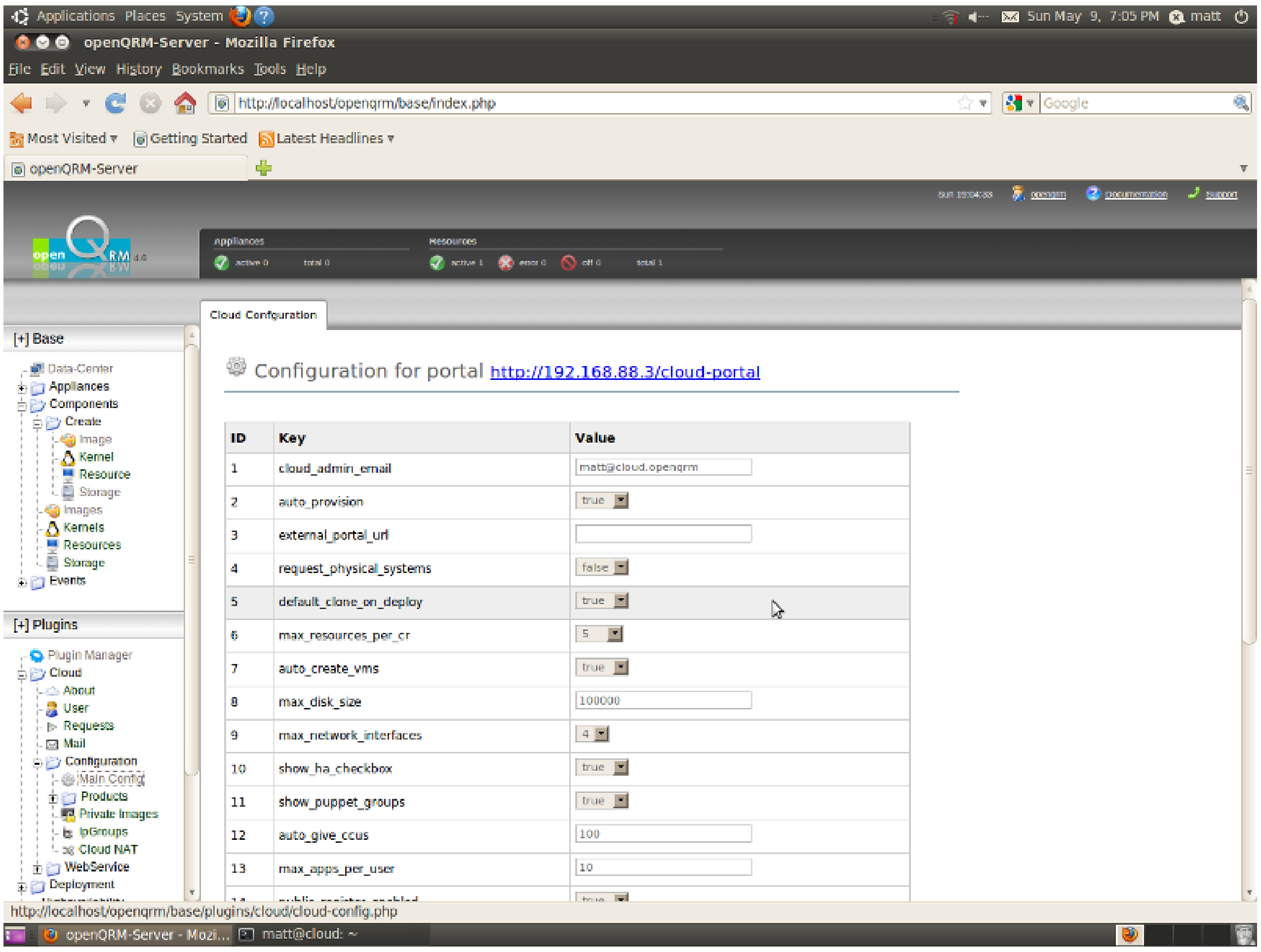
10. Configure the openQRM Cloud

After that please go to Plugins → Cloud → Configuration → Main Config and configure the following items :

- cloud_admin_email → your valid email address (may be a local address depending on your postfix setup)
- auto_provision → true
- external_portal_url → (optional) external Cloud Portal URL
- request_physical_systems → false
- auto_give_ccus → 100
- show_disk_resize → (optional) true
- show_private_image → true
- cloud_currency → (optional) set to US or Euro
- cloud_1000_ccus → How much 1000 CCUs are worh in US/Euro

For all other configuration items you can continue with the defaults. Save your configuration.

Here a screenshot of the Main Cloud Configuration page :

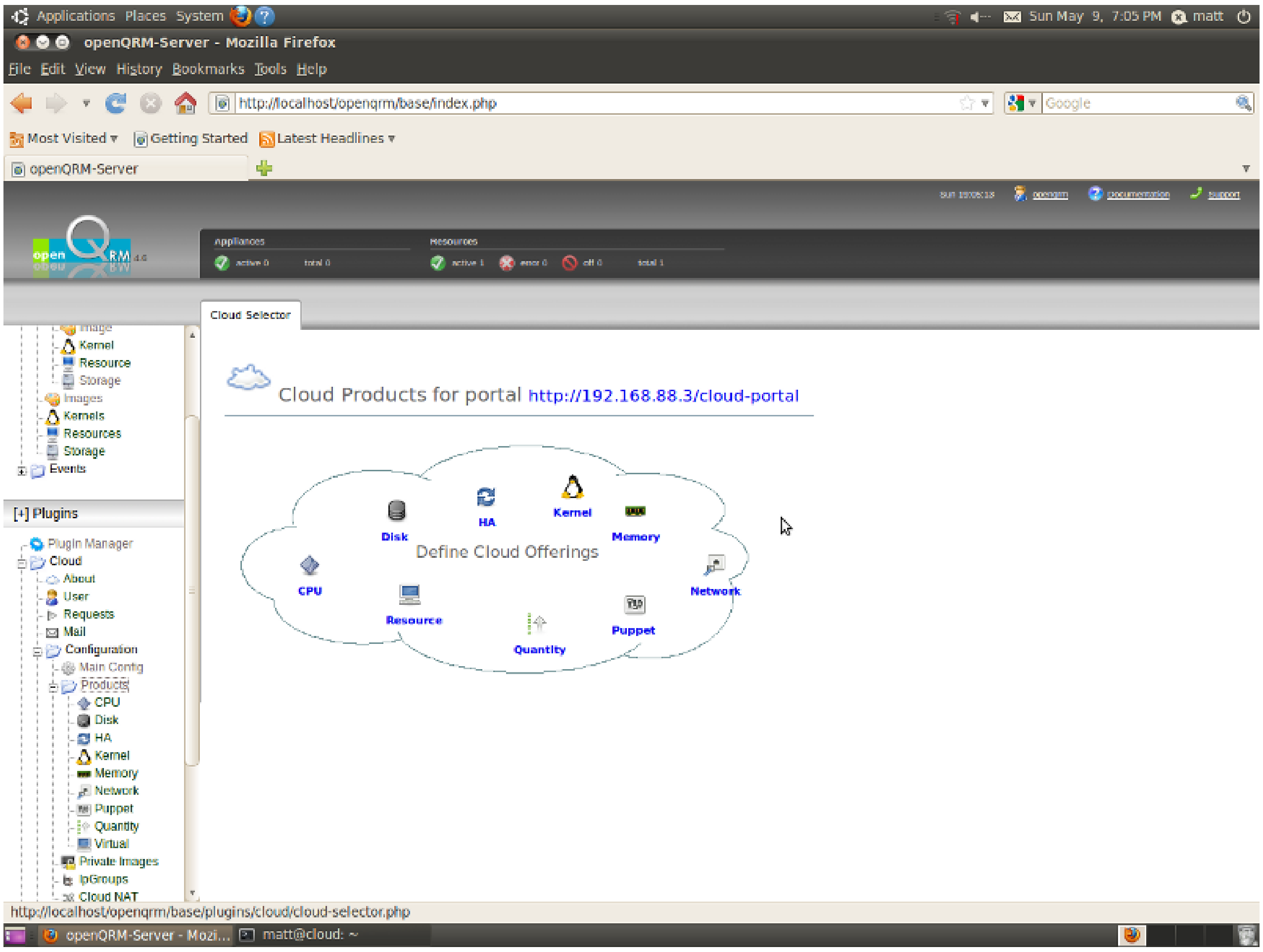


openQRM-Server - Mozilla Firefox
http://localhost/openqrm/base/index.php
[+] Base
[+] Plugins
Main Config
Cloud Configuration

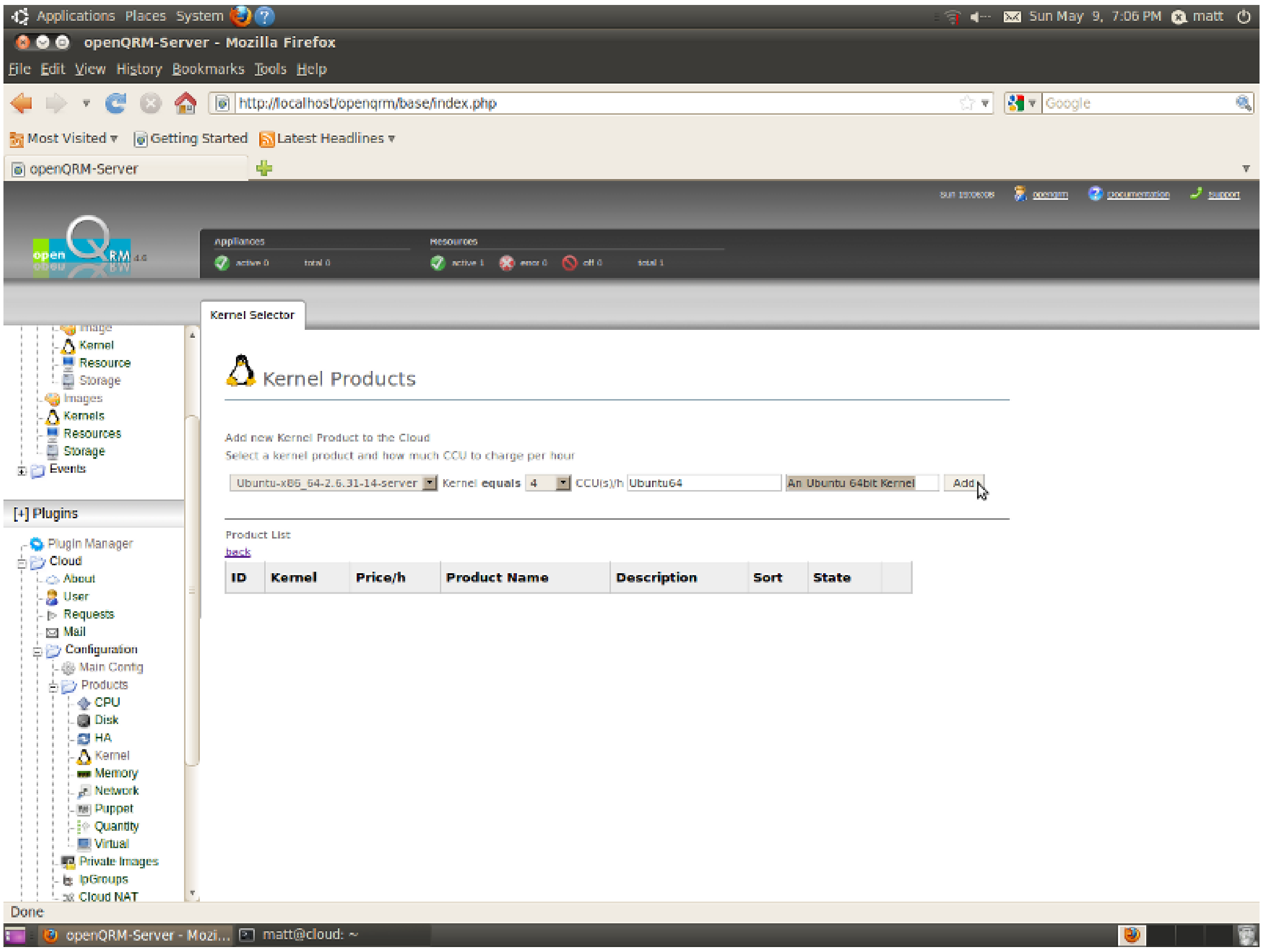
Configuration for portal http://192.168.88.3/cloud-portal

