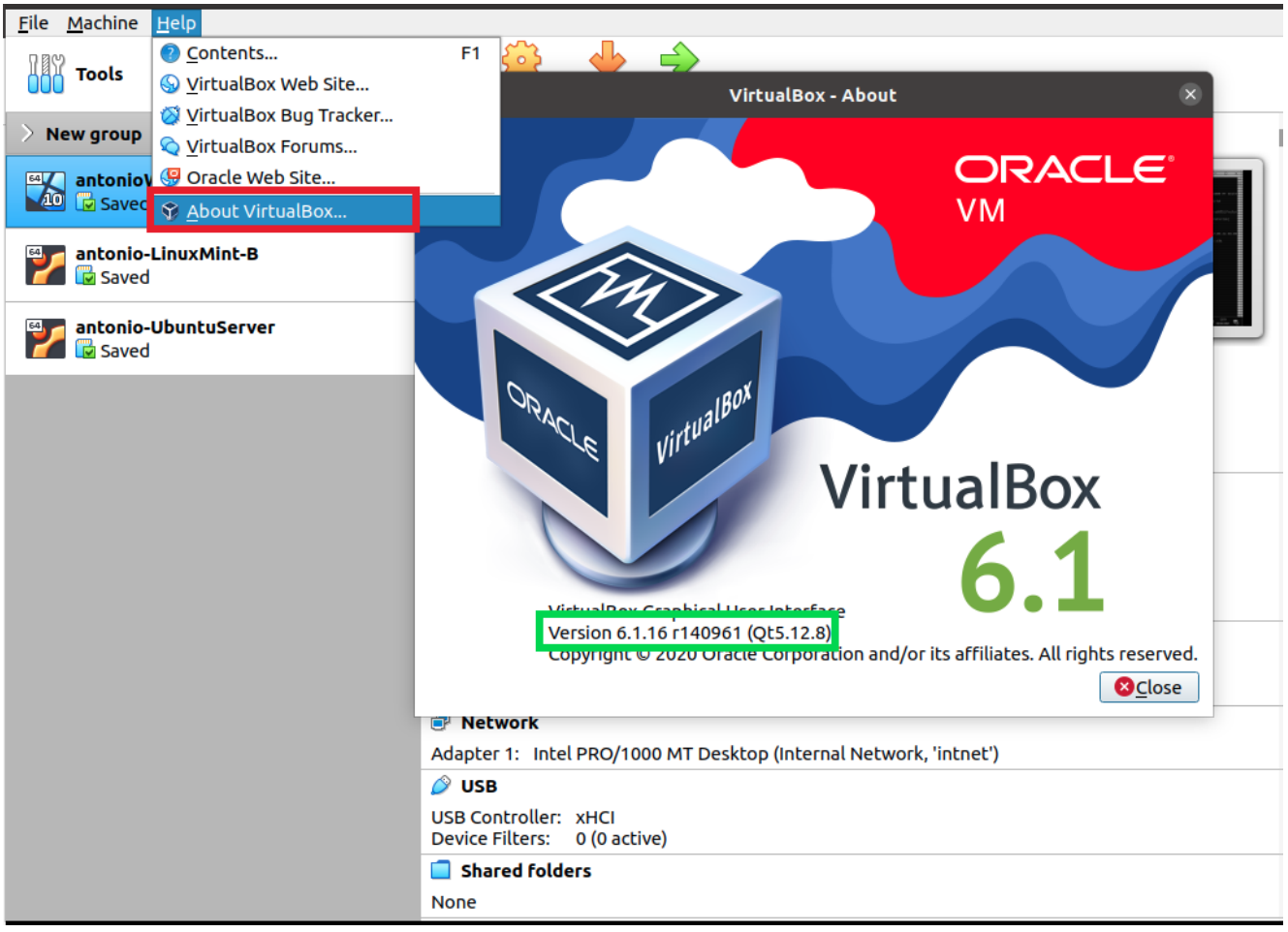
Instalacións

C) Instalación de Odoo nun Server

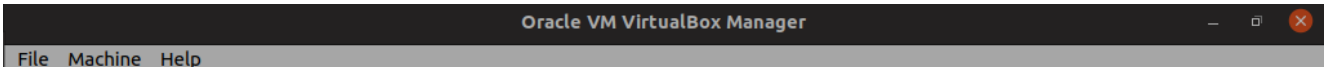
Instalar as Virtualbox Additions nunha máquina GNU/Linux Server

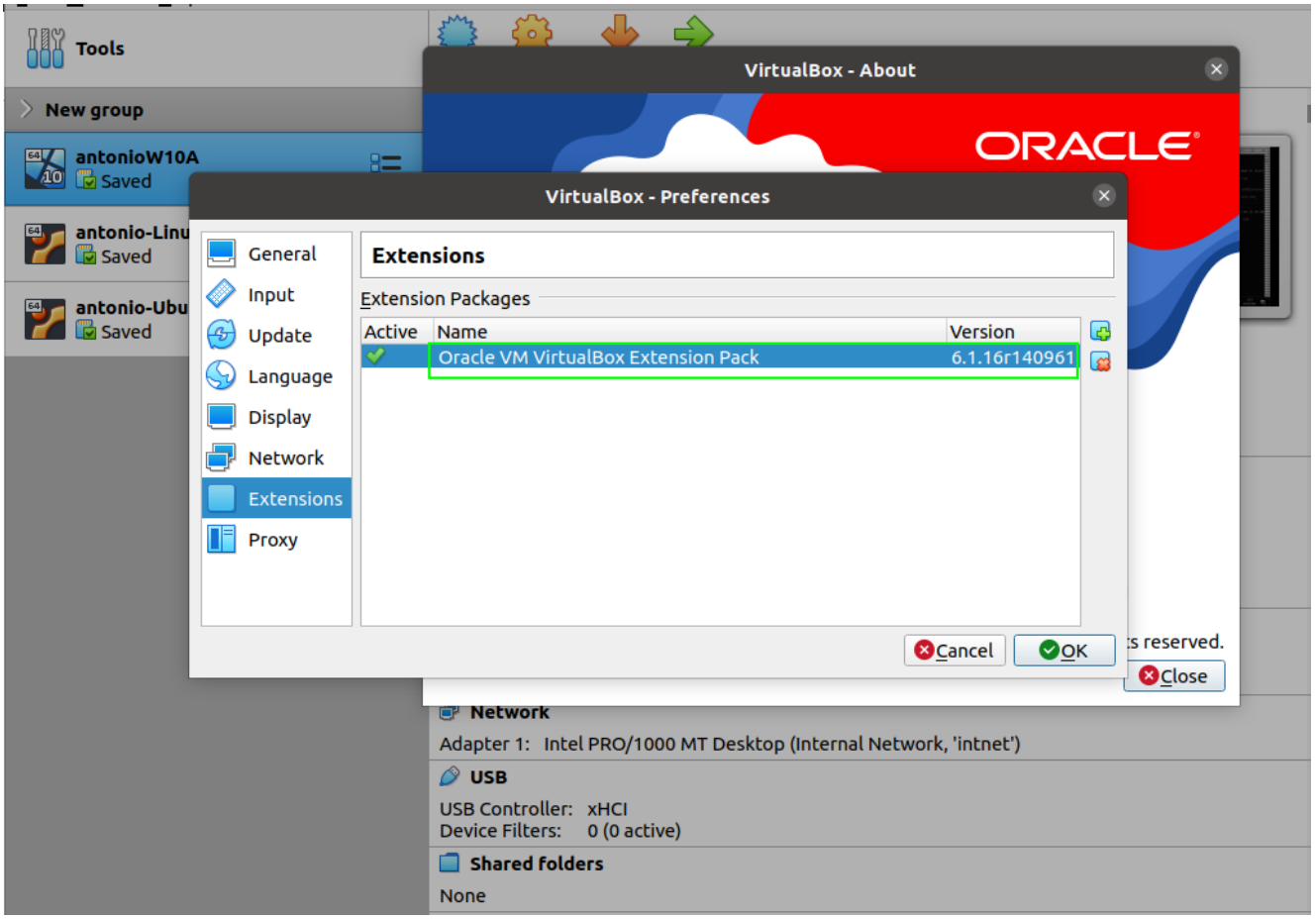
Parar instalar as Virtualbox Additions:

- Comprobamos a versión de Virtualbox

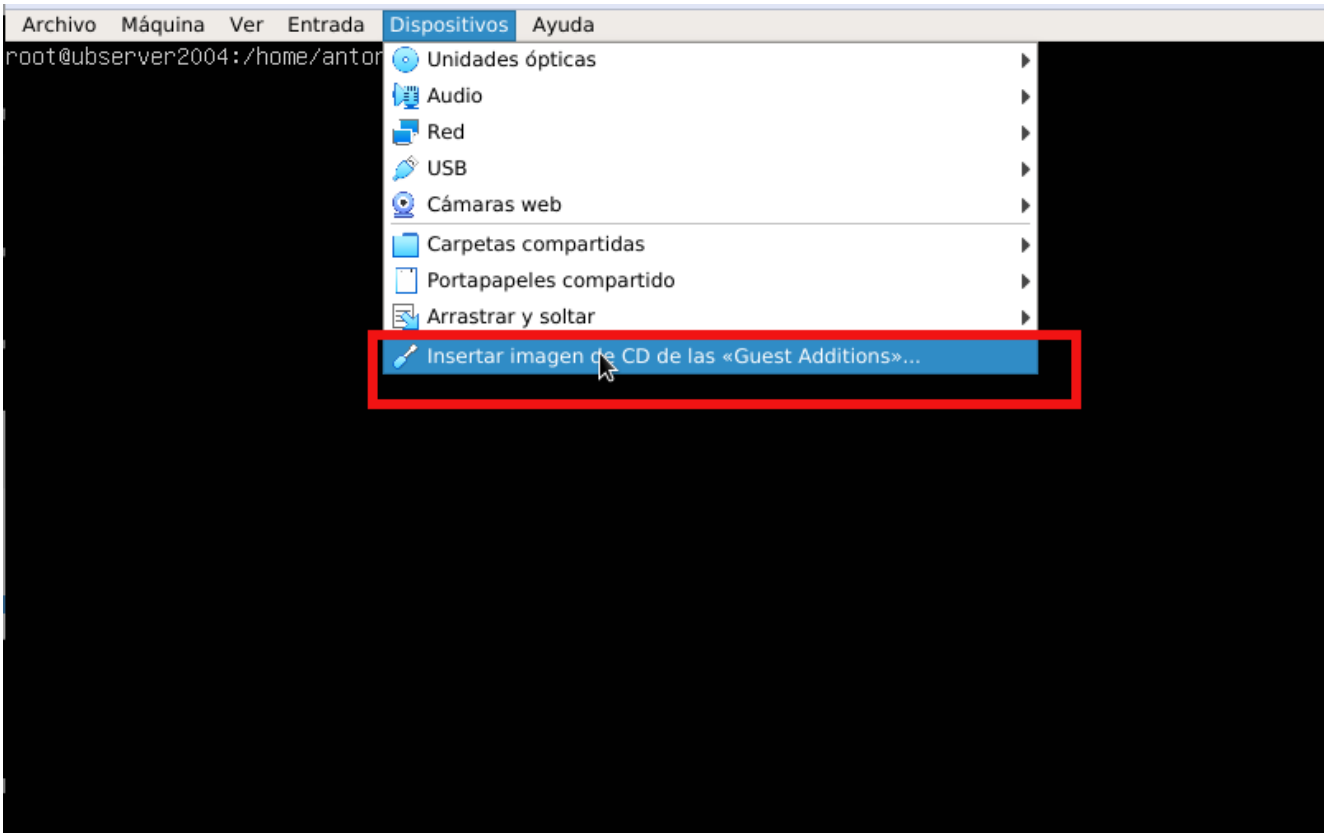


- Comprobamos a versión do "extension Pack" no Menu File >> Preferences:

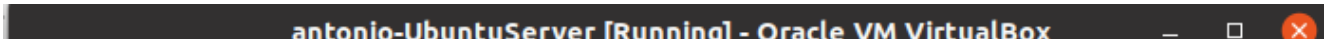


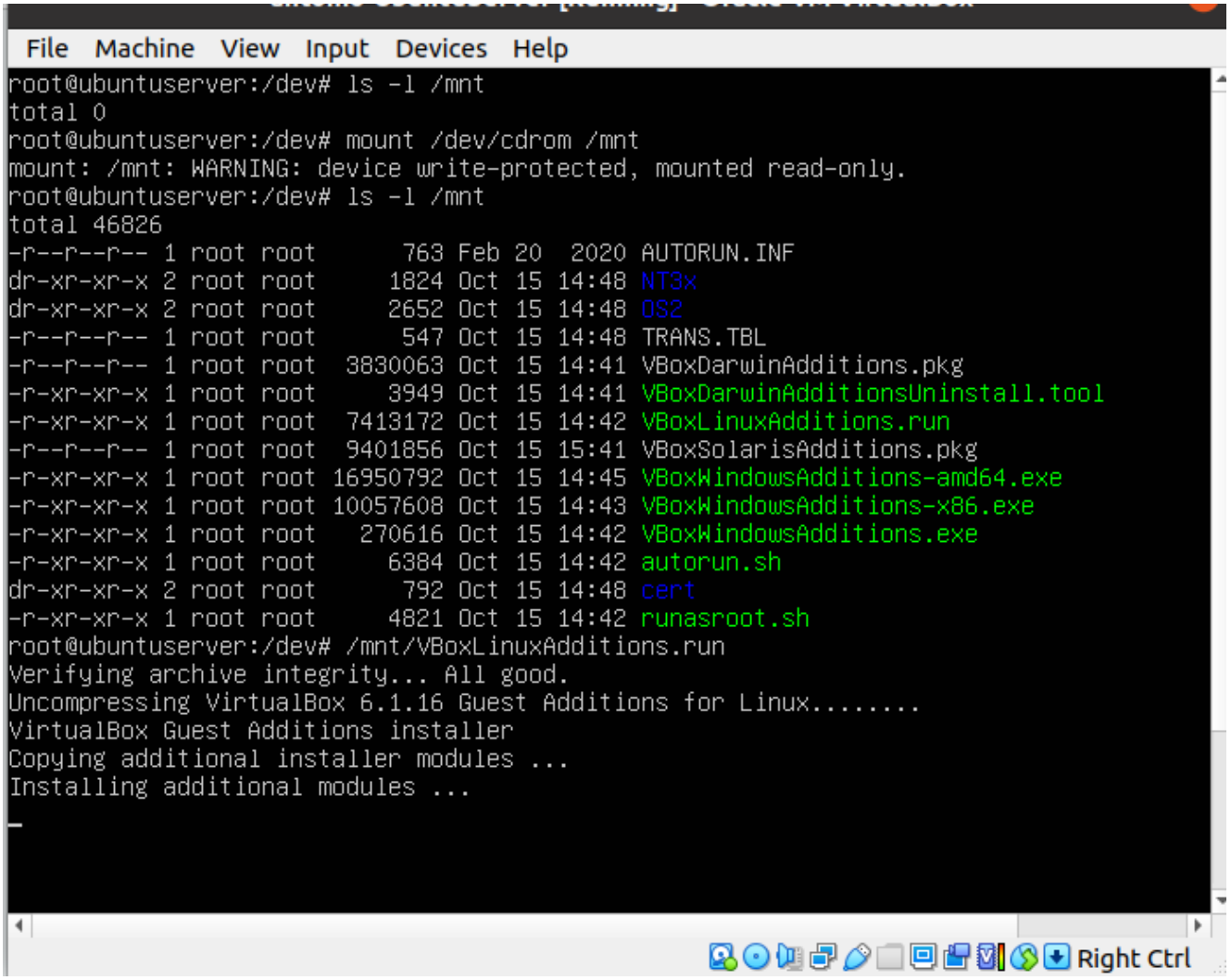


- Introducimos o CD das Virtualbox Additions



- Montamos o cd como se ve na captura seguinte, e instalamos as additions





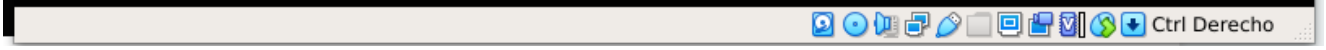
- Vemos que nos da un erro e nos suxire instalar gcc make e perl

```
root@ubserver2004:/dev/disk/by-id# cd /mnt
root@ubserver2004:/mnt# ls -l
total 46826
-r--r--r-- 1 root root      763 feb 20  2020 AUTORUN.INF
-r-xr-xr-x 1 root root     6384 oct 15 14:42 autorun.sh
dr-xr-xr-x 2 root root      792 oct 15 14:48 cert
dr-xr-xr-x 2 root root     1824 oct 15 14:48 NT3x
dr-xr-xr-x 2 root root     2652 oct 15 14:48 OS2
-r-xr-xr-x 1 root root     4821 oct 15 14:42 runasroot.sh
-r--r--r-- 1 root root      547 oct 15 14:48 TRANS.TBL
-r--r--r-- 1 root root  3830063 oct 15 14:41 VBoxDarwinAdditions.pkg
-r-xr-xr-x 1 root root     3949 oct 15 14:41 VBoxDarwinAdditionsUninstall.tool
-r-xr-xr-x 1 root root  7413172 oct 15 14:42 VBoxLinuxAdditions.run
-r--r--r-- 1 root root  9401856 oct 15 15:41 VBoxSolarisAdditions.pkg
