How to install Windows 7 in an LXD Virtual Machine and Configure Windows 7 for Windows RemoteApp's to be run from the Ubuntu Host.

By Brian Mullan (c) MIT (9/10/2020)
bmullan.mail@gmail.com

Several months ago LXD gained the capability to create Qemu Virtual Machines (VMs) using the same CLI syntax as you are familiar with when using LXD Containers.

> *LXD 3.19 will ship with virtual machine support.*
> *This effectively lets you mix and match containers and virtual machines on the same system based on the workloads you want to run. Those virtual machines use the same profiles, networks and storage pools as the containers.*
>
> *The VMs are run through qemu using separate VM images. To attempt to get feature parity with LXD containers, an agent is available which when run in the VM makes it possible to use lxc exec, lxc file, … the same way you would with a container.*

You can create Linux and Windows VMs although creating and using the Windows VMs takes a bit more manual steps.

This document will try to record all the steps required to:

1. Create a Windows 7 Ultimate LXD VM
2. Enable Networking for Windows 7
3. Configure the Default Route for Windows 7 to reach the Internet
4. Download and Install Kim Knight's fantastic RemoteAppTool onto the Windows 7 VM.
5. Configure the RemoteAppTool so you, from you Ubuntu Host, start ANY Windows Application you have installed on that Windows 7 VM and have that Windows Application appear and run inside a normal Gnome "window" just as any other Linux application does on your Desktop.

Your Window's RemoteAPP window is resizeable, you can minimize/maximize it just like any normal Linux application you use !

NOTE: For my Guide I am using Windows 7 Ultimate

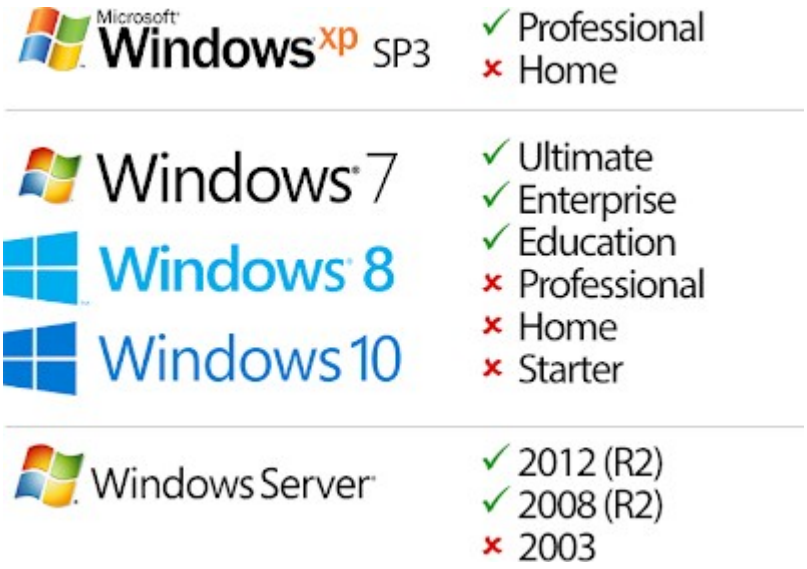
But ***Kim Knight*** *the Author* of the **RemoteAppTool:**

http://www.kimknight.net/remoteapptool

lists which versions of Windows will work:

Requirements

- Microsoft .Net Framework 4
- WiX Toolset (If you want to create MSIs. Reboot after installing.)
- A supported version/edition of Windows:



So you will need a .ISO file of one of the above and if its not Windows 7… some of my Guide may be different that what you will need to do (I don't know).

Lastly, you "should" be using a Windows Version you actually own/owned and have the Activation/Registration code for it available as the installation asks for it to be entered.

However, if you don't have it .. I think Microsoft still lets you use your version of Windows but will put up a NAG message every so often to enter the code.

Lastly, if all you have is your Windows CD and don't know how to create a .ISO file from that CD go watch one of these YouTube Video's. One of them will make sense to you about how to create the .ISO from a CD.

https://www.youtube.com/results?search_query=make+bootable+iso+from+windows+cd

So before proceeding make your windows .ISO file!

Lets Get Started

Step 1

Create a temporary sub-directory to your own “home” directory.

I tried to keep names short/simple so I named mine “win” so typing later is easier to do:

> *$ mkdir ~/win*

*Copy your Windows 7 Ultimate .ISO to that directory (~/win) and just name it “**win.iso**”*

Step 2

Using Stephane Graber’s post about how to create Linux and Windows VMs with LXD:

https://discuss.linuxcontainers.org/t/running-virtual-machines-with-lxd-4-0/7519

The Section describing the Windows VM creation is what is important here from the above!