ID	Key	Value
1	cloud_admin_email	matt@cloud.openqrm
2	auto_provision	true
3	external_portal_url	
4	request_physical_systems	false
5	default_clone_on_deploy	true
6	max_resources_per_cr	5
7	auto_create_vms	true
8	max_disk_size	100000
9	max_network_interfaces	4
10	show_ha_checkbox	true
11	show_puppet_groups	true
12	auto_give_ccus	100
13	max_apps_per_user	10

Next step is to configure the Cloud Products via the “Cloud-Selector”.



Go to Plugins → Cloud → Configuration → Products → Kernel and create a new “Ubuntu64” kernel product.



Kernel Products

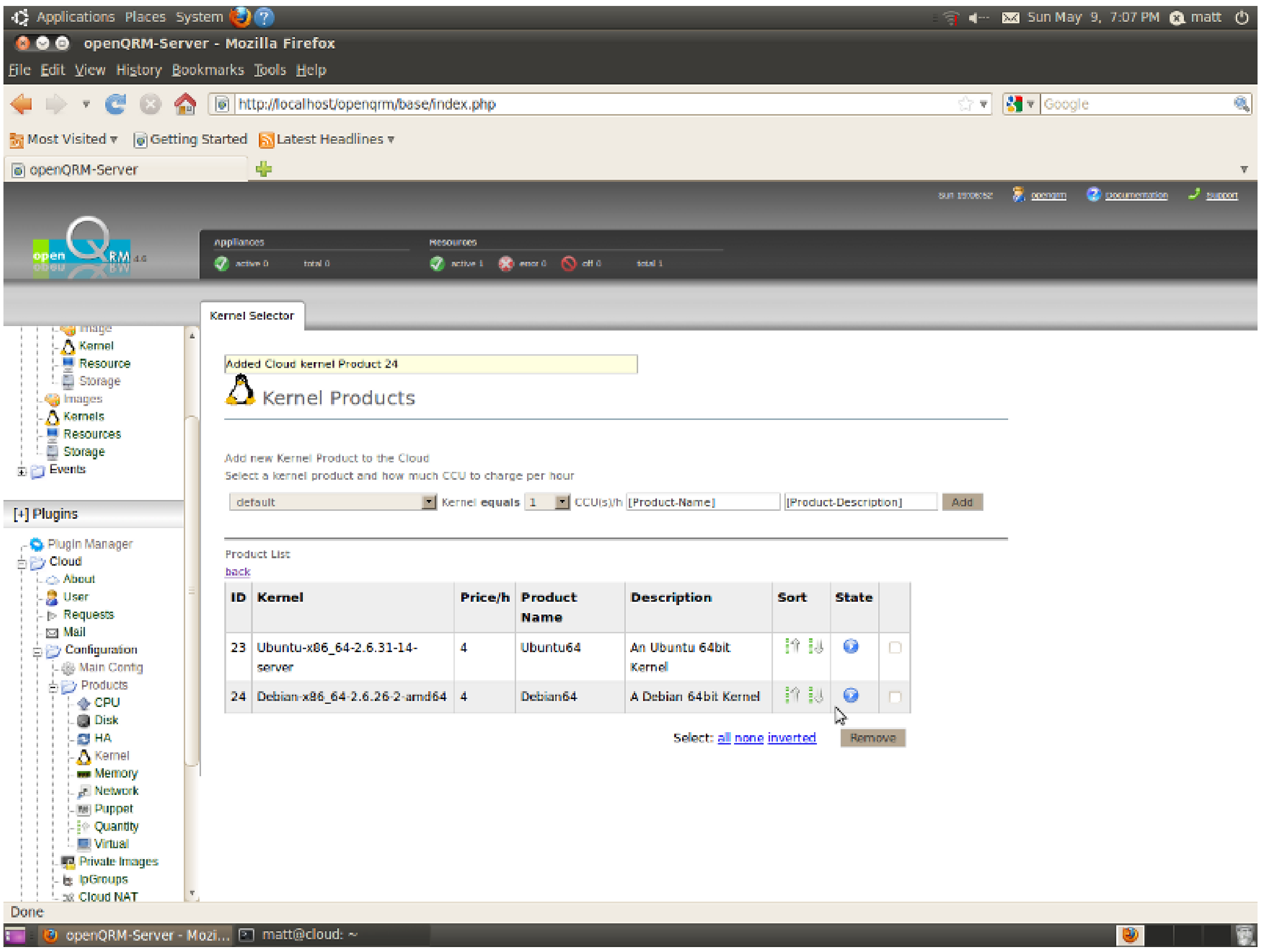
Add new Kernel Product to the Cloud
Select a kernel product and how much CCU to charge per hour

Ubuntu-x86_64-2.6.31-14-server Kernel equals 4 CCU(s)/h Ubuntu64 An Ubuntu 64bit Kernel Add

Product List
back

ID	Kernel	Price/h	Product Name	Description	Sort	State

Repeat this step and create a “Debian64” kernel product.



Added Cloud kernel Product 24

Add new Kernel Product to the Cloud
Select a kernel product and how much CCU to charge per hour

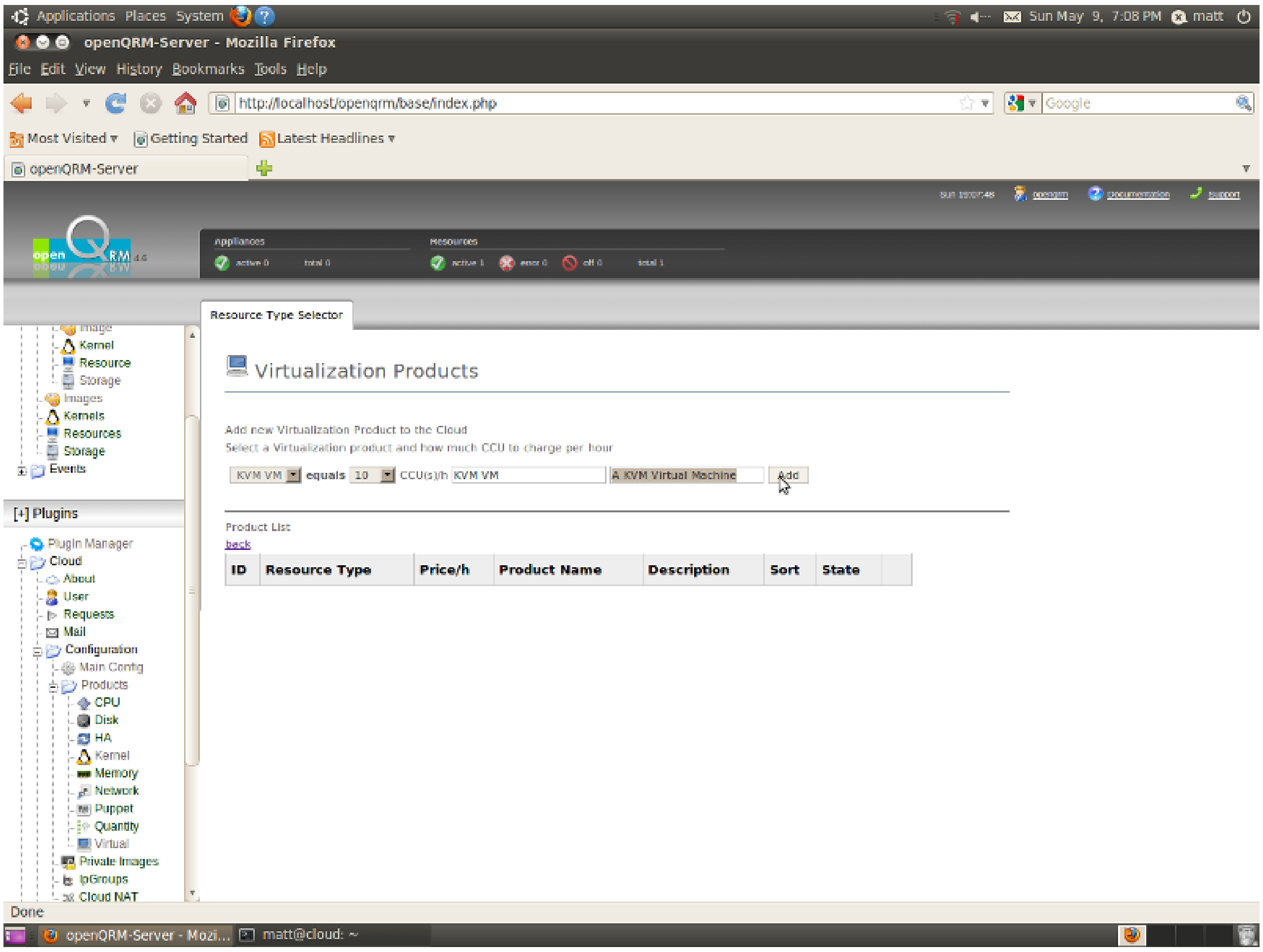
default Kernel **equals** 1 CCU(s)/h [Product-Name] [Product-Description]

Product List
back

ID	Kernel	Price/h	Product Name	Description	Sort	State	
23	Ubuntu-x86_64-2.6.31-14-server	4	Ubuntu64	An Ubuntu 64bit Kernel			☐
24	Debian-x86_64-2.6.26-2-amd64	4	Debian64	A Debian 64bit Kernel			☐

Select: all none inverted

Now create a “KVM VM” virtualization product.