-r-xr-xr-x 1 root root 16950792 oct 15 14:45 VBoxWindowsAdditions-amd64.exe
-r-xr-xr-x 1 root root   270616 oct 15 14:42 VBoxWindowsAdditions.exe
-r-xr-xr-x 1 root root 10057608 oct 15 14:43 VBoxWindowsAdditions-x86.exe
root@ubserver2004:/mnt# ./VBoxLinuxAdditions.run
Verifying archive integrity... All good.
Uncompressing VirtualBox 6.1.16 Guest Additions for Linux.....
VirtualBox Guest Additions installer
Copying additional installer modules ...
Installing additional modules ...
VirtualBox Guest Additions: Starting.
VirtualBox Guest Additions: Building the VirtualBox Guest Additions kernel
modules.  This may take a while.
VirtualBox Guest Additions: To build modules for other installed kernels, run
VirtualBox Guest Additions:   /sbin/rcvboxadd quicksetup <version>
VirtualBox Guest Additions: or
VirtualBox Guest Additions:   /sbin/rcvboxadd quicksetup all
VirtualBox Guest Additions: Building the modules for kernel 5.4.0-62-generic.
```

```
This system is currently not set up to build kernel modules.
Please install the gcc make perl packages from your distribution.
VirtualBox Guest Additions: Running kernel modules will not be replaced until
the system is restarted
```

- Instalamos os paquetes co comando apt install gcc make perl

```
root@ubserver2004:/mnt# apt install gcc make perl
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
perl ya está en su versión más reciente (5.30.0-9ubuntu0.2).
fijado perl como instalado manualmente.