> ***IMPORTANT NOTE:***
> *Stephane’s info is about creating a Windows 10 LXD VM but I wanted a Windows 7 VM so you will see the naming difference in the rest of this guide regarding Names.*

Get the latest Virtio Drivers for Windows and put it into the ~/win directory

> ***$ cd ~/win***
>
> *The following is 1 line...*
>
> ***$ wget https://fedorapeople.org/groups/virt/virtio-win/direct-downloads/archive-virtio/virtio-win-0.1.173-9/virtio-win.iso***

Rename it** for simplicity to: **virtiowin.iso

Verify that in the directory ~/win you now have 2 .ISO files named:

> *$ ~/win$ ls*
> ***virtiowin.iso win7.iso***

To simplify Stephane’s instructions and tailor them for myself I created a Bash Script which made it easy to edit and easy execute/re-execute.

Being able to re-execute is important because if you are like me you may screw something up a few times and need to start over more than once :-) .

I just called my script: mk-win7vm.sh

```
#!/bin/bash

# Create an empty VM with beefier CPU/RAM and SecureBoot disabled.
#
# note that I called my LXD VM “win7” w 4 cpu and 8GB of RAM.
# you can specify the CPU and RAM you want use but for this scripts purposes leave
# the VM name as “win7” ! (this command is all 1 line)

lxc init win7 --profile default --empty --vm -c security.secureboot=false -c limits.cpu=4 -c limits.memory=8GB

# Grow its root disk to a reasonable size (I made mine 40GB)

lxc config device override win7 root size=40GB

# Enable “temporary” install and drivers media (all 1 line)

echo -n '-drive file=/home/bmullan/win/win7.iso,index=0,media=cdrom,if=ide -drive file=/home/bmullan/win/virtiowin.iso,index=1,media=cdrom,if=ide' | lxc config set win7 raw.qemu -

# modify the apparmor setting so VM is permitted Read/Write
# and file locking on anythong under /home

lxc config set win7 raw.apparmor "/home/** rwk,"
```

Save the BASH file and make it executable:

```
$ chmod +x mk-win7vm.sh
```

Execute that script:

```
$ ./mk-win7vm.sh
```

Step 3

Now comes the *trickiest part* as mentioned by Stephane.

> Note for the Start Command below:
> I did this quite a few times (re-do's) before I got the timing right. If you screw up you will need to Stop the LXD container named Win7 and retry again:
>
> **$ lxc stop win7 --force**

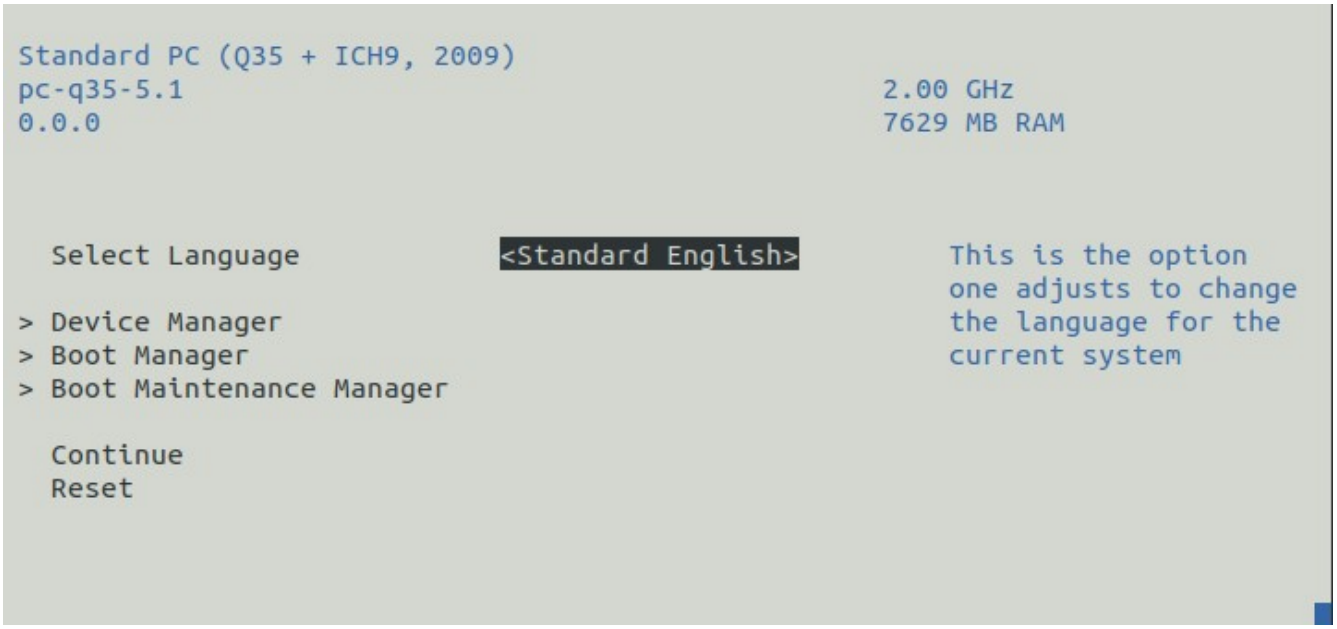
Start the VM:

```
$ lxc start win7 --console
```