Virtualization Products

Add new Virtualization Product to the Cloud

Select a Virtualization product and how much CCU to charge per hour

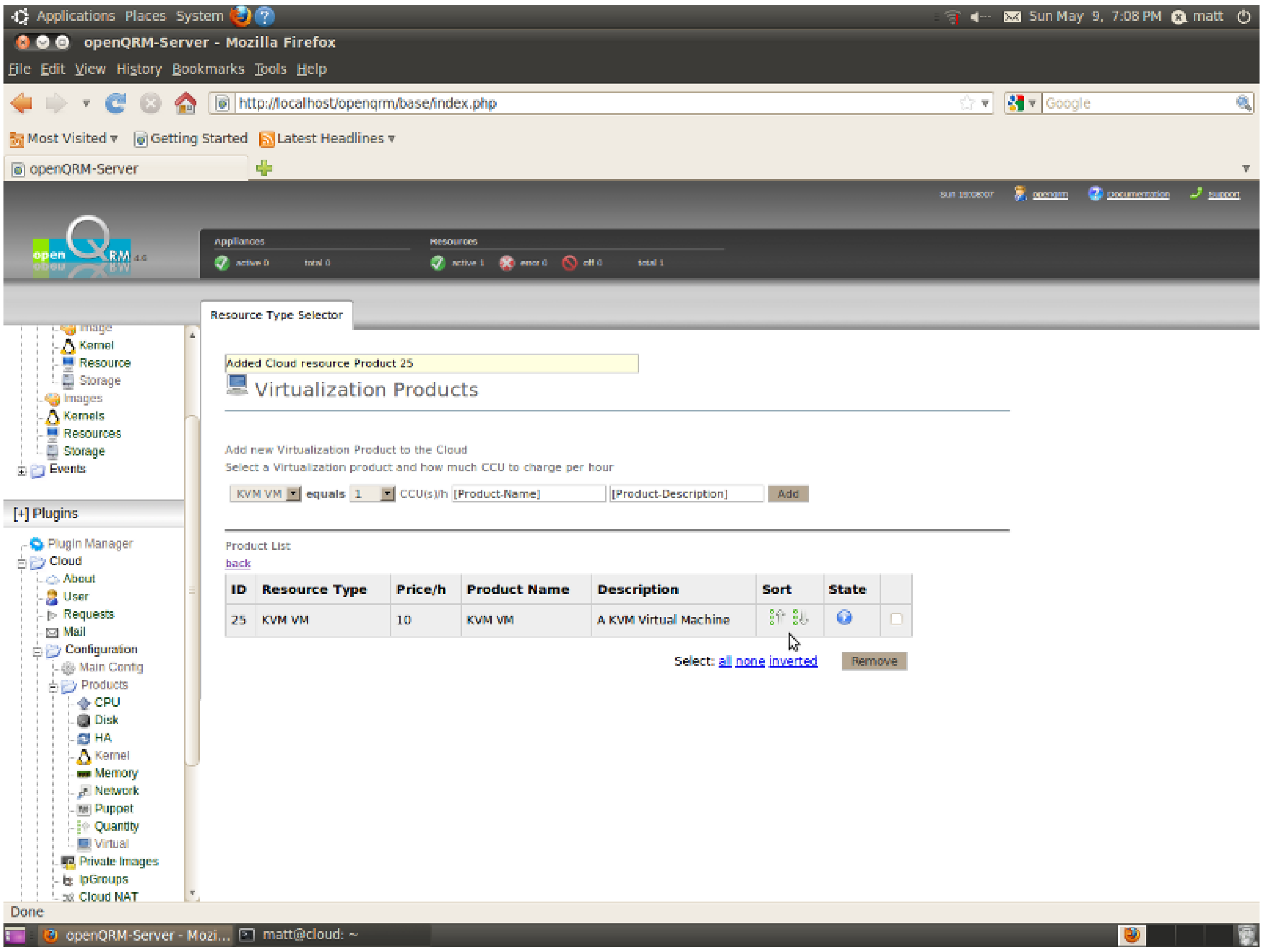
KVM VM equals 10 CCU(s)/h KVM VM | A KVM Virtual Machine | Add

Product List

back

ID	Resource Type	Price/h	Product Name	Description	Sort	State	

It will look like this :



Added Cloud resource Product 25

Virtualization Products

Add new Virtualization Product to the Cloud

Select a Virtualization product and how much CCU to charge per hour

KVM VM equals 1 CCU(s)/h [Product-Name] [Product-Description] Add

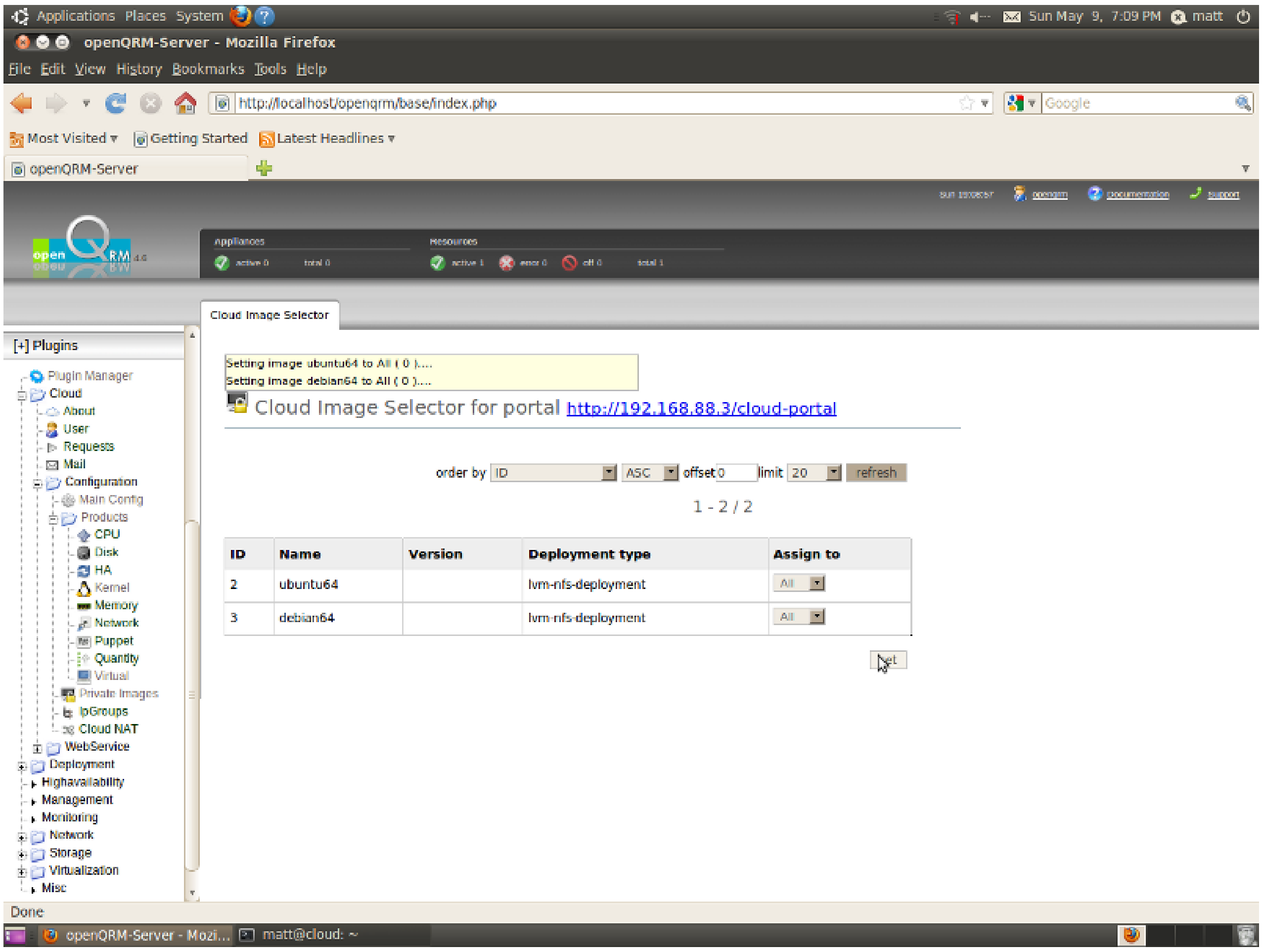
Product List

back

ID	Resource Type	Price/h	Product Name	Description	Sort	State	
25	KVM VM	10	KVM VM	A KVM Virtual Machine			☐

Select: all none inverted Remove

The next step is to tell the Cloud which “images” to show to the CloudUsers. Go to Plugins → Cloud → Configuration → Private Images and select “All” in the checkboxes for the ubuntu64 and debian64 “image”.



```
Setting image ubuntu64 to All ( 0 )....
Setting image debian64 to All ( 0 )....
```