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  linux-headers-5.4.0-62 linux-headers-5.4.0-62-generic linux-image-5.4.0-62-generic
  linux-modules-5.4.0-62-generic linux-modules-extra-5.4.0-62-generic
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-9 gcc-9 gcc-9-base libasan5
  libatomic1 libbinutils libc-dev-bin libc6 libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0
  libgcc-9-dev libisl22 libitm1 liblsan0 libmpc3 libquadmath0 libtsan0 libubsan1 linux-libc-dev
  manpages-dev
Paquetes sugeridos:
  binutils-doc cpp-doc gcc-9-locales gcc-multilib autoconf automake libtool flex bison gdb gcc-doc
  gcc-9-multilib gcc-9-doc glibc-doc make-doc
Se instalarán los siguientes paquetes NUEVOS:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-9 gcc gcc-9 gcc-9-base libasan5
  libatomic1 libbinutils libc-dev-bin libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0
  libgcc-9-dev libisl22 libitm1 liblsan0 libmpc3 libquadmath0 libtsan0 libubsan1 linux-libc-dev
  make manpages-dev
Se actualizarán los siguientes paquetes:
  libc6
1 actualizados, 28 nuevos se instalarán, 0 para eliminar y 15 no actualizados.
Se necesita descargar 31,3 MB de archivos.
Se utilizarán 123 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```



- Reinstalamos de novo as additions

```
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
Configurando libcrypt-dev:amd64 (1:4.4.10-10ubuntu4) ...
Configurando libisl22:amd64 (0.22.1-1) ...
Configurando libbinutils:amd64 (2.34-6ubuntu1) ...
Configurando libc-dev-bin (2.31-0ubuntu9.2) ...
Configurando libcc1-0:amd64 (10.2.0-5ubuntu1~20.04) ...
Configurando liblsan0:amd64 (10.2.0-5ubuntu1~20.04) ...
Configurando libitm1:amd64 (10.2.0-5ubuntu1~20.04) ...
Configurando gcc-9-base:amd64 (9.3.0-17ubuntu1~20.04) ...
Configurando libtsan0:amd64 (10.2.0-5ubuntu1~20.04) ...
Configurando libctf0:amd64 (2.34-6ubuntu1) ...
Configurando libasan5:amd64 (9.3.0-17ubuntu1~20.04) ...
Configurando cpp-9 (9.3.0-17ubuntu1~20.04) ...
Configurando libc6-dev:amd64 (2.31-0ubuntu9.2) ...
Configurando binutils-x86-64-linux-gnu (2.34-6ubuntu1) ...
Configurando binutils (2.34-6ubuntu1) ...
Configurando libgcc-9-dev:amd64 (9.3.0-17ubuntu1~20.04) ...
Configurando cpp (4:9.3.0-1ubuntu2) ...
Configurando gcc-9 (9.3.0-17ubuntu1~20.04) ...
Configurando gcc (4:9.3.0-1ubuntu2) ...
Procesando disparadores para man-db (2.9.1-1) ...
Procesando disparadores para libc-bin (2.31-0ubuntu9.1) ...
root@ubserver2004:/mnt# ./VBoxLinuxAdditions.run
Verifying archive integrity... All good.
Uncompressing VirtualBox 6.1.16 Guest Additions for Linux........
VirtualBox Guest Additions installer
Removing installed version 6.1.16 of VirtualBox Guest Additions...
Copying additional installer modules ...
Installing additional modules ...
```

```
VirtualBox Guest Additions: Starting.
VirtualBox Guest Additions: Building the VirtualBox Guest Additions kernel
modules.  This may take a while.
VirtualBox Guest Additions: To build modules for other installed kernels, run
VirtualBox Guest Additions:   /sbin/rcvboxadd quicksetup <version>
VirtualBox Guest Additions: or
VirtualBox Guest Additions:   /sbin/rcvboxadd quicksetup all
VirtualBox Guest Additions: Building the modules for kernel 5.4.0-64-generic.
_
```

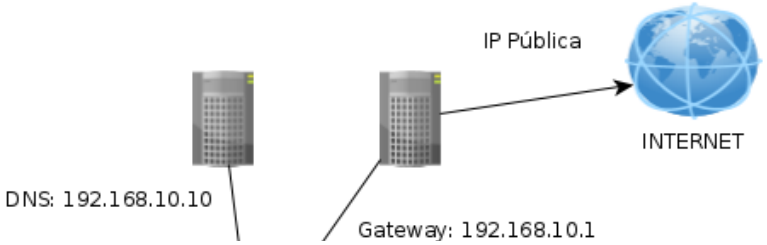
- Reiniciamos a máquina e comprobamos que temos a última versión instalada:

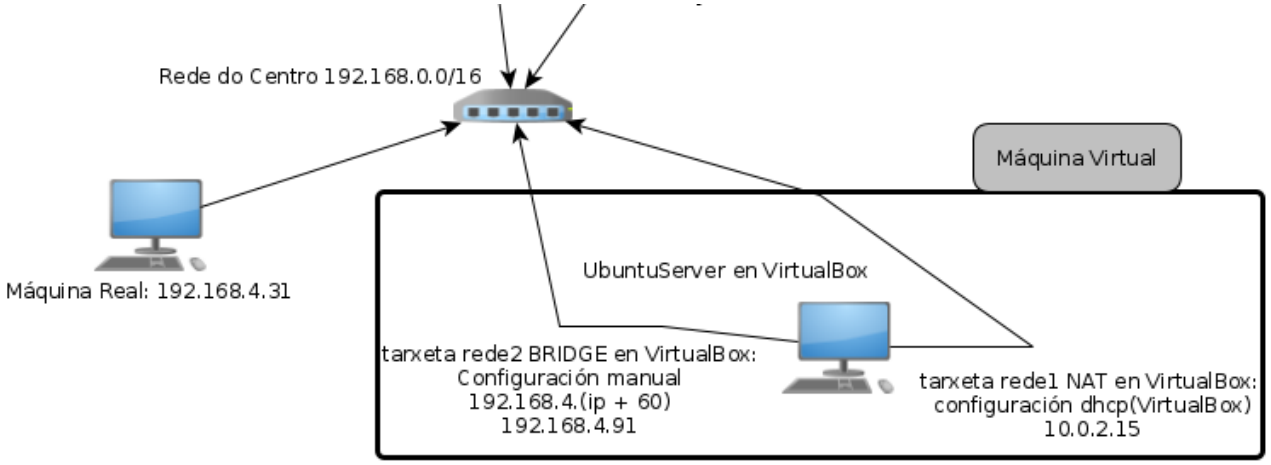
```
root@ubserver2004:/home/antonio# lsmod | grep vbox
vboxvideo              36864  0
vboxguest             348160  1
ttm                   106496  2 vmwgfx,vboxvideo
drm_kms_helper        184320  2 vmwgfx,vboxvideo
drm                   491520  5 vmwgfx,drm_kms_helper,vboxvideo,ttm
root@ubserver2004:/home/antonio# modinfo vboxguest
filename:       /lib/modules/5.4.0-65-generic/misc/vboxguest.ko
version:        6.1.16 r140961
license:        GPL
description:    Oracle VM VirtualBox Guest Additions for Linux Module
author:         Oracle Corporation
srcversion:     72CDCB8BA3ED3B5FE9AE9E8
alias:          pci:v000080EEd0000CAFEsv00000000sd00000000bc*sc*i*
depends:
retpoline:      Y
name:           vboxguest
vermagic:       5.4.0-65-generic SMP mod_unload
root@ubserver2004:/home/antonio#
```

Configuración de Postgresql para poder acceder con pgadmin4 dende outra máquina

Este punto só é no caso de que queiramos conectarnos a Postgresql con pgadmin4 dende outra máquina, (en conexión remota, non como localhost).

Por exemplo instalamos unha máquina virtual en VirtualBox (UbuntuServer) que non terá entorno gráfico.