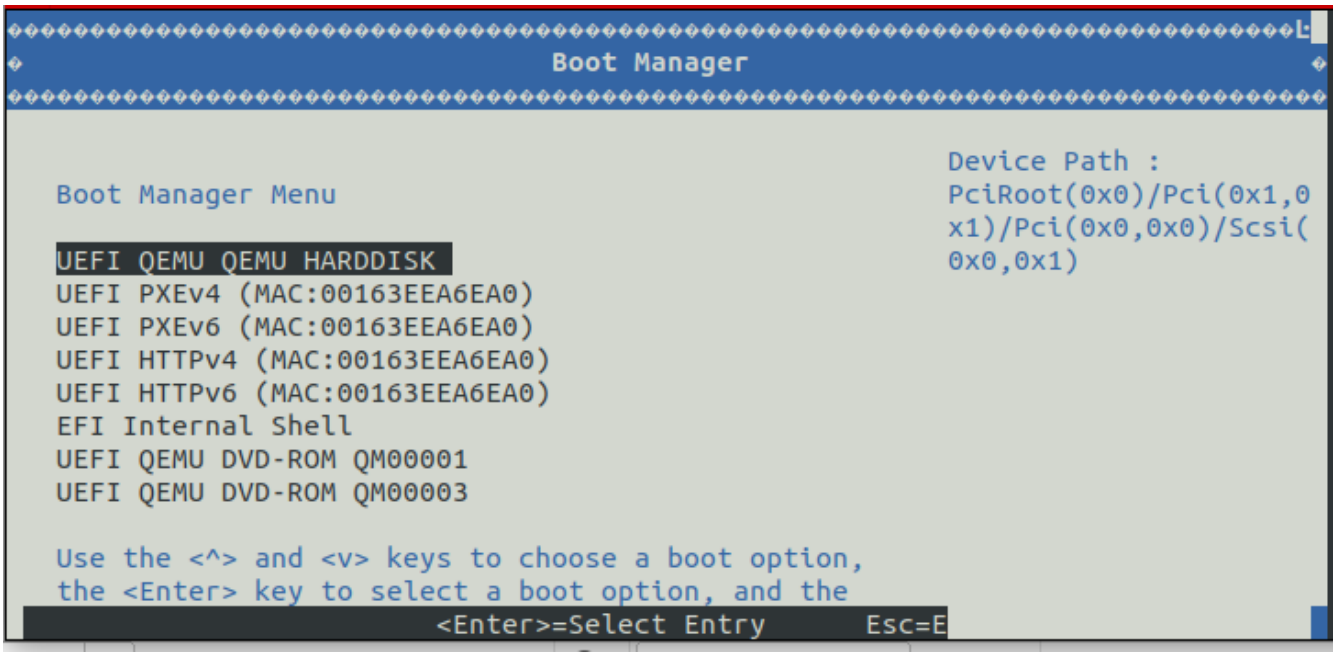
IMPORTANT: IMMEDIATELY & REPEATEDLY

> ***Hit ESC*** *– EVEN BEFORE ANYTHING APPEARS ON YOUR SCREEN* in order to enter the Windows Firmware Menu.

You will/should see the Windows Firmware Menu that looks like this:



From that Firmware Menu “arrow” down and select “`Boot Manager`” and you will see a screen like the followings:



Arrow down and highlight - **UEFI QEMU DVD-ROM QM00001** then press Enter key

When you do that your “Console” will clear:

but its really invisible and expecting you to Hit Enter for an “Invisible Boot Prompt”

So now ***Hit ENTER 2-3 times*** *to answer* ***the invisible boot prompt.***

THEN… disconnect from that "console" using ***`ctrl+a`*** `then` ***`q`*** `keys`

At this point you will be back at your Ubuntu Terminal Prompt.