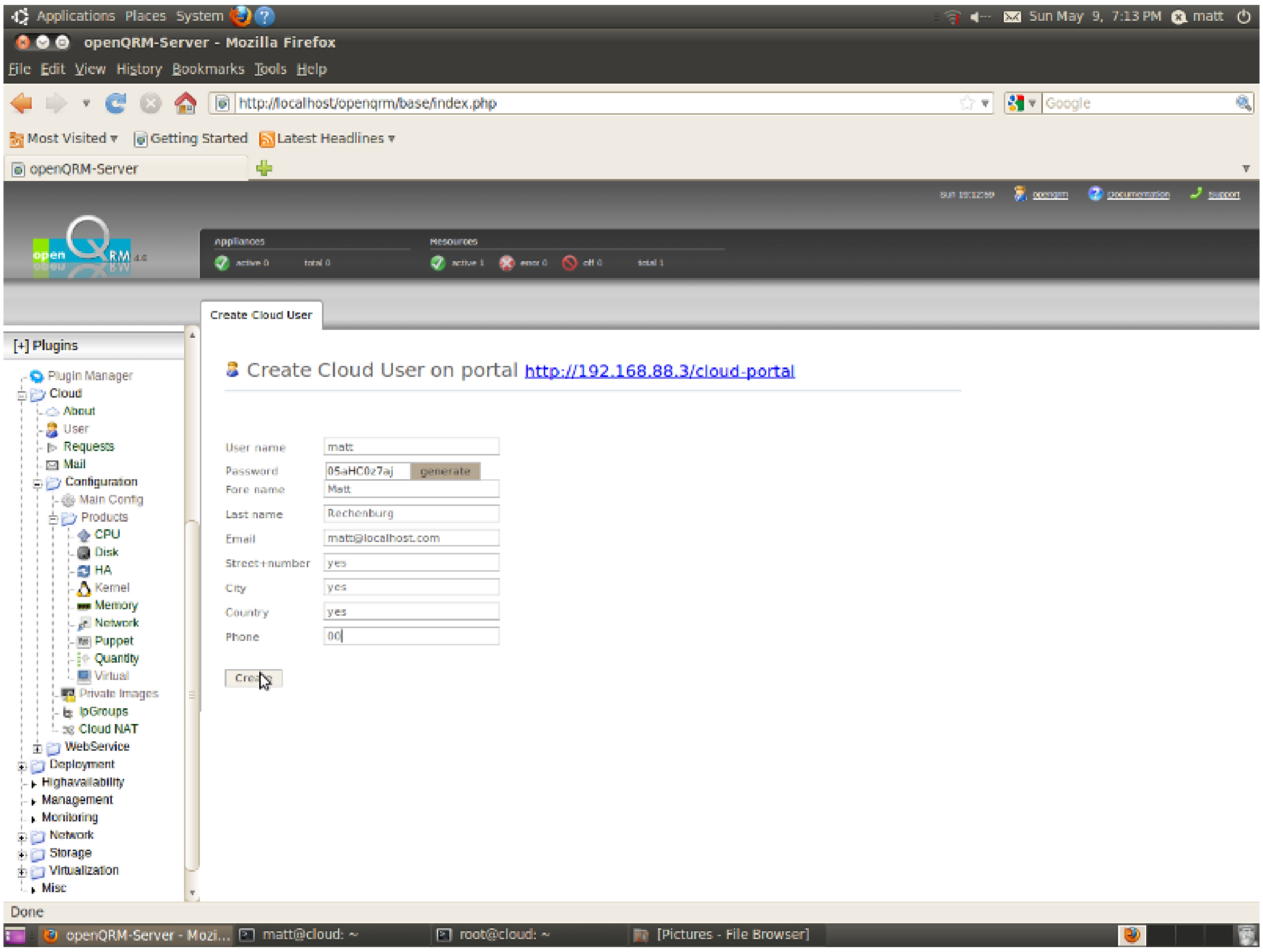
Cloud Image Selector for portal http://192.168.88.3/cloud-portal

order by ID ASC offset 0 limit 20 refresh

1 - 2 / 2

ID	Name	Version	Deployment type	Assign to
2	ubuntu64		lvm-nfs-deployment	All
3	debian64		lvm-nfs-deployment	All

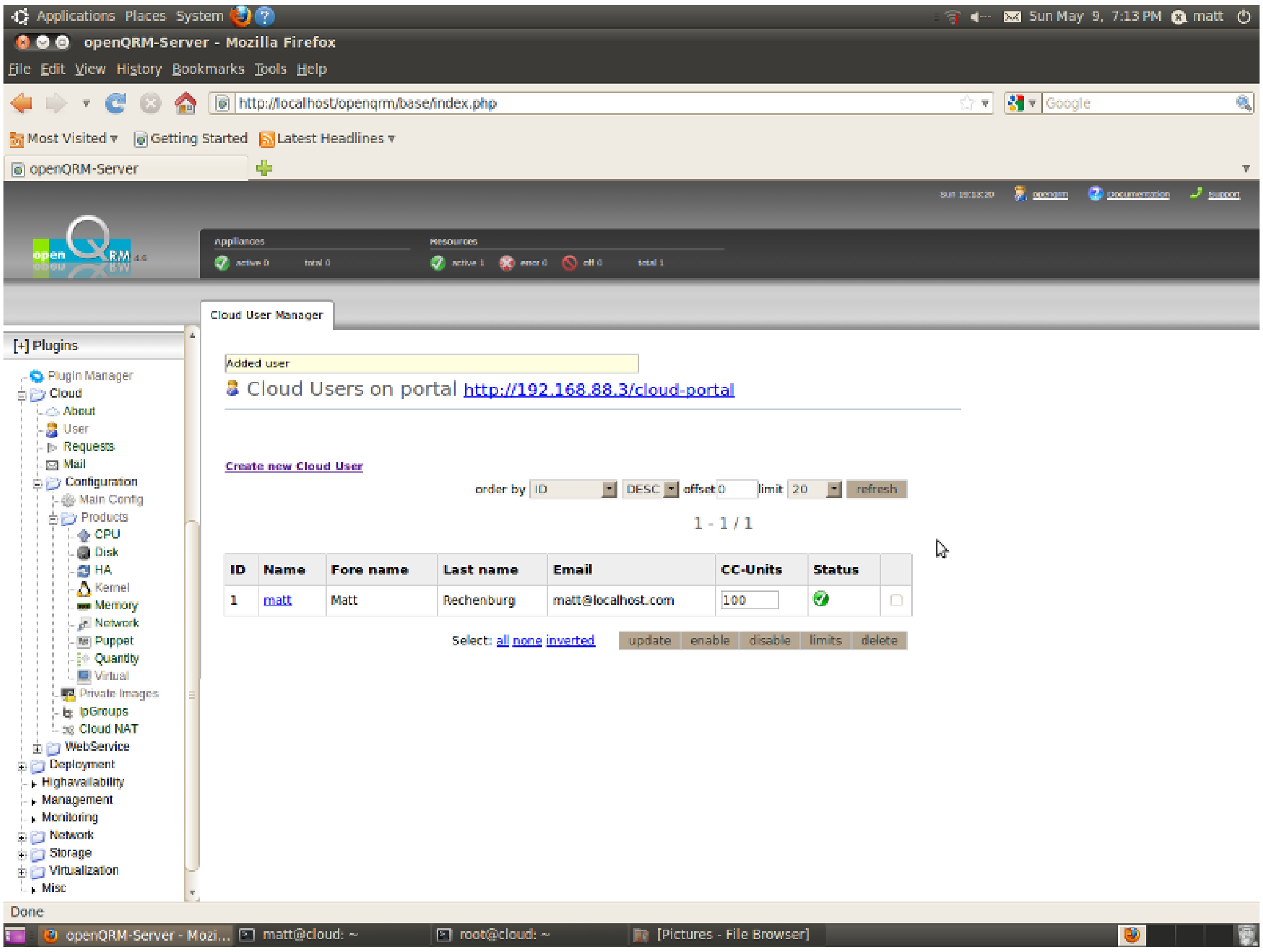
Following step is to create one or more CloudUser. Go to Plugins → Cloud → User and add a new Clouduser with a valid email address.


Applications Places System
openQRM-Server - Mozilla Firefox
http://localhost/openqrm/base/index.php
Create Cloud User
[+] Plugins

Create Cloud User on portal http://192.168.88.3/cloud-portal

User name	matt
Password	05aHC0z7aj
Fore name	Matt
Last name	Rechenburg
Email	matt@localhost.com
Street+number	yes
City	yes
Country	yes
Phone	00

The CloudUser List now looks like the screenshot below :



Added user

Cloud Users on portal http://192.168.88.3/cloud-portal

Create new Cloud User

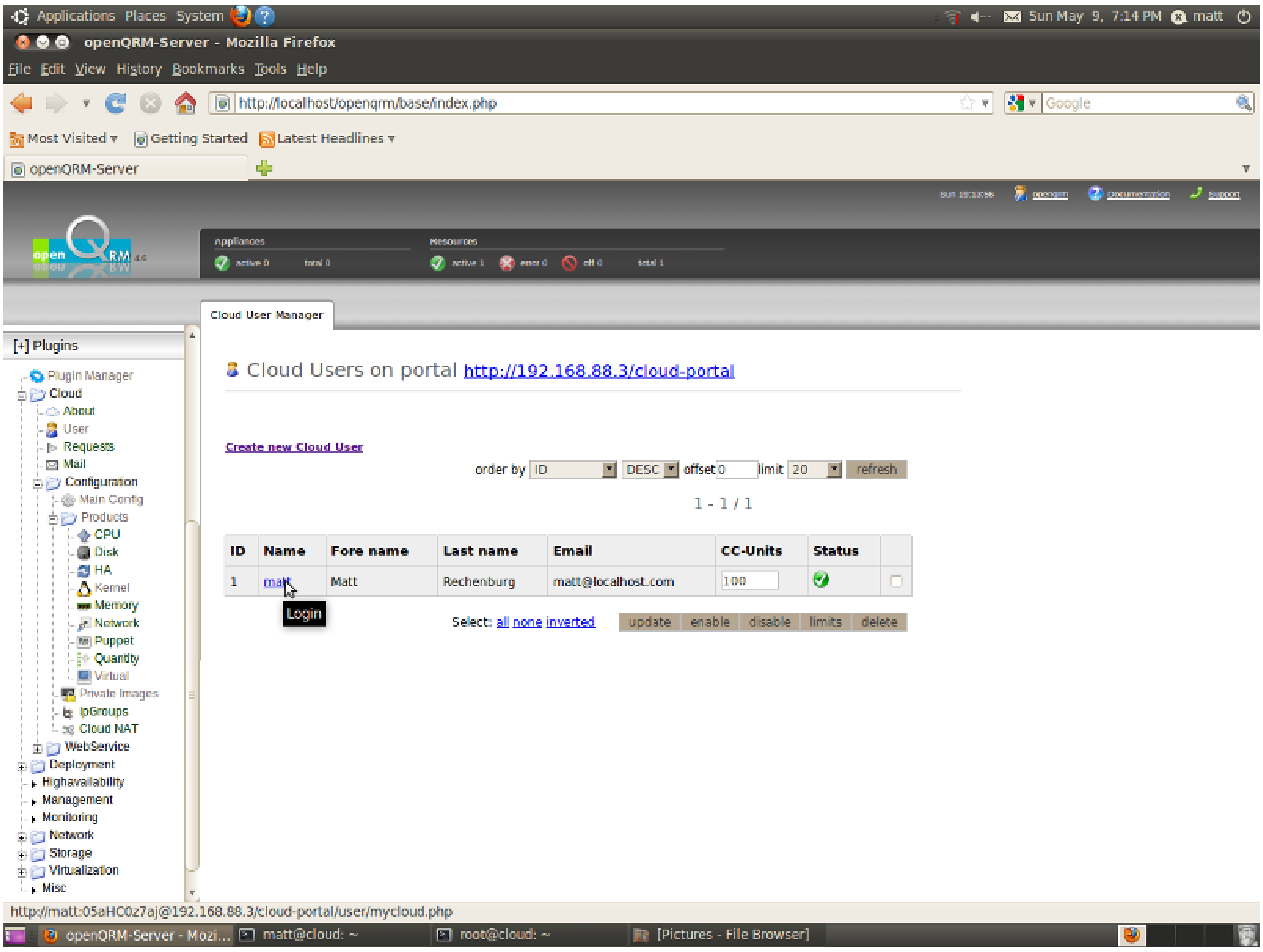
order by ID DESC offset 0 limit 20 refresh

1 - 1 / 1

ID	Name	Fore name	Last name	Email	CC-Units	Status	
1	matt	Matt	Rechenburg	matt@localhost.com	100	✓	☐

Select: all none inverted update enable disable limits delete

As the Cloud-Administrator you can simply login as a specific CloudUser by clicking on the CloudUser name.