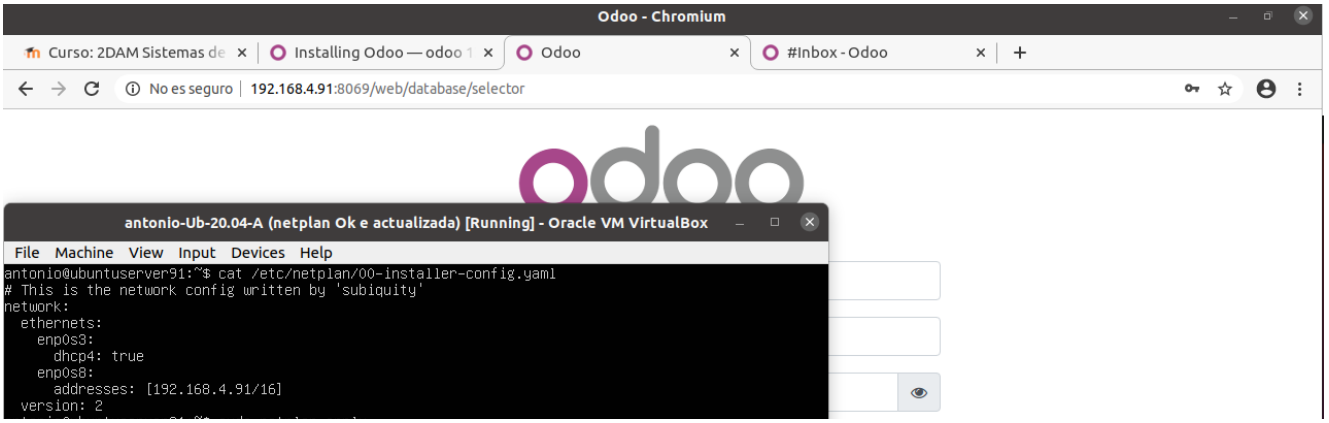


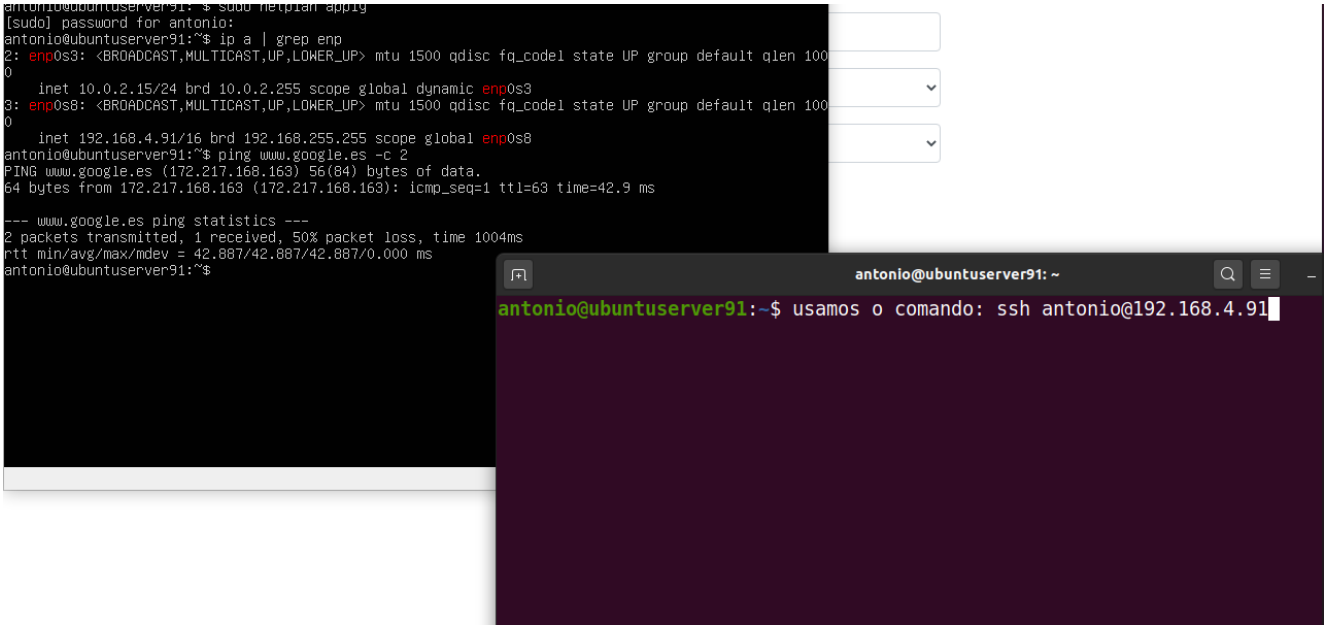
Imos poñerlle a esa máquina virtual 2 tarxetas de rede. Configuradas en VirtualBox unha en NAT (para poder sair a Internet coa ip da nosa máquina real) e a outra en Bridge (para acceder a máquina virtual dende a máquina real)

- NAT: A tarxeta estará configurada por DHCP terá a IP que lle facilita VirtualBox 10.0.2.15 e mediante NAT se traducirán os paquetes coa IP da nosa máquina real por iso podremos navegar por Internet.
- Bridge: A tarxeta estará configurada de xeito manual. Terá como IP os 3 primeiros octetos igual que a IP da nosa máquina real e no último octeto a IP da nosa máquina sumandolle por exemplo 60. No meu caso se a IP da máquina é 192.168.4.31 configuremos coa IP 192.168.4.91/16

Para realizar a configuración seguiremos os seguintes puntos:

- Configuración de 2 tarxetas de rede en Ubuntu Server 20.04. Para configurar as tarxetas en Ubuntu temos que utilizar netplan https://netplan.io/examples
- Se desexamos poder acceder de xeito remoto mediante o usuario root (o cal non é aconsellable por seguridade) teremos que poñer unha password para root e configurar o servidor ssh do Ubuntu Server https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/v2v_guide/preparation_before_the_p2v_migration-enable_root_login_over_ssh





Vemos dende a consola da máquina virtual como temos que ter configurada a rede mediante netplan.

Co comando **ip a** vemos as tarxetas configuradas correctamente:

- Unha tarxeta mediante DHCP
- E a outra tarxeta coa asignación manual de IP. (OLLO: na tarxeta que configuramos con IP manual só necesitamos configurar a IP e a máscara, xa que a información do gateway e o DNS será a que obtemos mediante a tarxeta configurada por DHCP)

Vemos tamén mediante o comando **ping** que temos acceso a Internet .

E dende a máquina real podemos conectarnos mediante SSH.

E tamén vemos como accedemos (dende o navegador web) ao Odoo instalado na máquina virtual mediante a IP desa máquina **192.168.4.91** e o porto **8069**

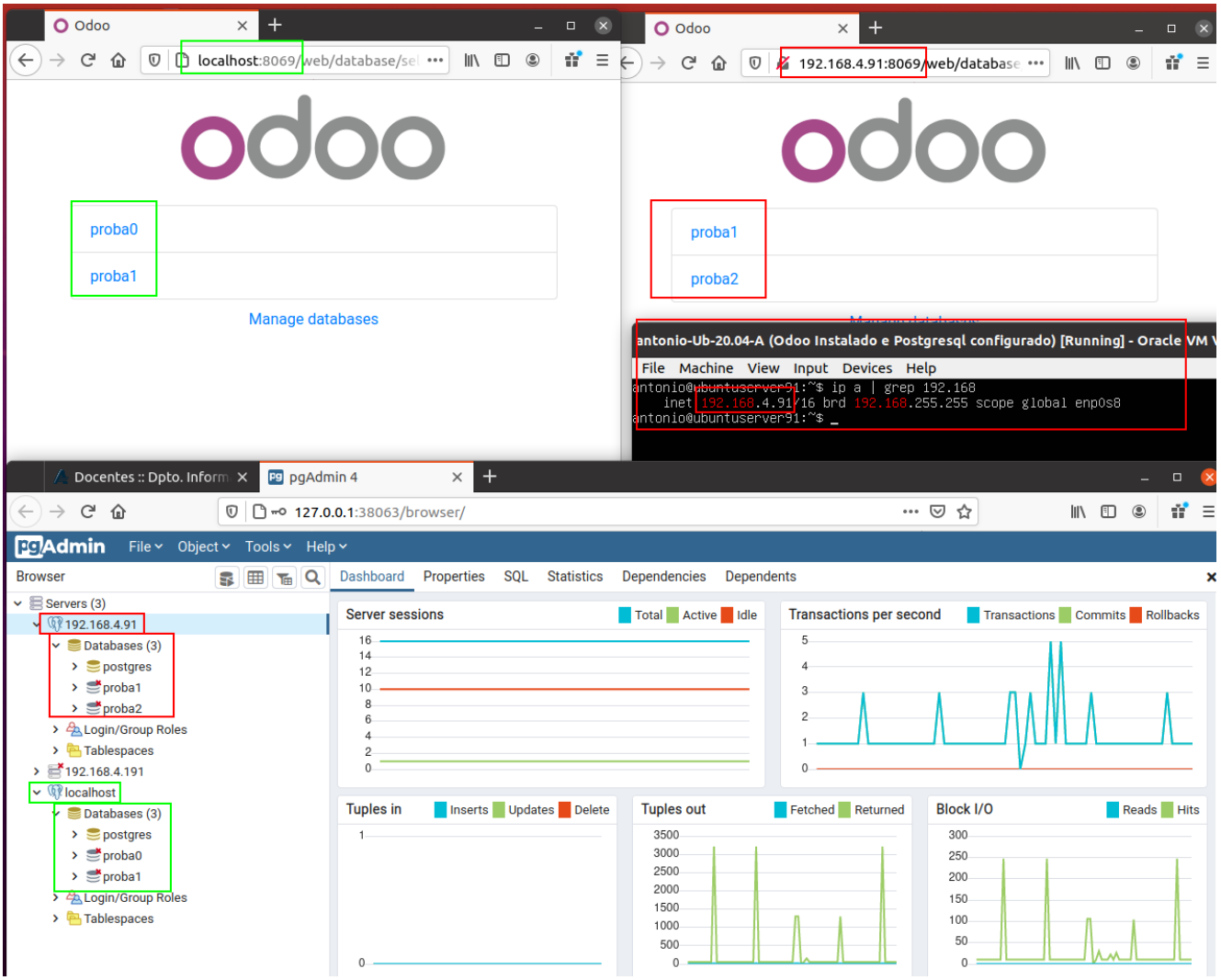
Por seguridade a **conexión remota a Base de Datos Postgresql** está deshabilitada por tanto temos que cambiar a configuración de Postgresql para poder conectarnos dende **pgadmin4**:

- Para que postgresql escoite por todas as ip da máquina. Modificamos o arquivo **/etc/postgresql/12/main/postgresql.conf** (tendo en conta que 12 é a versión) descomentamos e modificamos a liña:
  - #listen_addresses = 'localhost' por listen_addresses = '*'
  - Despois reiniciamos o servizo: **systemctl restart postgresql**
- Para que postgresql confie nas solicitudes feitas dende unha máquina ou subrede en concreto modificamos o arquivo **/etc/postgresql/12/main/pg_hba.conf** (tendo en conta que 12 é a versión)

- host all all **ip/mascara** trust
  - Para unha máquina en concreto onde ip será a ip da máquina en concreto e 32 a máscara. Por exemplo 192.168.10.15/32
  - Para unha subrede. Por exemplo 192.168.0.0/16
- Despois reiniciamos o servizo: **systemctl restart postgresql**

Vemos as BBDD mediante odoo tanto en localhost(verde) como na máquina virtual(vermello).

E mediante o **pgadmin4** que temos instalado na máquina real xa podemos acceder tanto ao postgresql que está instalado en localhost(verde) como ao postgresql que está instalado na máquina virtual con ip 192.168.4.91 (vermello).