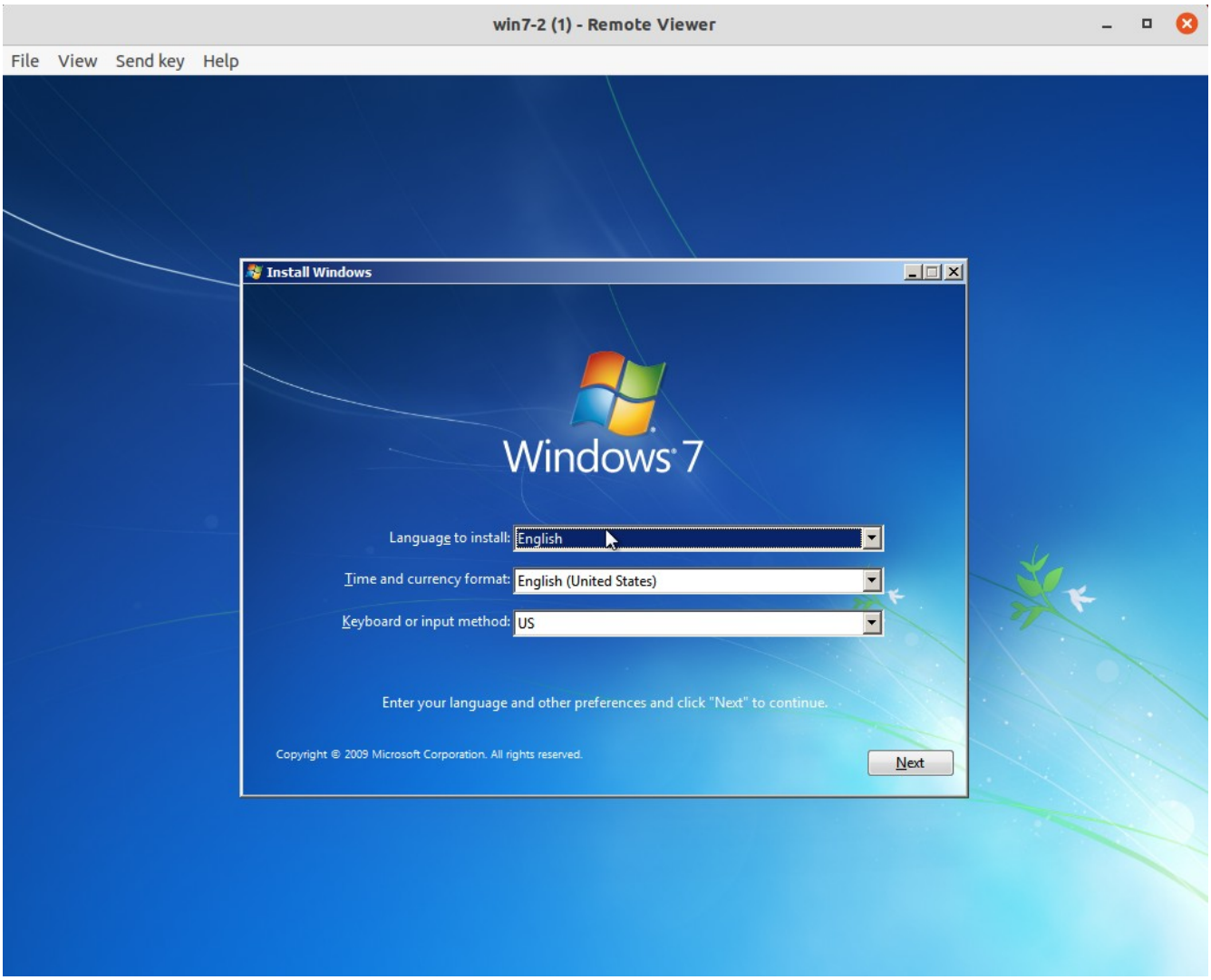
Step 4

Next you will use the the LXD VGA console with the command:

```
$ lxc console win7 --type=vga
```