Cloud Users on portal http://192.168.88.3/cloud-portal

Create new Cloud User

order by ID DESC offset 0 limit 20 refresh

1 - 1 / 1

ID	Name	Fore name	Last name	Email	CC-Units	Status	
1	matt	Matt	Rechenburg	matt@localhost.com	100	✓	☐

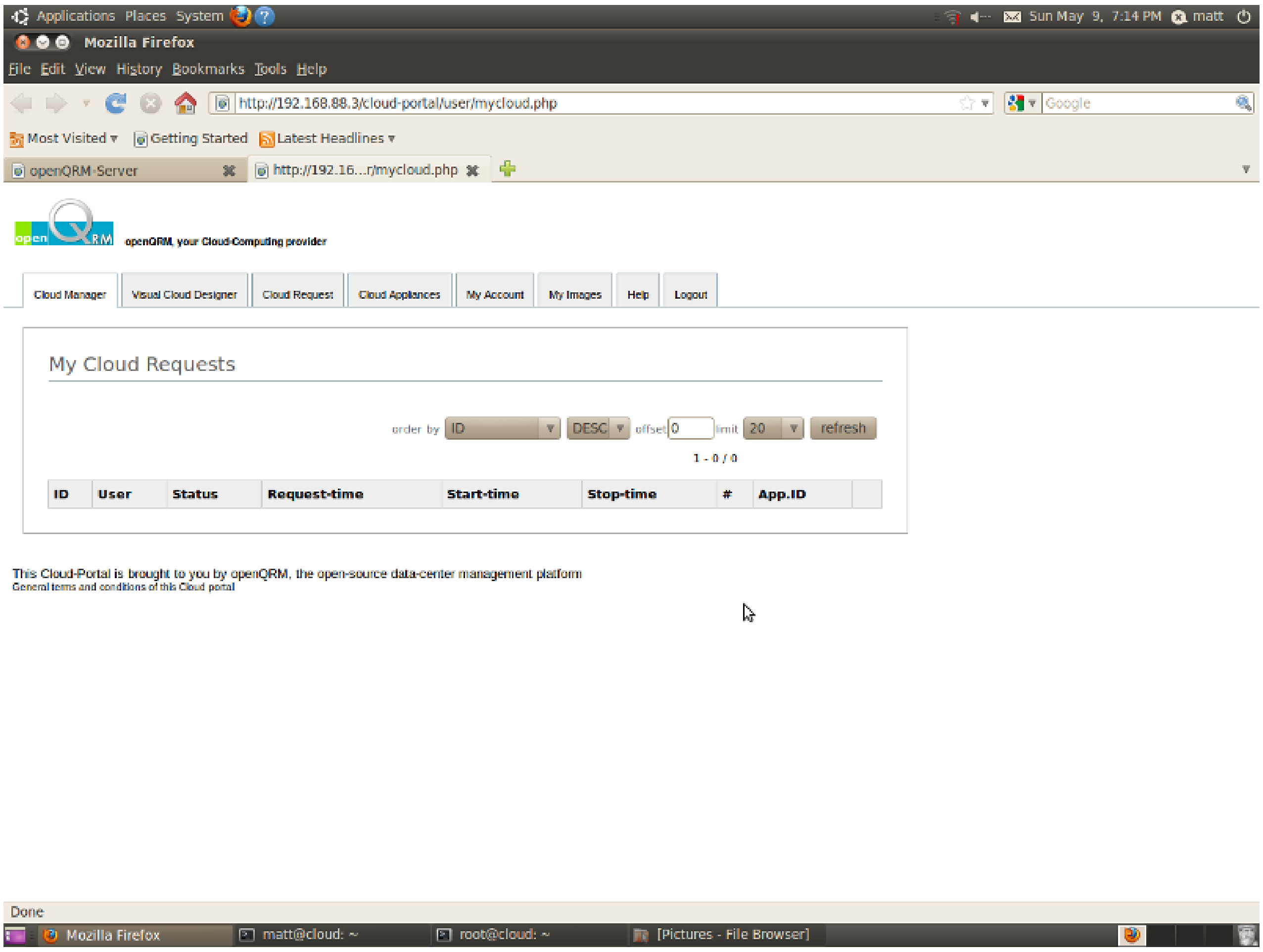
Login

Select: all none inverted

update enable disable limits delete

http://matt:05aHC0z7aj@192.168.88.3/cloud-portal/user/mycloud.php

This is the openQRM Cloud-Portal after login :

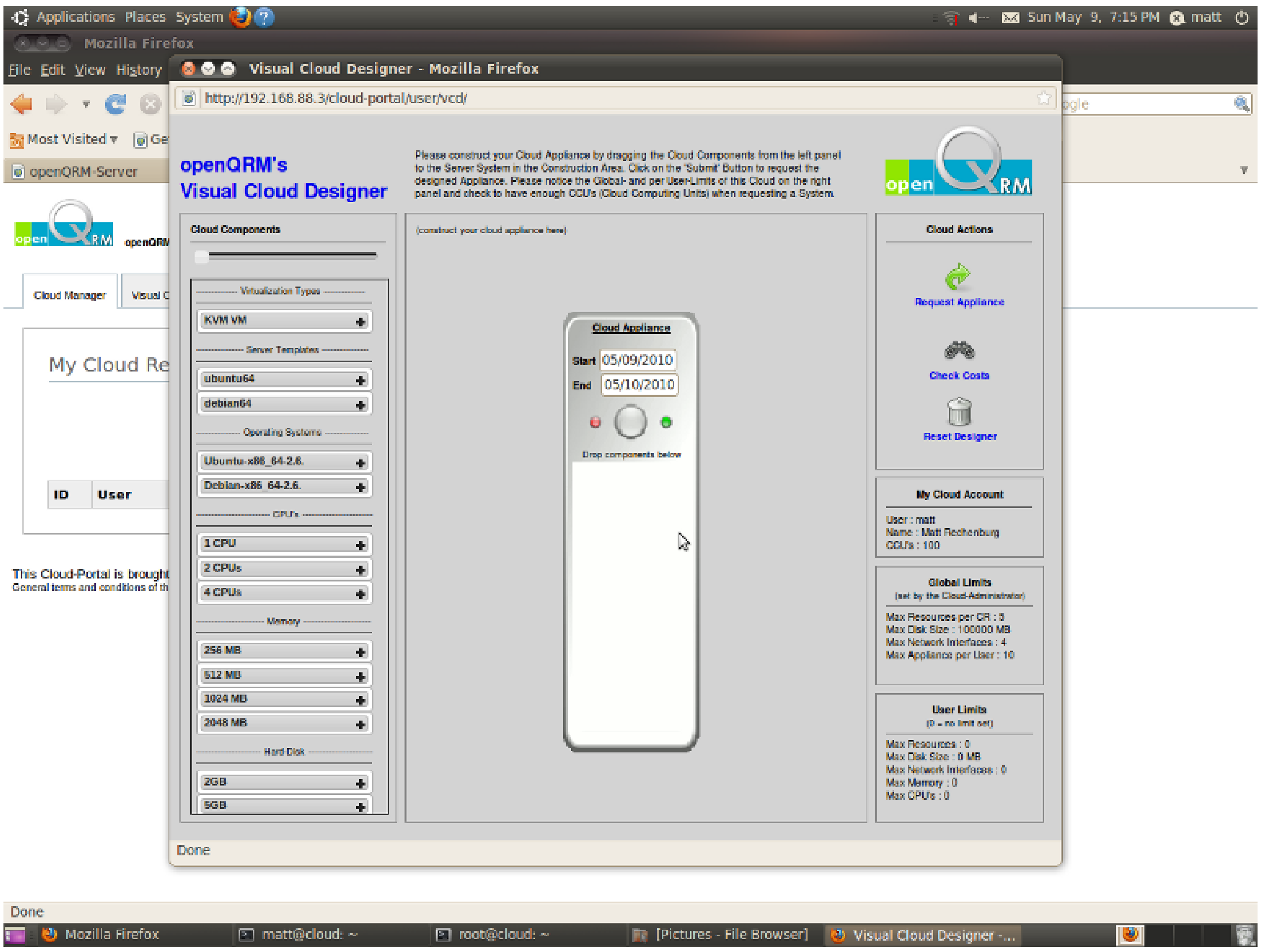

Applications Places System
Sun May 9, 7:14 PM matt
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://192.168.88.3/cloud-portal/user/mycloud.php
Most Visited Getting Started Latest Headlines
openQRM-Server
http://192.16...r/mycloud.php
openQRM, your Cloud-Computing provider
Cloud Manager
Visual Cloud Designer
Cloud Request
Cloud Appliances
My Account
My Images
Help
Logout
order by ID DESC offset 0 limit 20 refresh
1 - 0 / 0
Done
Mozilla Firefox
matt@cloud: ~
root@cloud: ~
[Pictures - File Browser]

My Cloud Requests

ID	User	Status	Request-time	Start-time	Stop-time	#	App.ID

This Cloud-Portal is brought to you by openQRM, the open-source data-center management platform
General terms and conditions of this Cloud portal

Click on the 2. Tab "Visual Cloud Designer"



http://192.168.88.3/cloud-portal/user/vcd/

openQRM's Visual Cloud Designer

Please construct your Cloud Appliance by dragging the Cloud Components from the left panel to the Server System in the Construction Area. Click on the 'Submit' Button to request the designed Appliance. Please notice the Global- and per User-Limits of this Cloud on the right panel and check to have enough CCU's (Cloud Computing Units) when requesting a System.