Pasar de http a https

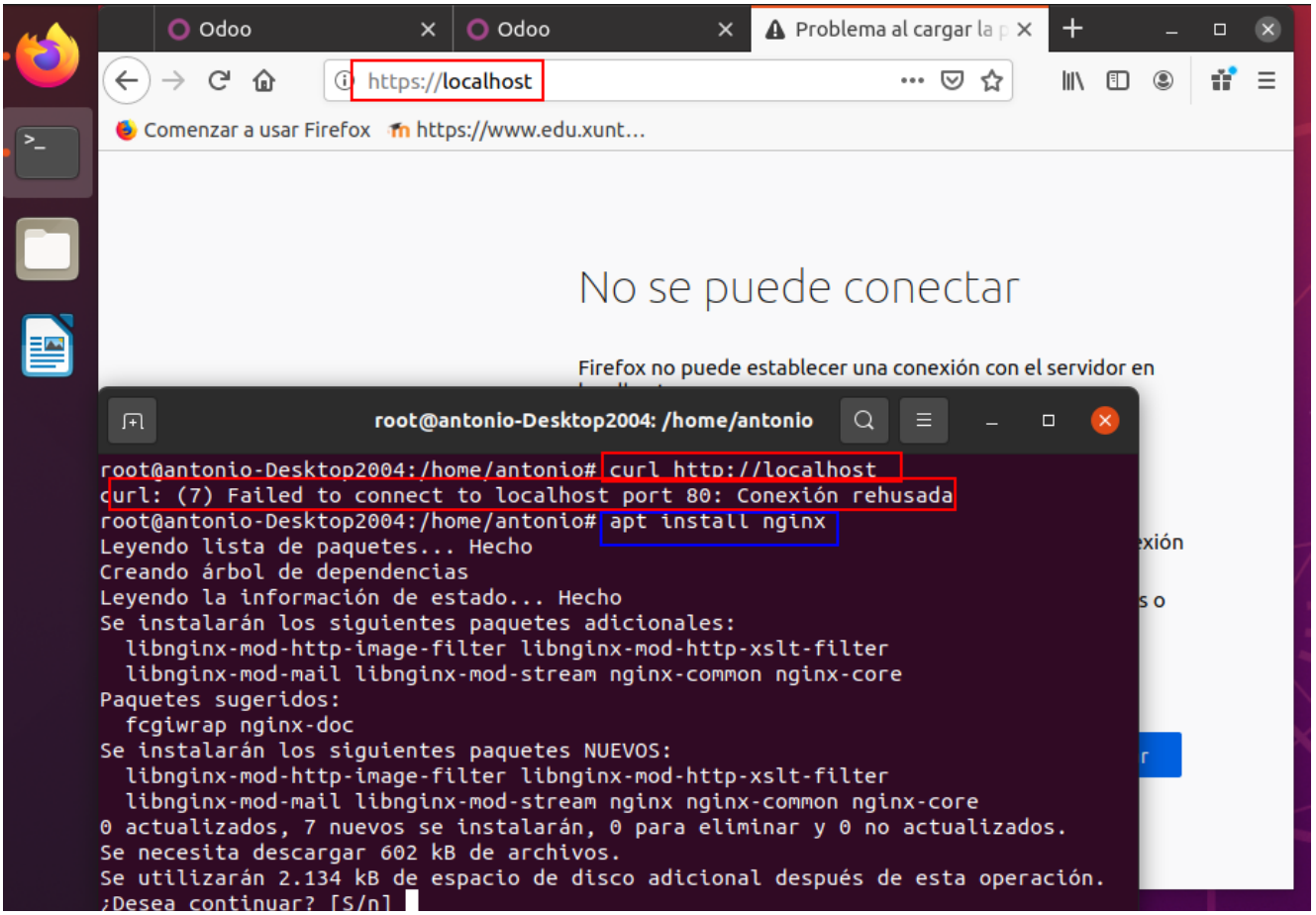
Se queremos entre outras melloras, que por exemplo a comunicación entre o cliente web e o noso servidor web de Odoo esté **cifrada** (o que implica que ninguén que intercepte a

comunicación poda ler información comprometida por exemplo as passwords...) temos que implentar **https** (para elo instalaremos un certificado SSL no noso servidor)

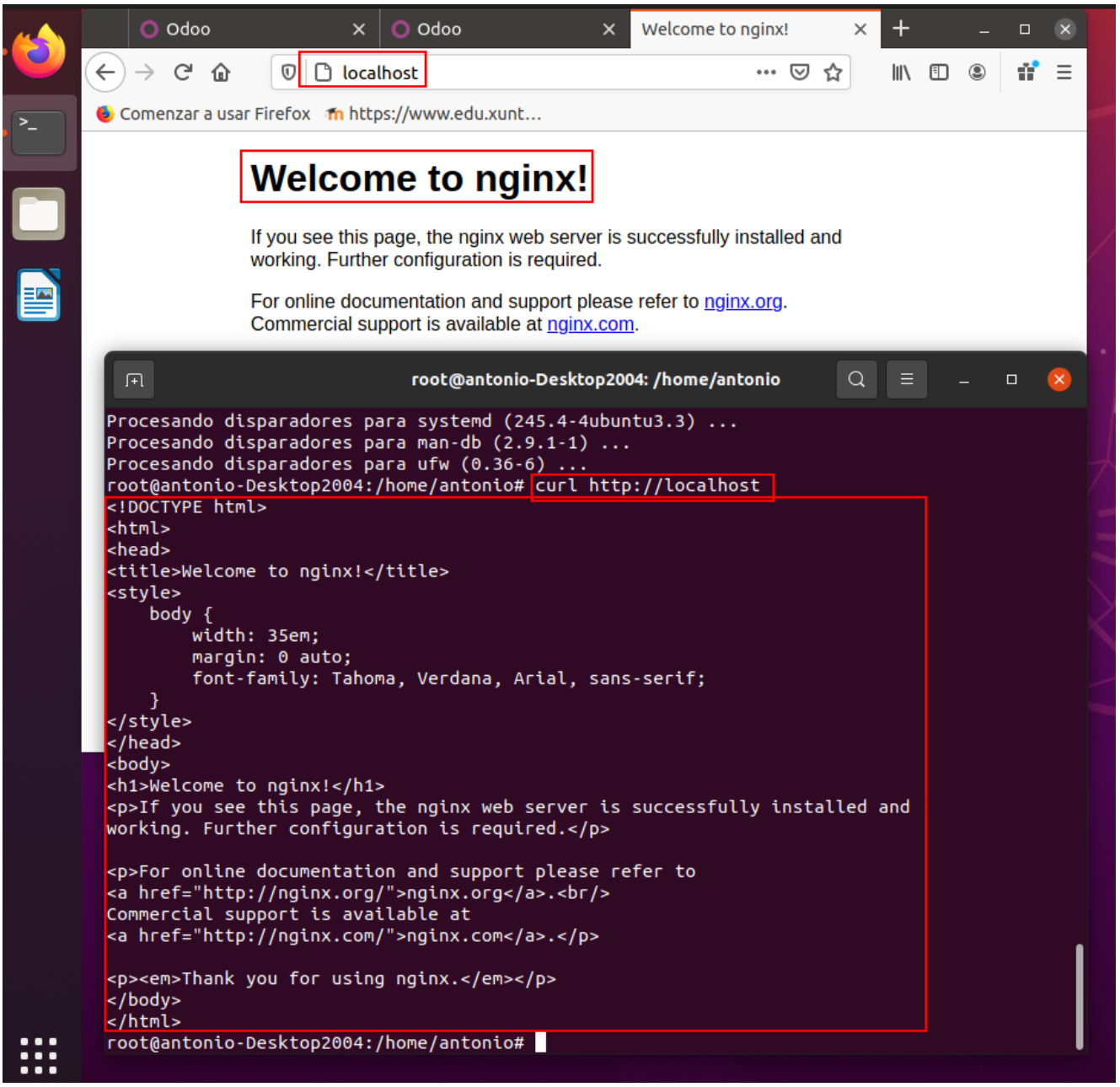
1. O primeiro paso é utilizar un servidor web por exemplo **nginx** para actuar como **reverse proxy** (con elo nginx escoitará no porto 80 do noso servidor e redireccionará o tráfico ao porto 8069 onde temos o noso servizo Odoo funcionando) Esto implica melloras tanto na velocidade (xa que nginx fará caché, posibilidade de balanceo de carga) como na seguridade.
2. Instalar en nginx un certificado para poder traballar con **https** e configurar nginx para que traballe con ese certificado e SSL(no porto 443)
   O noso certificado será gratuito xa que non estará avalado por unha CA (Autoridade de Certificación) o que implicará (como veremos despois) unha mensaxe de advertencia aos clientes que se conecten ao noso sitio web de que non é seguro.