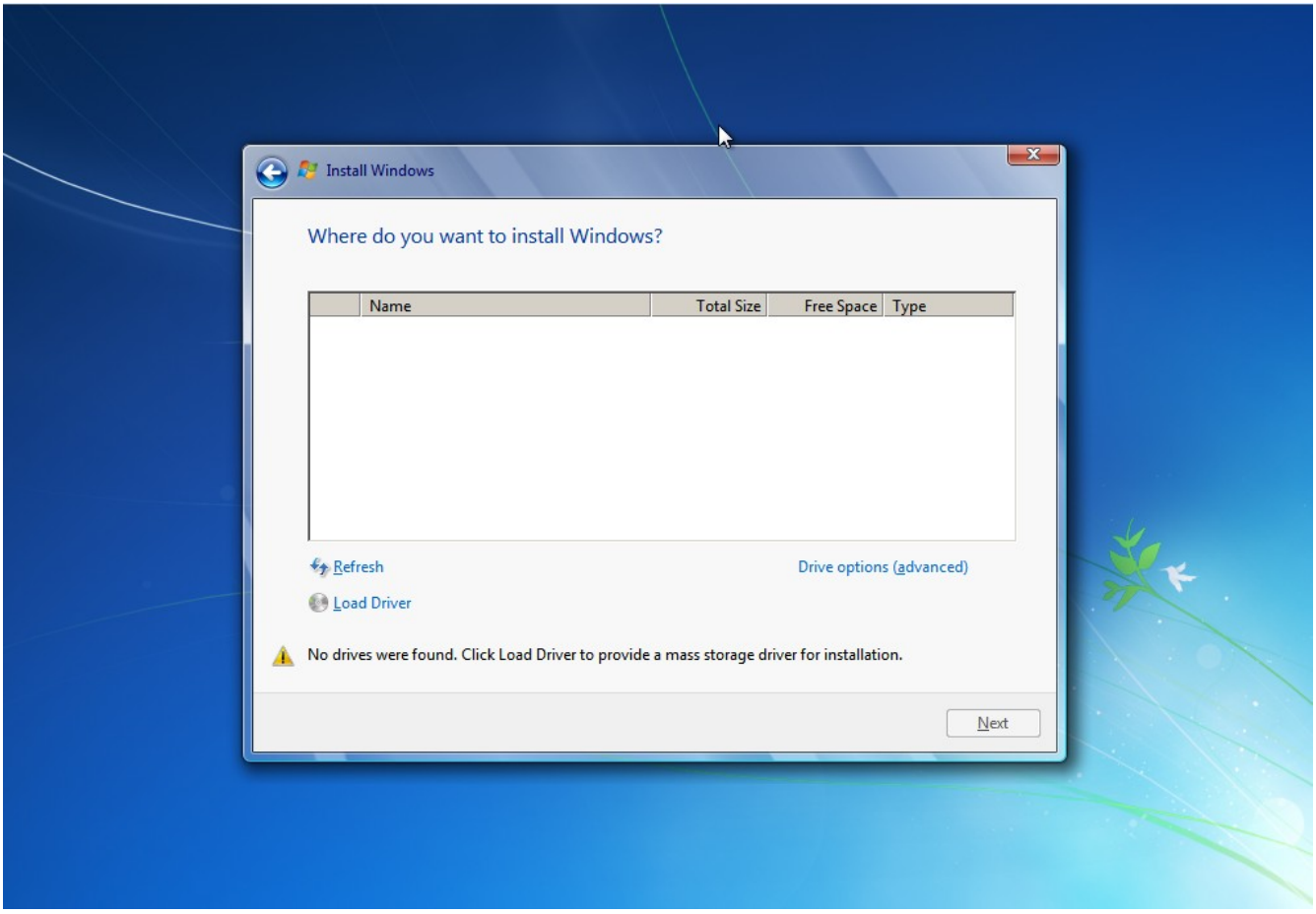
NOTE: to escape the Console you can press <ctrl <alt> keys so you can access your Ubuntu Desktop if needed.