Start 05/09/2010
End 05/10/2010

My Cloud Account

User : matt
Name : Matt Rechenburg
CCU's : 100

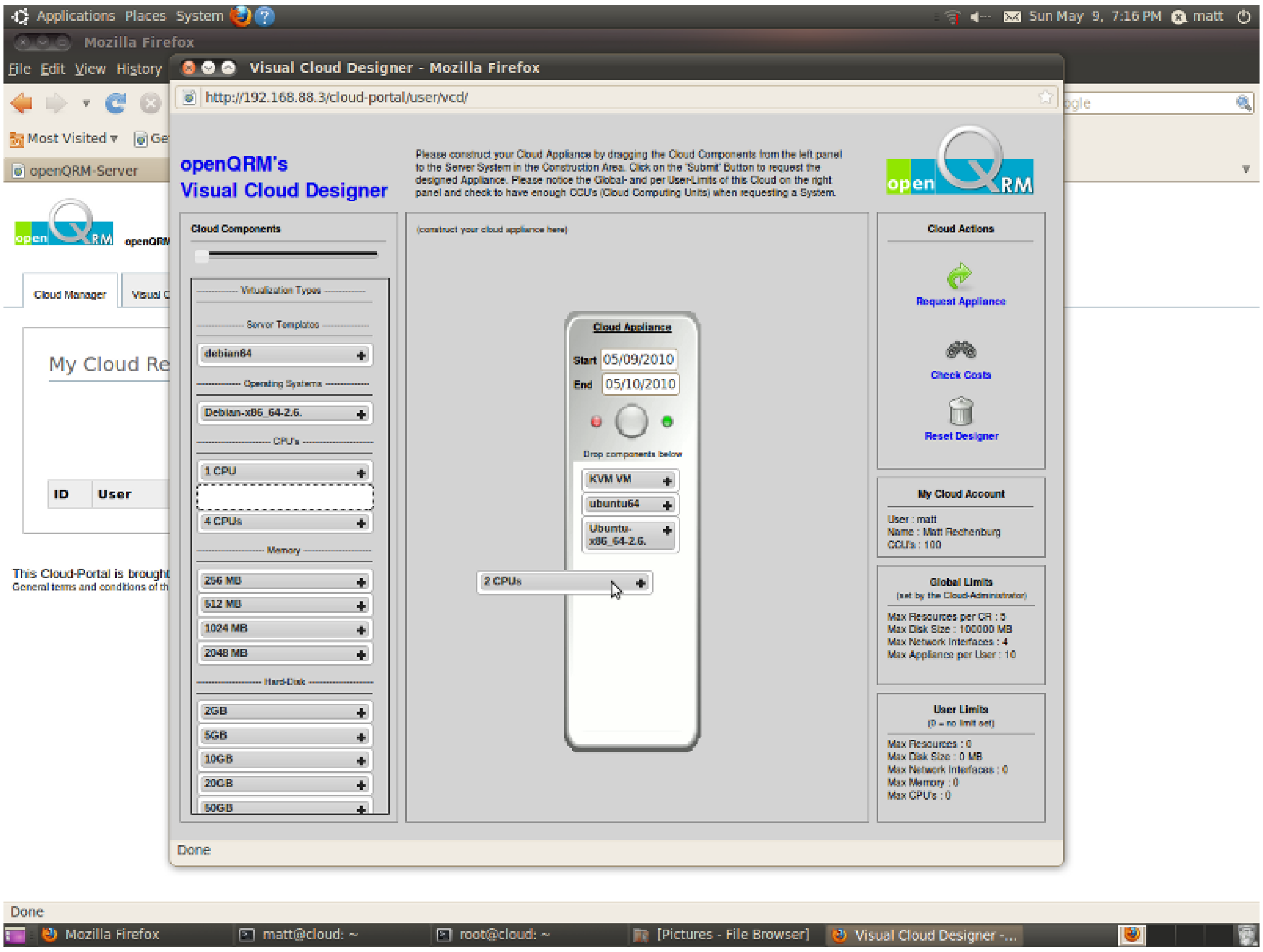
Global Limits
(set by the Cloud-Administrator)

Max Resources per CR : 5
Max Disk Size : 100000 MB
Max Network Interfaces : 4
Max Appliance per User : 10

User Limits
(0 = no limit set)

Max Resources : 0
Max Disk Size : 0 MB
Max Network Interfaces : 0
Max Memory : 0
Max CPU's : 0

The Visual Cloud Designer shows all components available in the Cloud. You can now start constructing your Cloud-Appliance by drag-and-drop.



Please construct your Cloud Appliance by dragging the Cloud Components from the left panel to the Server System in the Construction Area. Click on the 'Submit' Button to request the designed Appliance. Please notice the Global- and per User-Limits of this Cloud on the right panel and check to have enough CCU's (Cloud Computing Units) when requesting a System.

This Cloud-Portal is brought
General terms and conditions of th

My Cloud Account

User : matt
Name : Matt Rechenburg
CCU's : 100

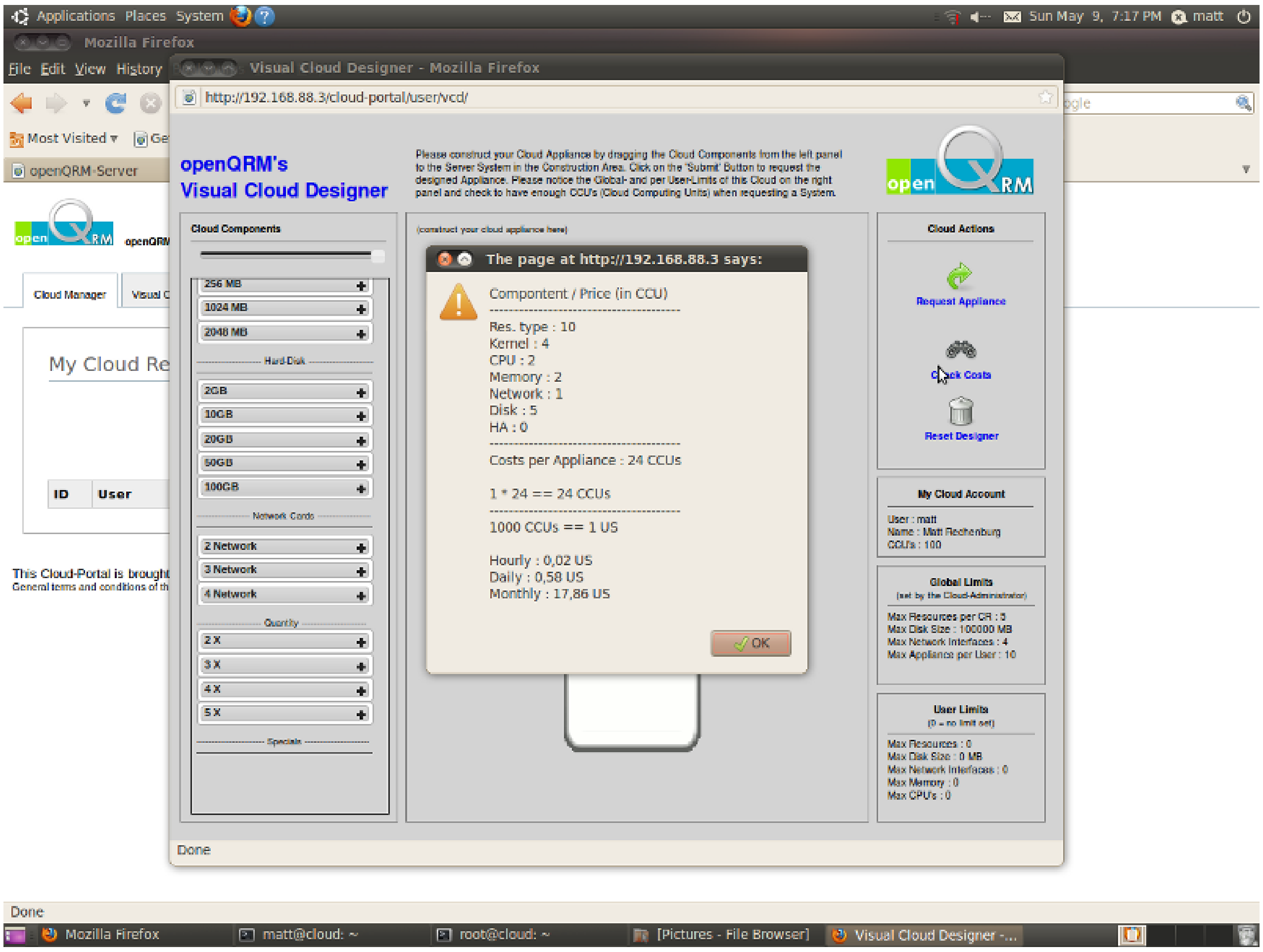
Global Limits
(set by the Cloud-Administrator)

Max Resources per CR : 5
Max Disk Size : 100000 MB
Max Network Interfaces : 4
Max Appliance per User : 10

User Limits
(0 = no limit set)

Max Resources : 0
Max Disk Size : 0 MB
Max Network Interfaces : 0
Max Memory : 0
Max CPU's : 0

Then check the costs for this Appliance (hourly, daily and monthly).



Please construct your Cloud Appliance by dragging the Cloud Components from the left panel to the Server System in the Construction Area. Click on the 'Submit' Button to request the designed Appliance. Please notice the Global- and per User-Limits of this Cloud on the right panel and check to have enough CCU's (Cloud Computing Units) when requesting a System.

(construct your cloud appliance here)

```
The page at http://192.168.88.3 says:

Compontent / Price (in CCU)
----------------------------------------
Res. type : 10
Kernel : 4
CPU : 2
Memory : 2
Network : 1
Disk : 5
HA : 0
----------------------------------------
Costs per Appliance : 24 CCUs

1 * 24 == 24 CCUs
----------------------------------------
1000 CCUs == 1 US

Hourly : 0,02 US
Daily : 0,58 US
Monthly : 17,86 US
```

My Cloud Account

User : matt
Name : Matt Rechenburg
CCU's : 100

Global Limits
(set by the Cloud-Administrator)