3. Finalmente xa poderemos navegar mediante https ou http

1. Instalar nginx e que reenvie do porto 80 ao 8069

Comprobamos (vermello) que non temos ningún servidor web funcionando no porto 80 e a continuación instalamos nginx (azul) :



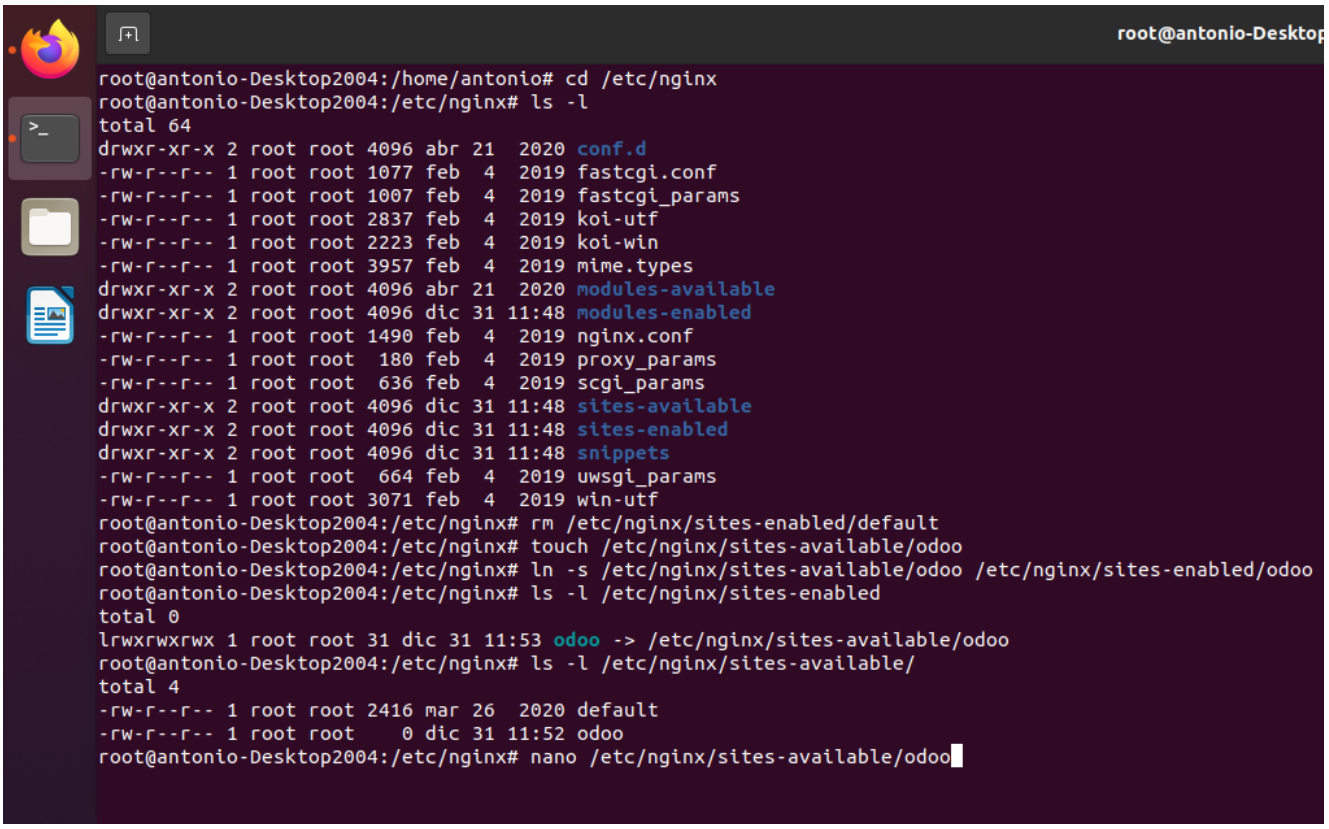
Despois de instalar nginx comprobamos que temos funcionando no porto 80 o servidor nginx coa configuración por defecto:



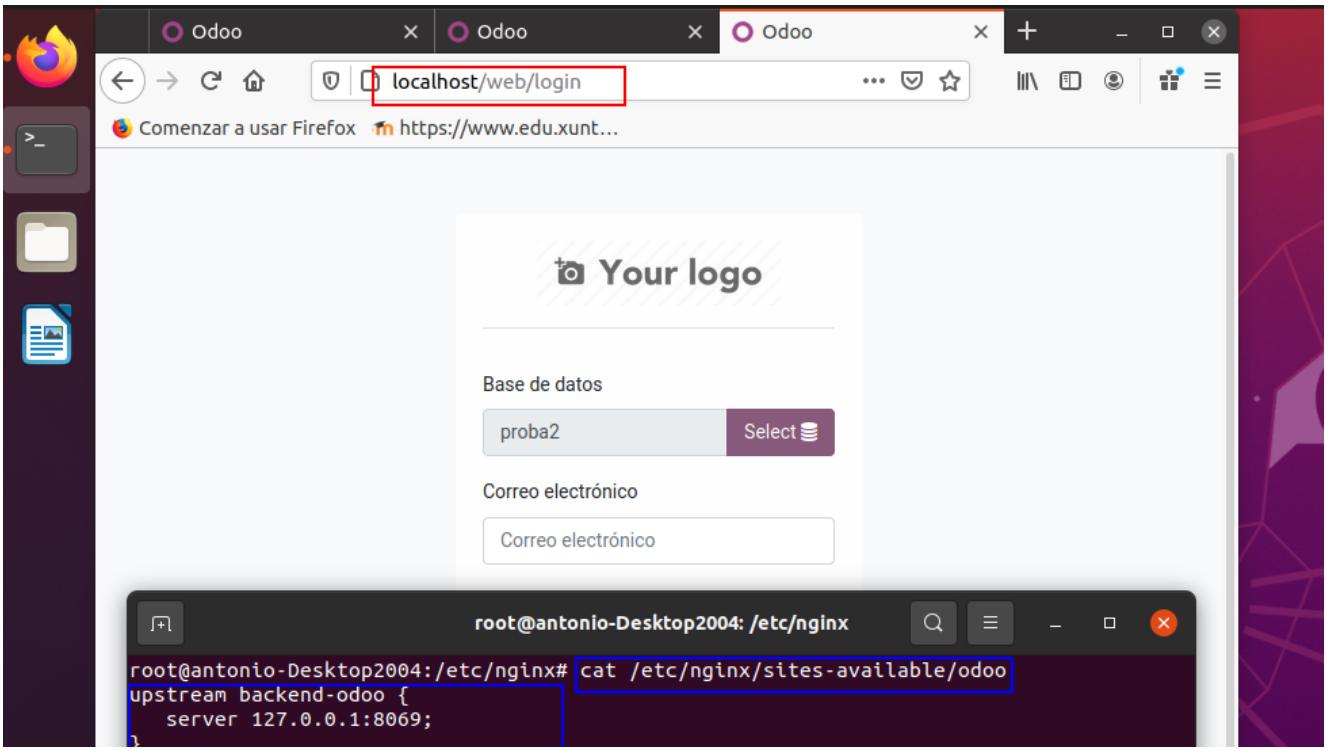
Despois preparamos a configuración do servidor web nginx para que reenvie do porto 80 ao porto 8069:

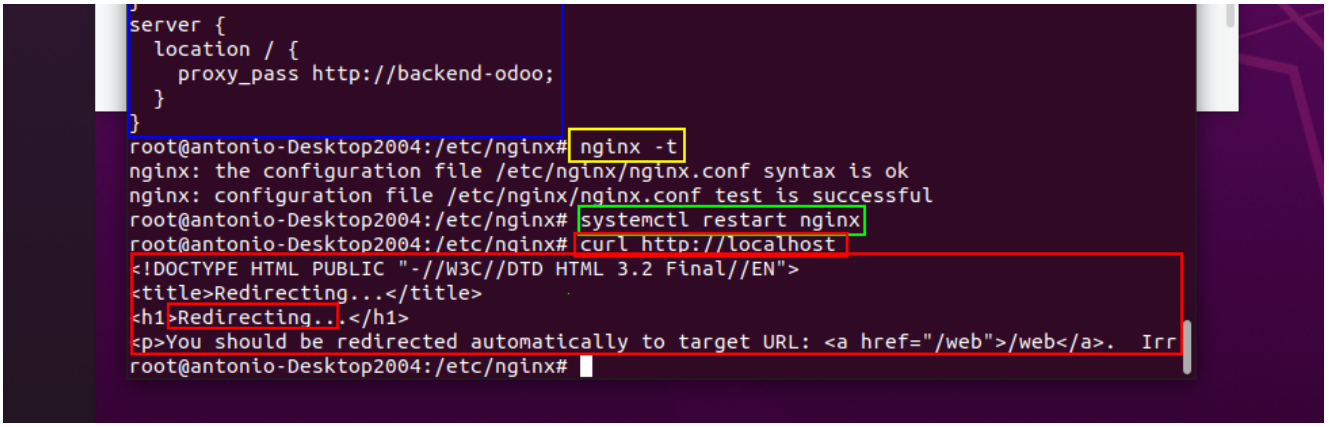
1. eliminamos o ficheiro coa configuración por defecto de sites-enabled
2. creamos o ficheiro para a configuración de odoo en sites-available
3. creamos un enlace simbólico (do arquivo odoo creado) entre o directorio sites-available e sites-enabled

4. editamos mediante nano o ficheiro odoo para establecer a configuración desexada



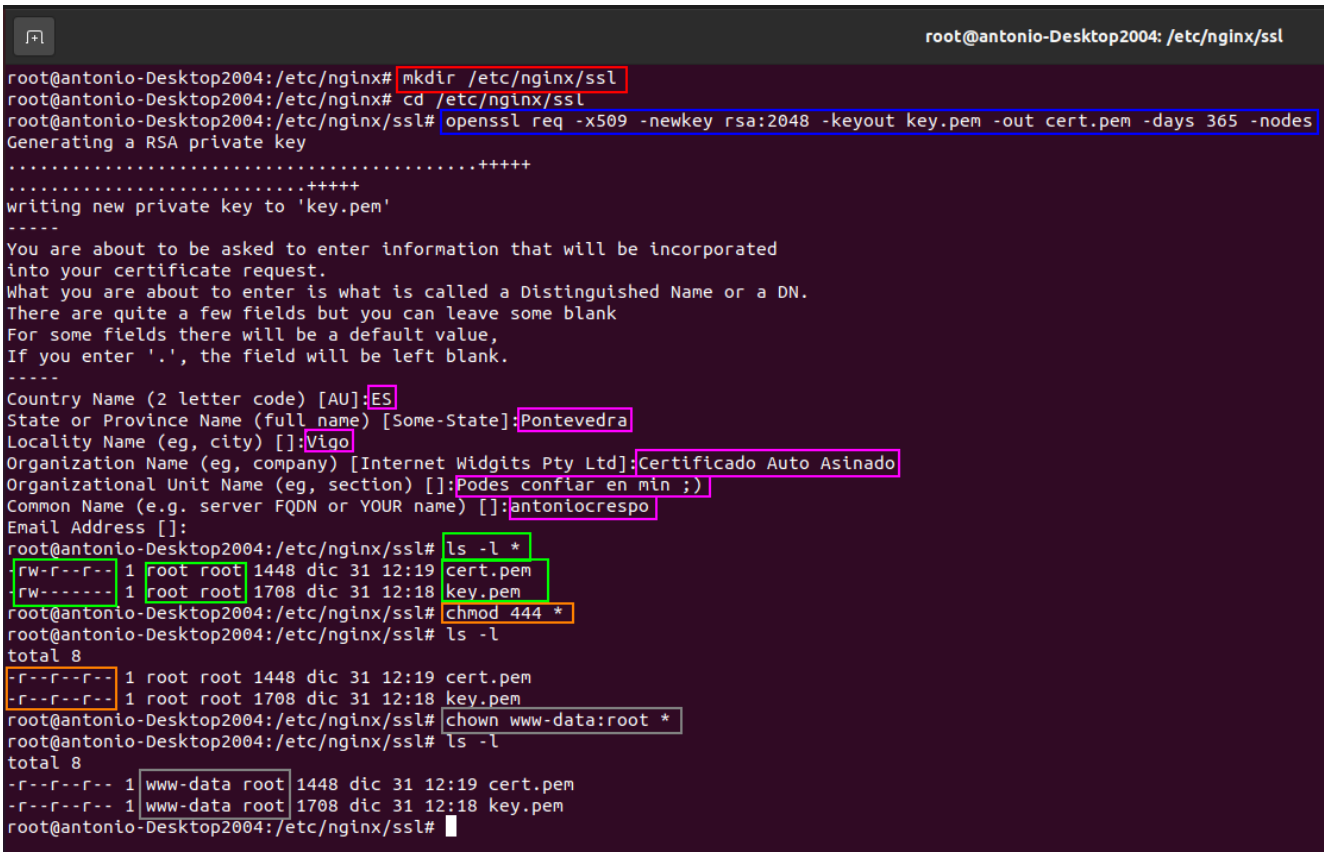
Despois de editalo con nano o ficheiro **/etc/nginx/sites-available/odoo** quedaría có seguinte aspecto(azul), a continuación comprobamos mediante *nginx -t* que a sintaxe está ben (amarelo) e reiniciamos o servizo de nginx para que surtan efecto os cambios (verde). Finalmente comprobamos que xa redirecciona entre o porto 80 e o 8069 (vermello). Xa que no navegador ao poñer localhost xa redireciona a localhost:8069 que é onde está escoitando odoo





2. Instalar certificado en nginx