At which point you should see the Windows 7 Installer "boot" screen

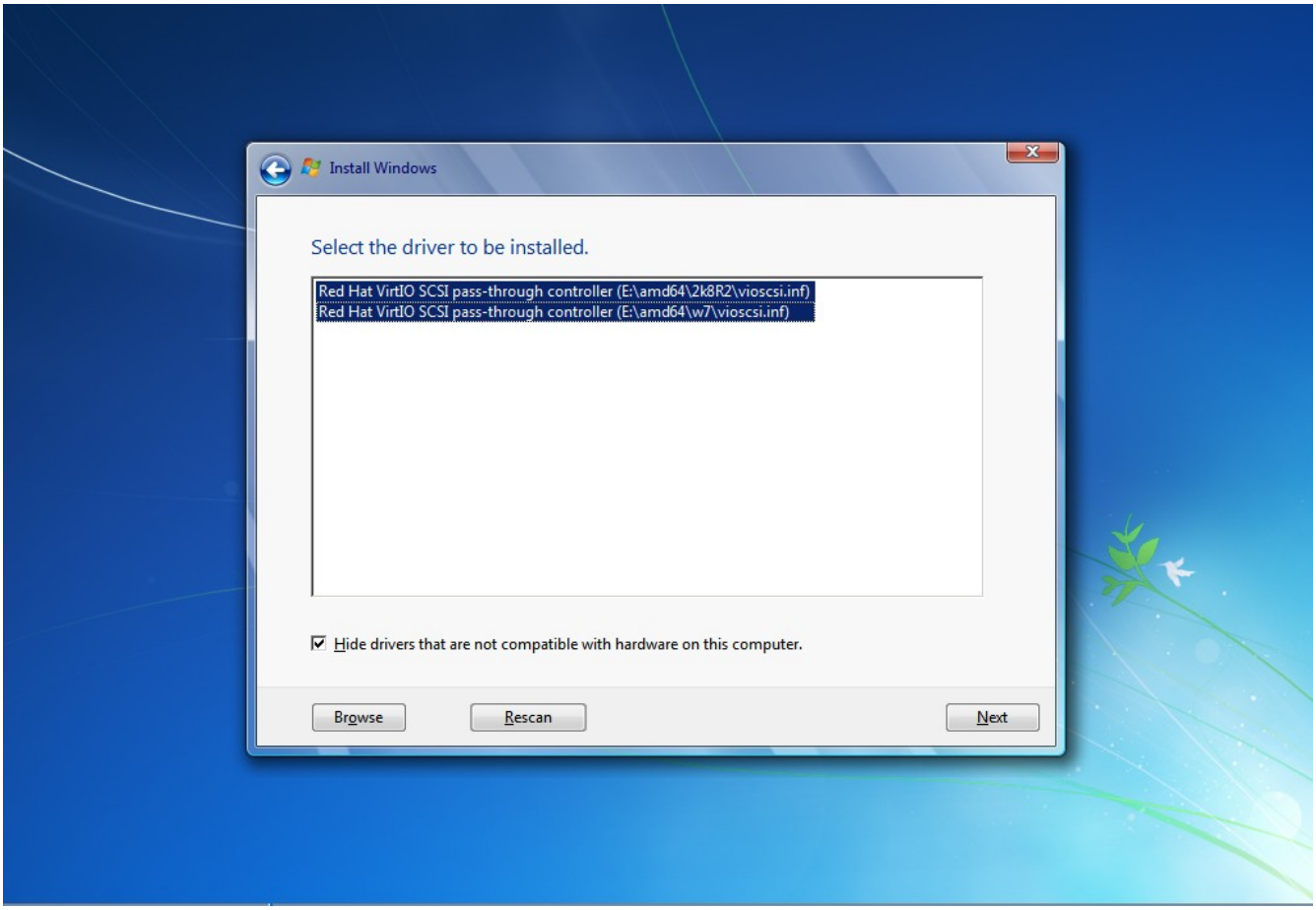


Step 5