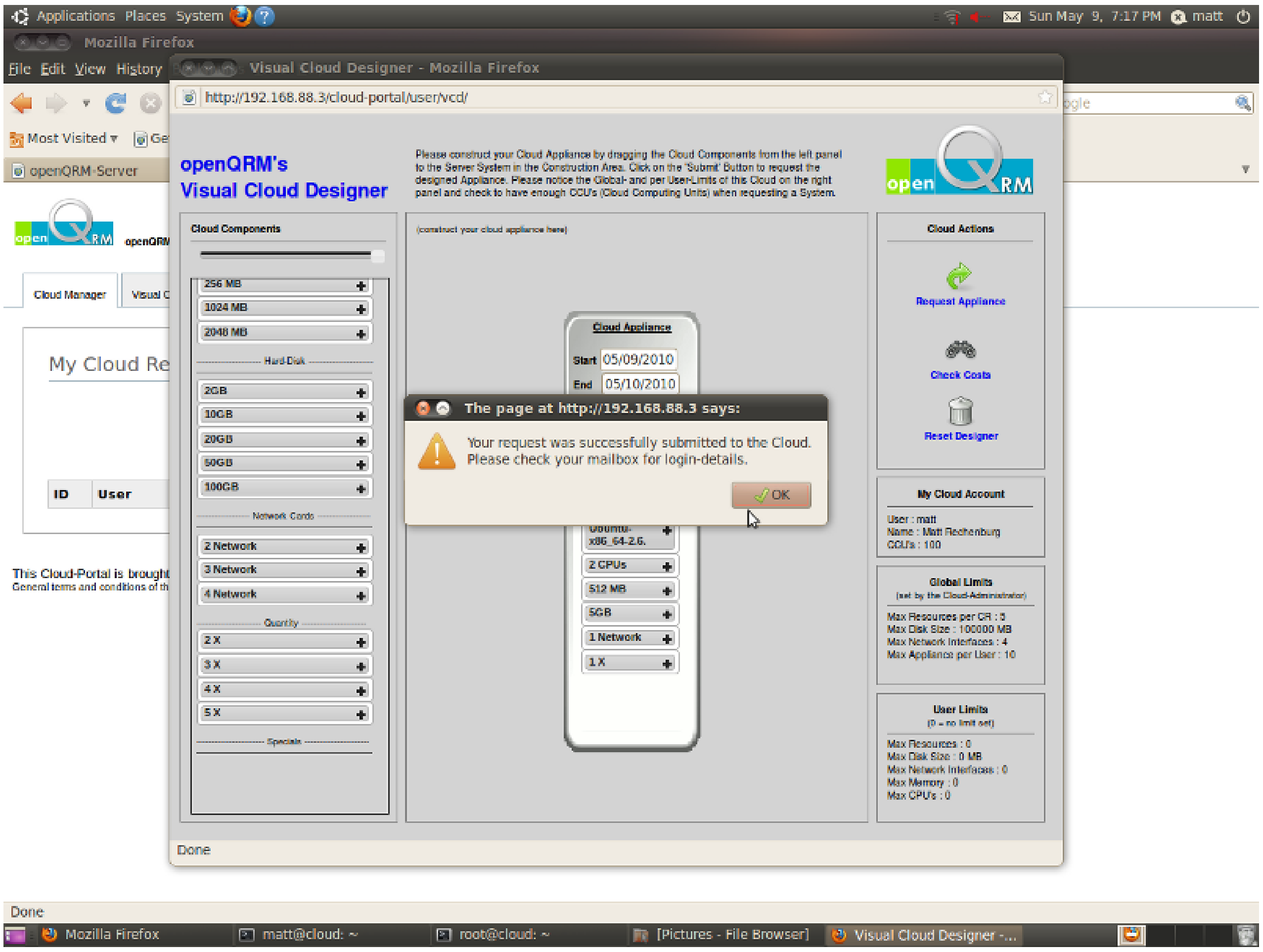
Max Resources per CR : 5
Max Disk Size : 100000 MB
Max Network Interfaces : 4
Max Appliance per User : 10

User Limits
(0 = no limit set)

Max Resources : 0
Max Disk Size : 0 MB
Max Network Interfaces : 0
Max Memory : 0
Max CPU's : 0

This Cloud-Portal is brought
General terms and conditions of th

And with a single click request this custom system from the openQRM Cloud.



http://192.168.88.3/cloud-portal/user/vcd/

openQRM's Visual Cloud Designer

Please construct your Cloud Appliance by dragging the Cloud Components from the left panel to the Server System in the Construction Area. Click on the 'Submit' Button to request the designed Appliance. Please notice the Global- and per User-Limits of this Cloud on the right panel and check to have enough CCU's (Cloud Computing Units) when requesting a System.

(construct your cloud appliance here)

Cloud Appliance

Start 05/09/2010

End 05/10/2010

The page at http://192.168.88.3 says:

Your request was successfully submitted to the Cloud.
Please check your mailbox for login-details.

My Cloud Account

User : matt
Name : Matt Rechenburg
CCU's : 100

Global Limits
(set by the Cloud-Administrator)

Max Resources per CR : 5
Max Disk Size : 100000 MB
Max Network Interfaces : 4
Max Appliance per User : 10

User Limits
(0 = no limit set)

Max Resources : 0
Max Disk Size : 0 MB
Max Network Interfaces : 0
Max Memory : 0
Max CPU's : 0

11. Enjoy

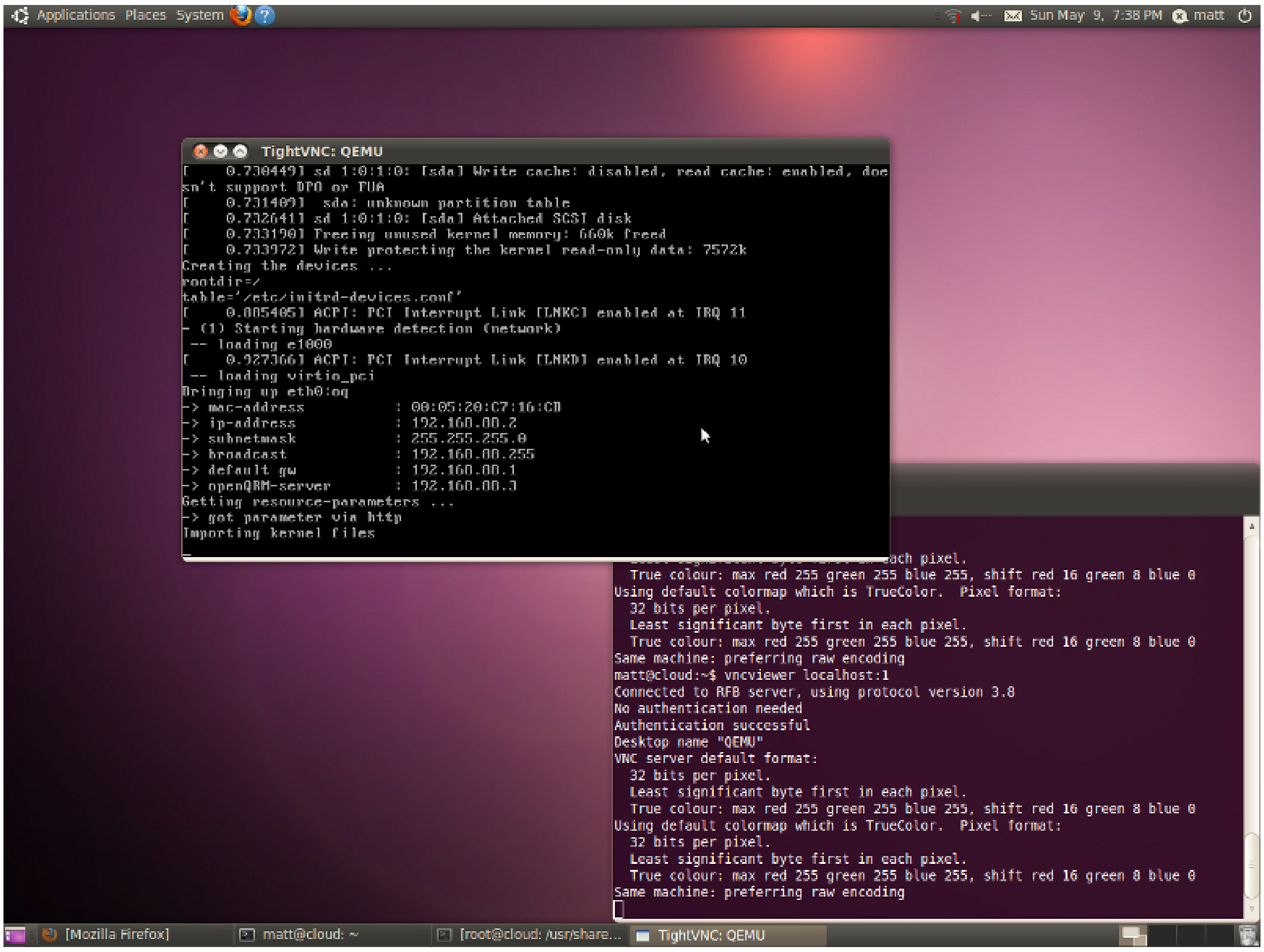
To get access to the KVM VM console please install the "xtightvncviewer" package.

```
matt@cloud:~$ sudo apt-get install xtightvncviewer
Reading package lists... Done
...<snip>
Setting up xtightvncviewer (1.3.9-6) ...
update-alternatives: using /usr/bin/xtightvncviewer to provide /usr/bin/vncviewer (vncviewer) in auto mode.
matt@cloud:~$
```

To VNC-Login to the first Cloud-Appliance (KVM VM) please run :

```
matt@cloud:~$ vncviewer localhost:1
```