Agora para traballar con SSL crearemos e instalaremos en nginx un **certificado auto-asinado**



Como vemos na anterior imaxe:

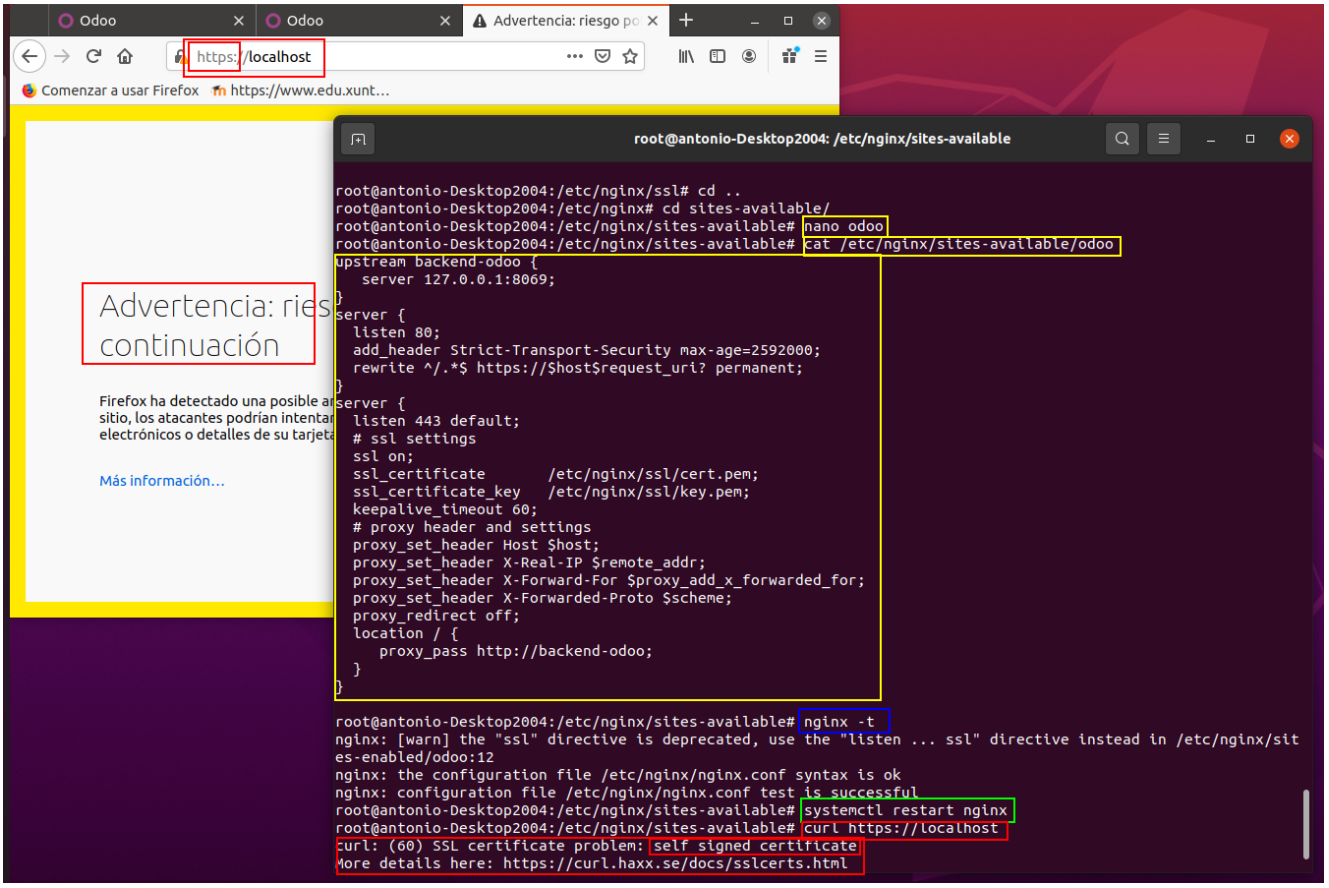
- creamos o directorio (vermello)
- cambiamos ao directorio /etc/nginx/ssl co comando cd
- creamos os arquivos mediante o comando **openssl** (azul)
- no proceso de creación dos ficheiros configuramos certos valores, que

posteriormente serán visibles no noso certificado (violeta)
- vemos os ficheiros creados cos seus permisos e propietario (verde)
- cambiamos os permisos a só lectura (marrón)
- cambiamos o propietario (gris)

Agora configuraremos (amarelo) mediante o comando nano o arquivo **/etc/nginx/sites-available/odoo** para traballar co noso certificado mediante SSL no porto 443

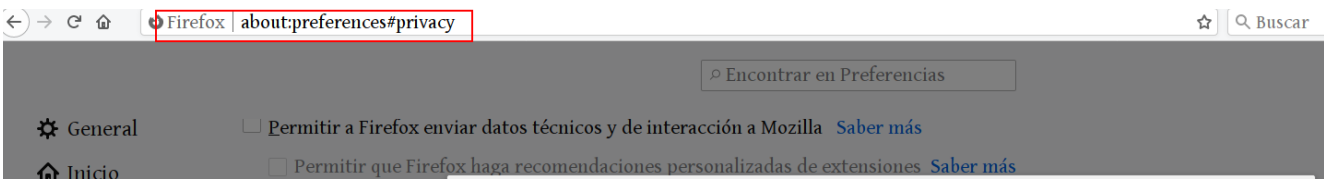
Comprobamos mediante nginx -t (azul) (vemos que nos da un warning) e reiniciamos o servizo nginx (verde)

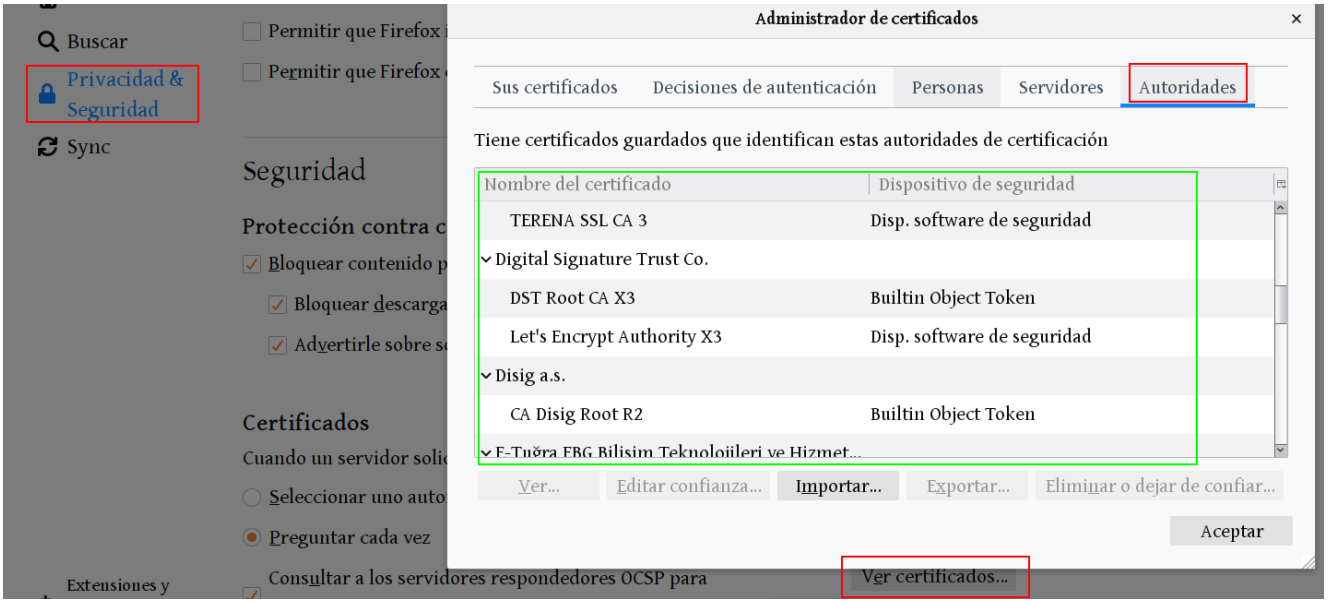
E xa podemos comprobar como ao poñer localhost no navegador xa accedemos mediante https (vermello) ou mediante curl



Iso si, como o noso **certificado** é **auto asinado**. Quere decir que non está avalado por ninguna Autoridade Certificadora[CA]

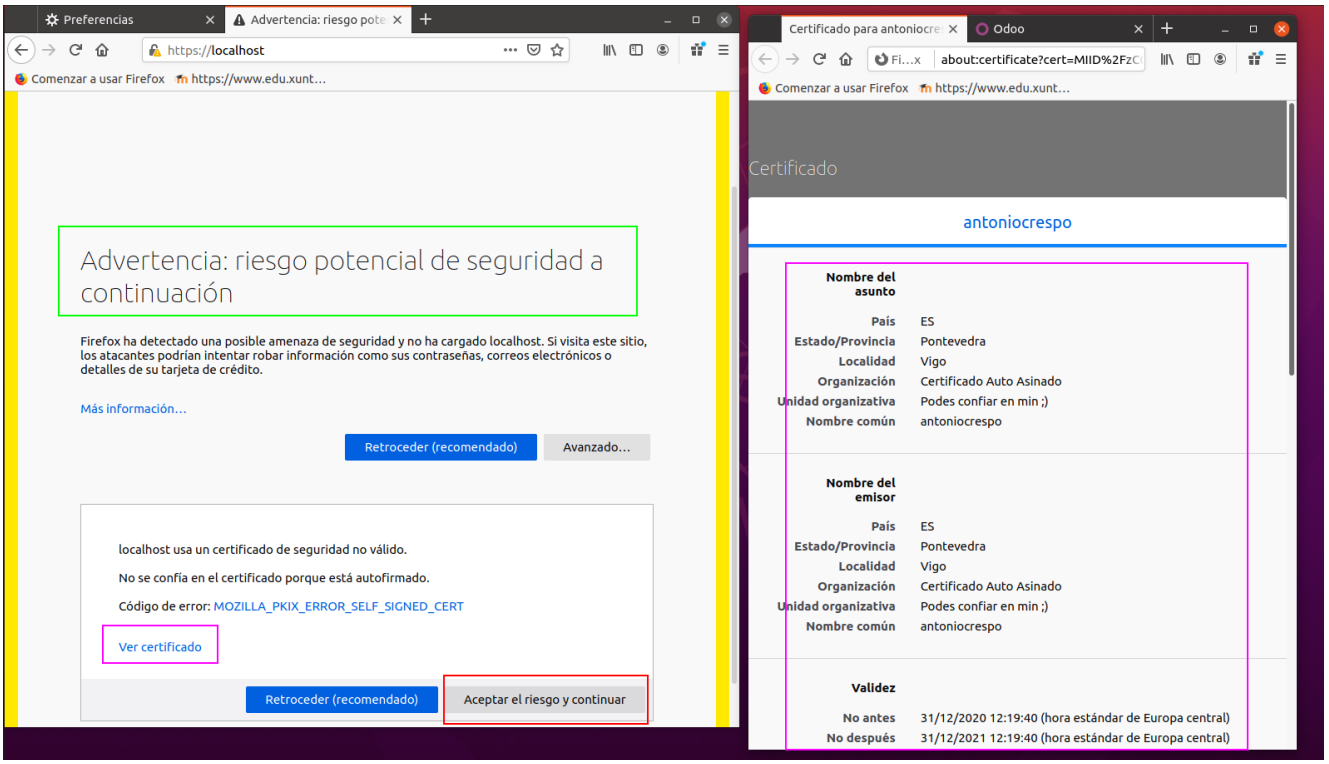
Para ver as CA que temos instaladas por defecto (se poden importar máis manualmente) imos a preferencias (vermello) do noso navegador e consultamolas (verde) :





Como decía anteriormente ao ser o noso un **certificado auto asinado**, os navegadores non confiarán no noso sitio web (verde). Podemos ver a información do noso certificado (violeta)

Finalmente para poder acceder ao noso sitio web teremos que engadir unha excepción no navegador para que confie no noso sitio web (vermello).

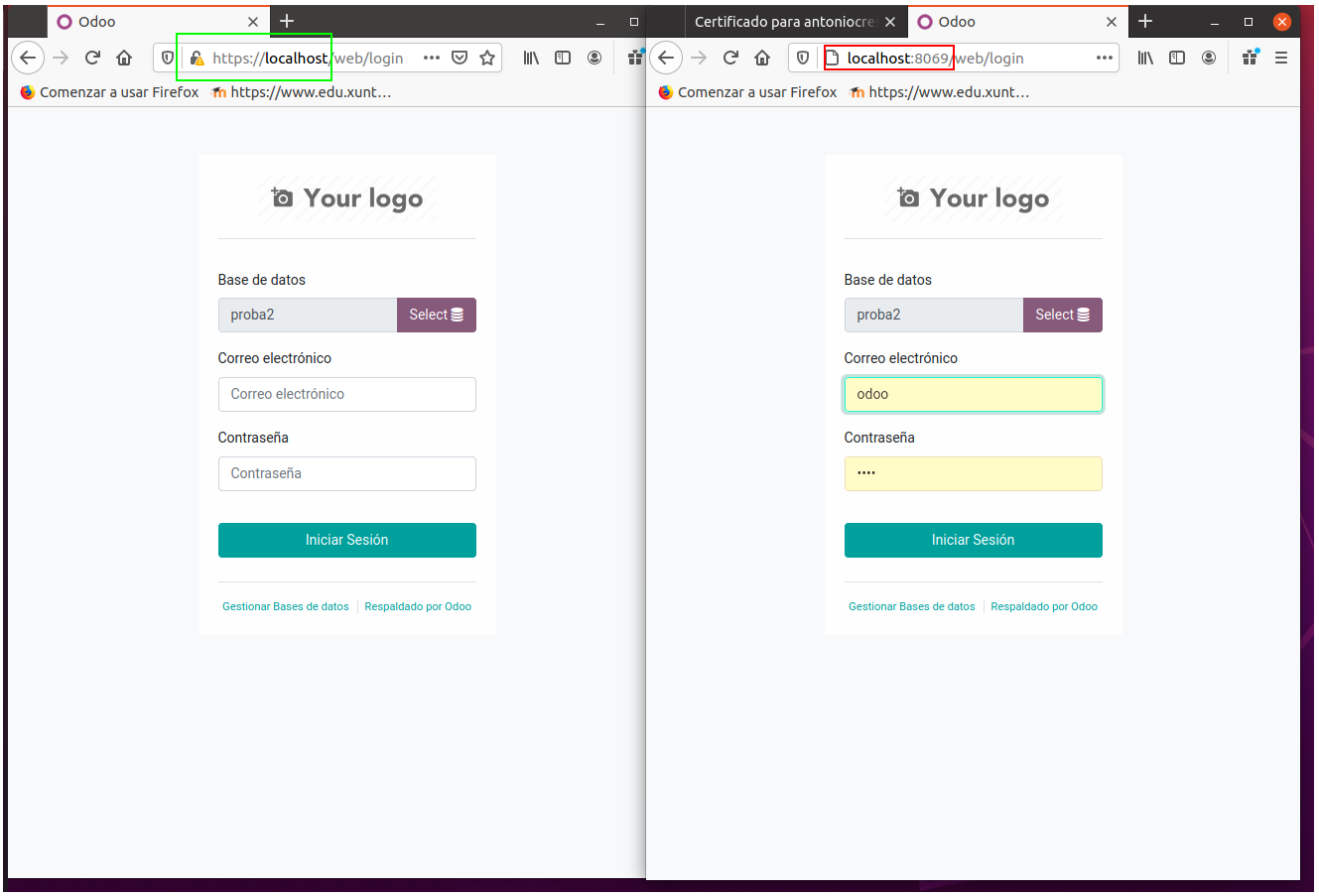


Mediante **http** o tráfico entre os clientes e o servidor web **non vai cifrado**. Por exemplo as passwods van en claro(con http calquera podería esnifar a rede e ver o contido dos paquetes). Unha vantaxe de usar **https** é que o tráfico vai **cifrado**.

3. Navegar por https ou http

Como vemos agora poderíamos :

- navegar con **https** (verde) no porto 80 (porto por defecto) xa que nginx redirecciona ao porto 443 (SSL) e finalmente ao 8069(odoo). **Tráfico cifrado**.
- ou navegar con **http** (vermello) directamente no porto 8069 (odoo). **Tráfico sen cifrar**.



Solución a confianza mediante Let's Encrypt

Se imos ter o noso Odoo accesible ao público en Internet é interesante ter un certificado asinado por unha Autoridade Certificadora CA. Así os clientes non terán que confiar no noso certificado auto asinado.

Podemos ter dun xeito gratuito un certificado asinado mediante unha Autoridade

Certificadora (CA), grazas a Let's Encrypt

Por exemplo se temos o noso Odoo en remoto instalado nun server con Ubuntu 20.04 con Nginx (como servidor web) e podemos acceder con ssh.

Teríamos, por exemplo, que seguir os seguintes pasos