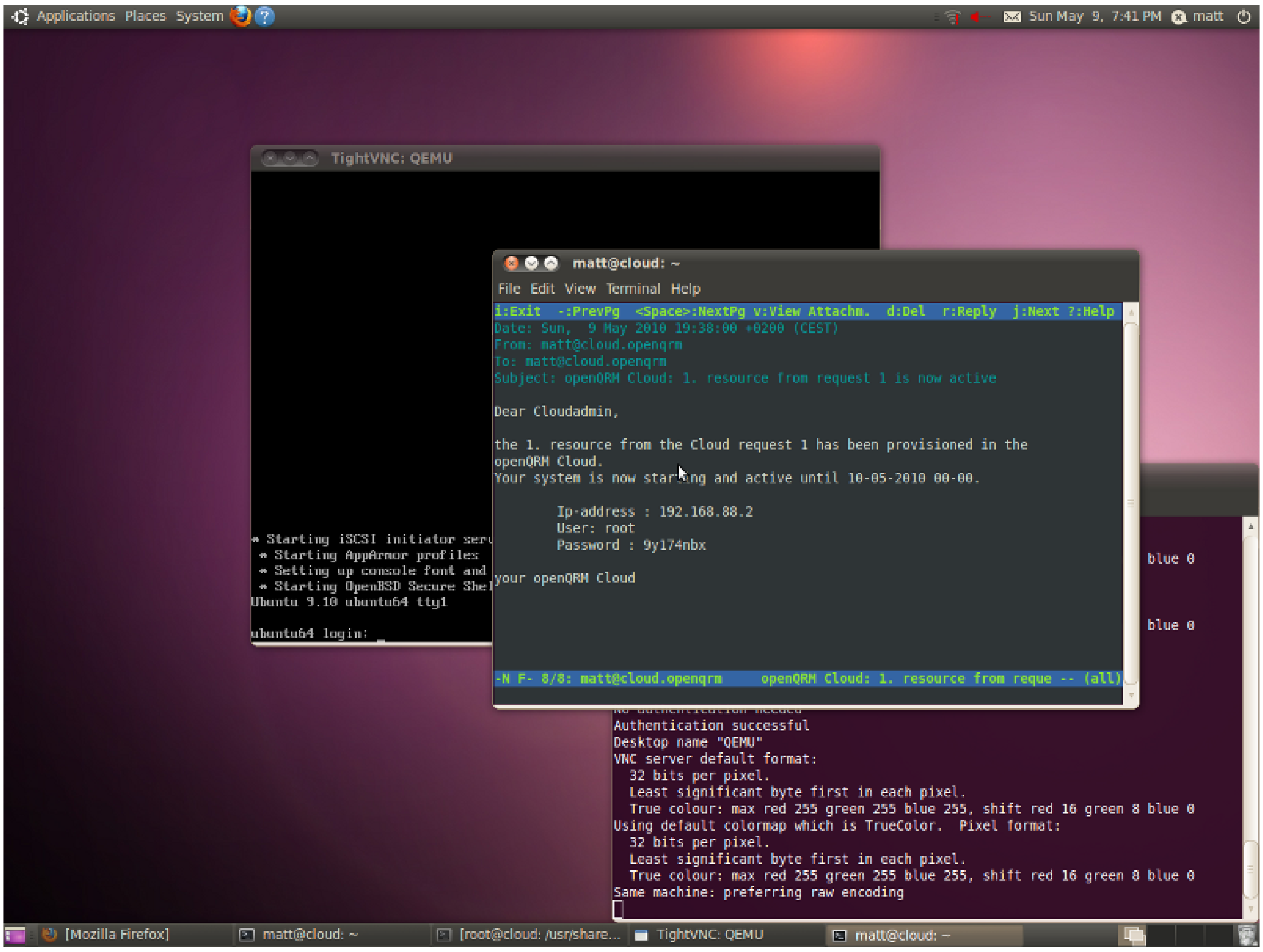
Here a screenshot of the booting Cloud-Appliance starting in a KVM VM.



```
[    0.730449] sd 1:0:1:0: [sda] Write cache: disabled, read cache: enabled, doe
sn't support DPO or FUA
[    0.731409]  sda: unknown partition table
[    0.732641] sd 1:0:1:0: [sda] Attached SCSI disk
[    0.733190] Freeing unused kernel memory: 660k freed
[    0.733972] Write protecting the kernel read-only data: 7572k
Creating the devices ...
rootdir=/
table='/etc/initrd-devices.conf'
[    0.805405] ACPI: PCI Interrupt Link [LNKC] enabled at IRQ 11
- (1) Starting hardware detection (network)
 -- loading e1000
[    0.927366] ACPI: PCI Interrupt Link [LNKD] enabled at IRQ 10
 -- loading virtio_pci
Bringing up eth0:oq
-> mac-address          : 00:05:20:C7:16:CD
-> ip-address           : 192.168.88.2
-> subnetmask           : 255.255.255.0
-> broadcast            : 192.168.88.255
-> default gw           : 192.168.88.1
-> openQRM-server       : 192.168.88.3
Getting resource-parameters ...
-> got parameter via http
Importing kernel files
_
```

```
                                each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Same machine: preferring raw encoding
matt@cloud:~$ vncviewer localhost:1
Connected to RFB server, using protocol version 3.8
No authentication needed
Authentication successful
Desktop name "QEMU"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Same machine: preferring raw encoding
```

And here the mail the openQRM Cloud sent to the CloudUser with the ip-address and login credentials.



```
i:Exit  -:PrevPg  <Space>:NextPg v:View Attachm.  d:Del  r:Reply  j:Next ?:Help
Date: Sun,  9 May 2010 19:38:00 +0200 (CEST)
From: matt@cloud.openqrm
To: matt@cloud.openqrm
Subject: openQRM Cloud: 1. resource from request 1 is now active

Dear Cloudadmin,

the 1. resource from the Cloud request 1 has been provisioned in the
openQRM Cloud.
Your system is now starting and active until 10-05-2010 00-00.

        Ip-address : 192.168.88.2
        User: root
        Password : 9y174nbx

your openQRM Cloud

-N F- 8/8: matt@cloud.openqrm      openQRM Cloud: 1. resource from reque -- (all)
```

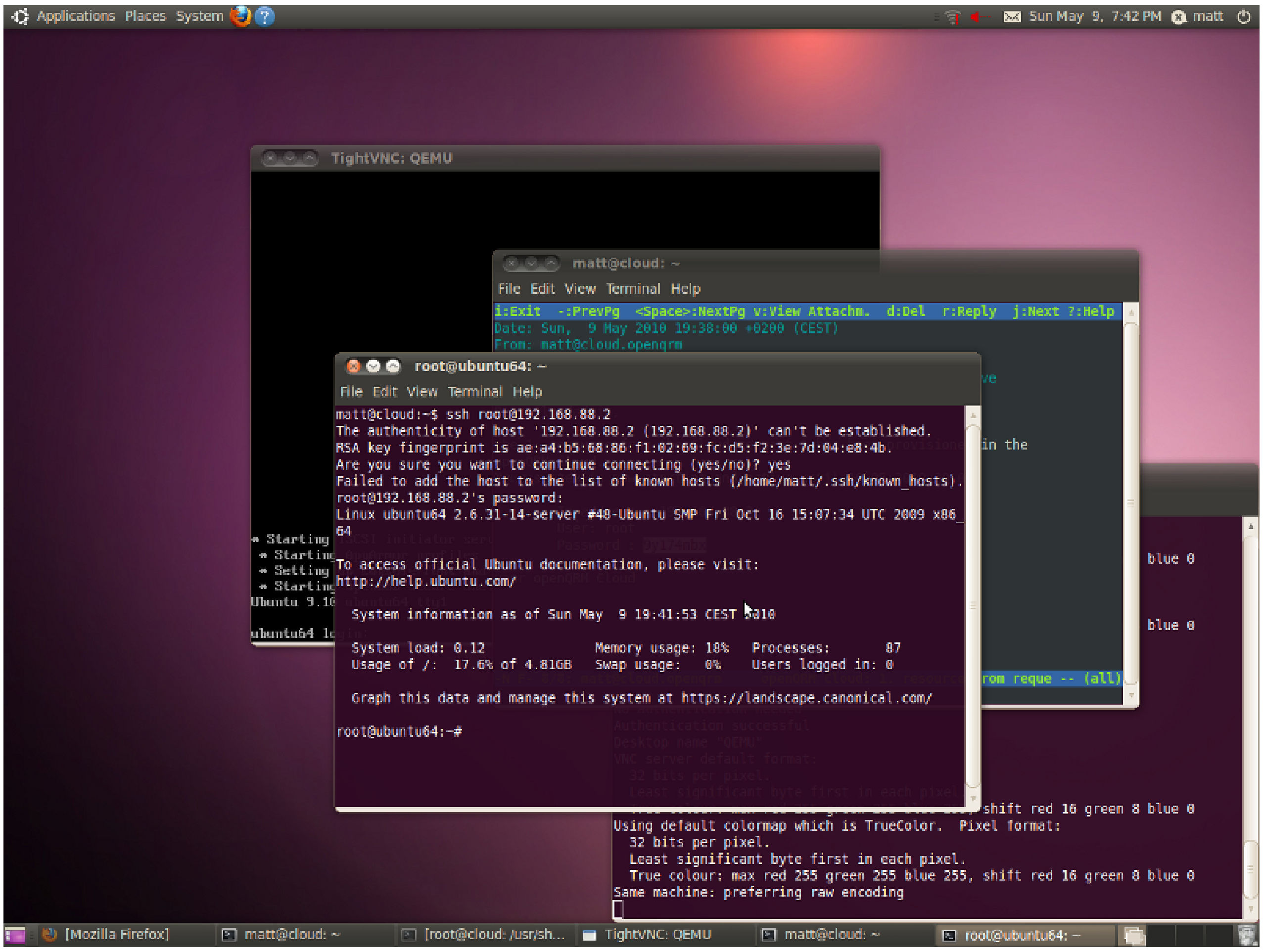
```
* Starting iSCSI initiator serv
 * Starting AppArmor profiles
 * Setting up console font and
 * Starting OpenBSD Secure Shel
Ubuntu 9.10 ubuntu64 tty1

ubuntu64 login: _
```

```
blue 0

blue 0
Authentication successful
Desktop name "QEMU"
VNC server default format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Same machine: preferring raw encoding
```

We can now login and use the requested Cloud-Appliance.



```
i:Exit  -:PrevPg  <Space>:NextPg v:View Attachm.  d:Del  r:Reply  j:Next ?:Help
Date: Sun,  9 May 2010 19:38:00 +0200 (CEST)
From: matt@cloud.openqrm
```

```
matt@cloud:~$ ssh root@192.168.88.2
The authenticity of host '192.168.88.2 (192.168.88.2)' can't be established.
RSA key fingerprint is ae:a4:b5:68:86:f1:02:69:fc:d5:f2:3e:7d:04:e8:4b.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/home/matt/.ssh/known_hosts).
root@192.168.88.2's password:
Linux ubuntu64 2.6.31-14-server #48-Ubuntu SMP Fri Oct 16 15:07:34 UTC 2009 x86_
64

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/

  System information as of Sun May  9 19:41:53 CEST 2010

  System load: 0.12              Memory usage: 10%   Processes:       87
  Usage of /:  17.6% of 4.81GB   Swap usage:   0%    Users logged in: 0

  Graph this data and manage this system at https://landscape.canonical.com/

root@ubuntu64:~#
```

```
shift red 16 green 8 blue 0
Using default colormap which is TrueColor.  Pixel format:
  32 bits per pixel.
  Least significant byte first in each pixel.
  True colour: max red 255 green 255 blue 255, shift red 16 green 8 blue 0
Same machine: preferring raw encoding
```

Enjoy your openQRM Cloud !!

12. How to continue from here

- Separating Storage, Hypvervisors and openQRM on dedicated Systems
- openQRM Server HA Setup
- Adding more Virtualization Host from different types
- Adding Physical Systems
- Adding more Storage Systems
- Enabling automatic Monitoring
- IP- and Network-Management
- Cloud-Billing
- Cloud Integration / SOAP WebService
- … and more

The best source of informations how to scale your basic openQRM Setup to a distributed, flexible and robust openQRM managed Datacenter Environment is the detailed technical documentation available at http://www.openqrm-enterprise.com/news/details /article/in-depth-documentation-of-openqrm-available.html [http://www.openqrm-enterprise.com/news/details/article/in-depth-documentation-of-openqrm-available.html].

There is also an active community maintaining the forums and mailing-lists of the openQRM Project.

For professional services and support please contact openQRM Enterprise - http://www.openqrm-enterprise.com [http://www.openqrm-enterprise.com]

Thanks

This HowTo is brought to you by openQRM Enterprise [http://www.openqrm-enterprise.com/]

openQRM Enterprise GmbH
Berrenrather Straße 188c
50937 Köln / Germany
Telefon : +49 (0) 221 995589-10
Fax : +49 (0) 221 995589-20
Mail : info@openqrm-enterprise.com