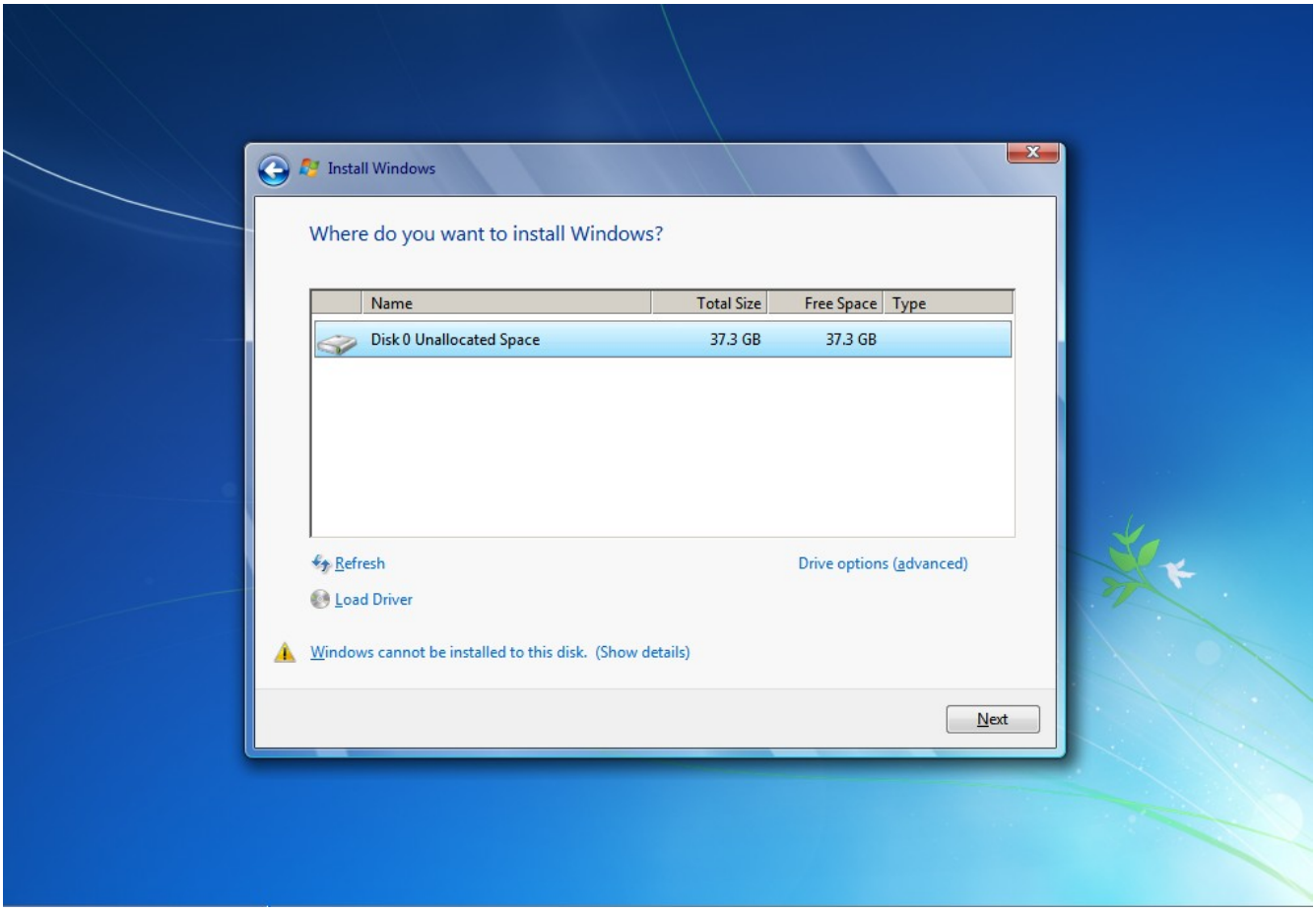
As you go thru Window's Install process when you get to this screen:



Click on "Load Driver"… and you will see:



Highlight BOTH entries and Click on Next and you will see the Next menu.



Again, click Next to install Windows to that Disk 0

Installing Windows usually takes about 15-30 minutes so just keep watch until you see Windows tell you Installation is complete.

From this point on, if you've installed Windows 7 before, everything should start to look familiar to you so install Windows 7 as you would normally. Create your own UserID and Password, enter your Windows Activation/Registration "Key" (or not).

NOTE:

Windows will reboot a couple times during the install.

EACH TIME Window Reboots you MUST re-execute the command:

$ lxc console win7 --type=vga

and log back in to Windows in order continue the Installation.

When Windows is done installing… you can reboot it and check things out.

Remember you will have to re-execute the command:

$ lxc console win7 --type=vga

Next we'll get into all the "magical" stuff that enables RemoteApps for your Ubuntu Host.

Step 6 – Configuring windows for RemoteApp support

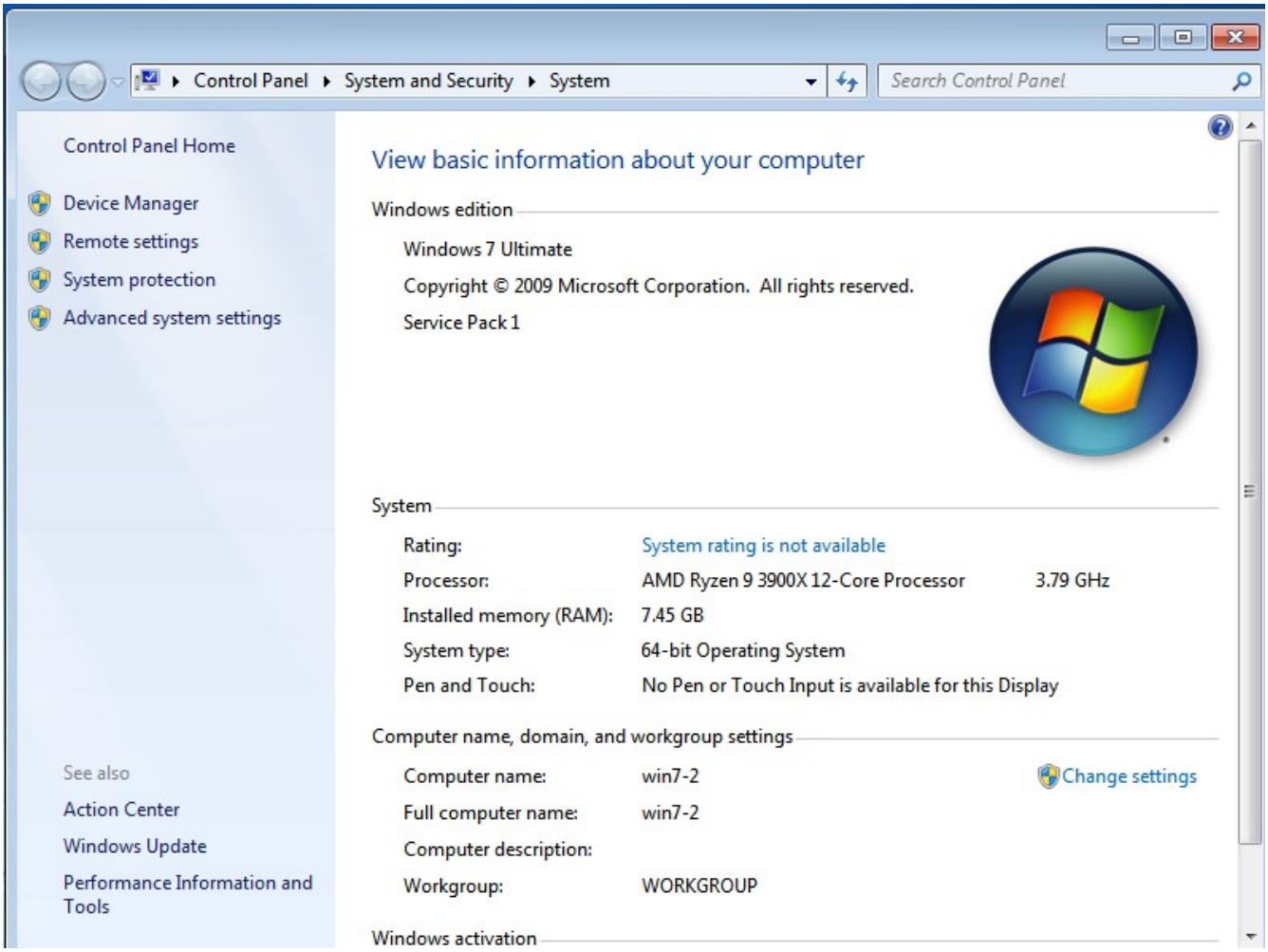
If you don't see a Windows Login Screen then again execute:

$ lxc console win7 --type=vga

Log into your Windows 7.

Click on START (lower left)

Right Click on Computer and select Properties and you will see…



Click on Remote Settings and **Select**

“Allow connections from computers running any version of Remote Desktop”

Click on Select Users

Click Add

Enter your Windows UserID

Click OK

Keep doing this if you have created other Window’s User Accounts (family etc) and when done Click Apply, then OK

Step 7

a) **Click on START**

b) **Click on Devices and Printers**

c) **RIGHT CLICK on Ethernet Controller**

d) **Click on the Hardware Tab**

e) **Click on Change Settings**

f) **Click on Driver Tab**

g) **Click on Update Driver**

h) **Click on "Browse my computer for Driver Software"**

i) **In the Box labeled: "Search for driver software in this location" enter:**

e:

j) **in the box that pops up select:**

CD Drive (E:) virtio-win-0.1.173

k) **Click OK**

l) **Click Next**

Windows will install the Virtio Driver for the Windows VM Nic so that the VM should now have Internet access.

Close all the windows till you get back to the Windows Desktop

Step 7.5 (recommended)

Create a Folder on the Windows Desktop named "winapps". Later you will copy Windows Shortcut Icons for Windows apps you want to use on Ubuntu to the "winapps" directory.

Click on your new "winapps" directory.

a) **Right Click** anywhere in the "**winapps**" directory and pick **New Shortcut**

b) **I**n the field labeled **"Type the location of the item"** just enter **"logoff"**

c) In the next field labeled **"Type the name of this shortcut:"** just enter **"logoff"**

d) Click Finish

When you click on this "logoff" icon in the future it will log you off of your Windows VM

Repeat steps a-d above but this time to create a "shutdown" icon which you can later use to shutdown your Windows 7 VM

Click on your new "winapps" directory.

a) **Right Click** anywhere in the "**winapps**" directory and pick **New Shortcut**

b) In the field labeled **"Type the location of the item"** just enter "**shutdown"**

c) In the next field labeled **"Type the name of this shortcut:"** enter **"shutdown -L"**

d) Click Finish

When you click on this "shutdown" icon in the future it will shutodown your Windows VM

At this point you might want to use Internet Explorer to download/install Chrome or Firefox (I don't like Windows Internet Explorer Web Browser).

Step 8

Using your Windows web browser search for "download .net 4.0" and click on one of the Microsoft sites listed to download Microsoft's .Net v4.0 to your Windows VM.

When the Download has completed Double Click on lower left side box labeled

dotNetFx40_Full_x….exe

Install the .Net 4.0 into your Windows VM

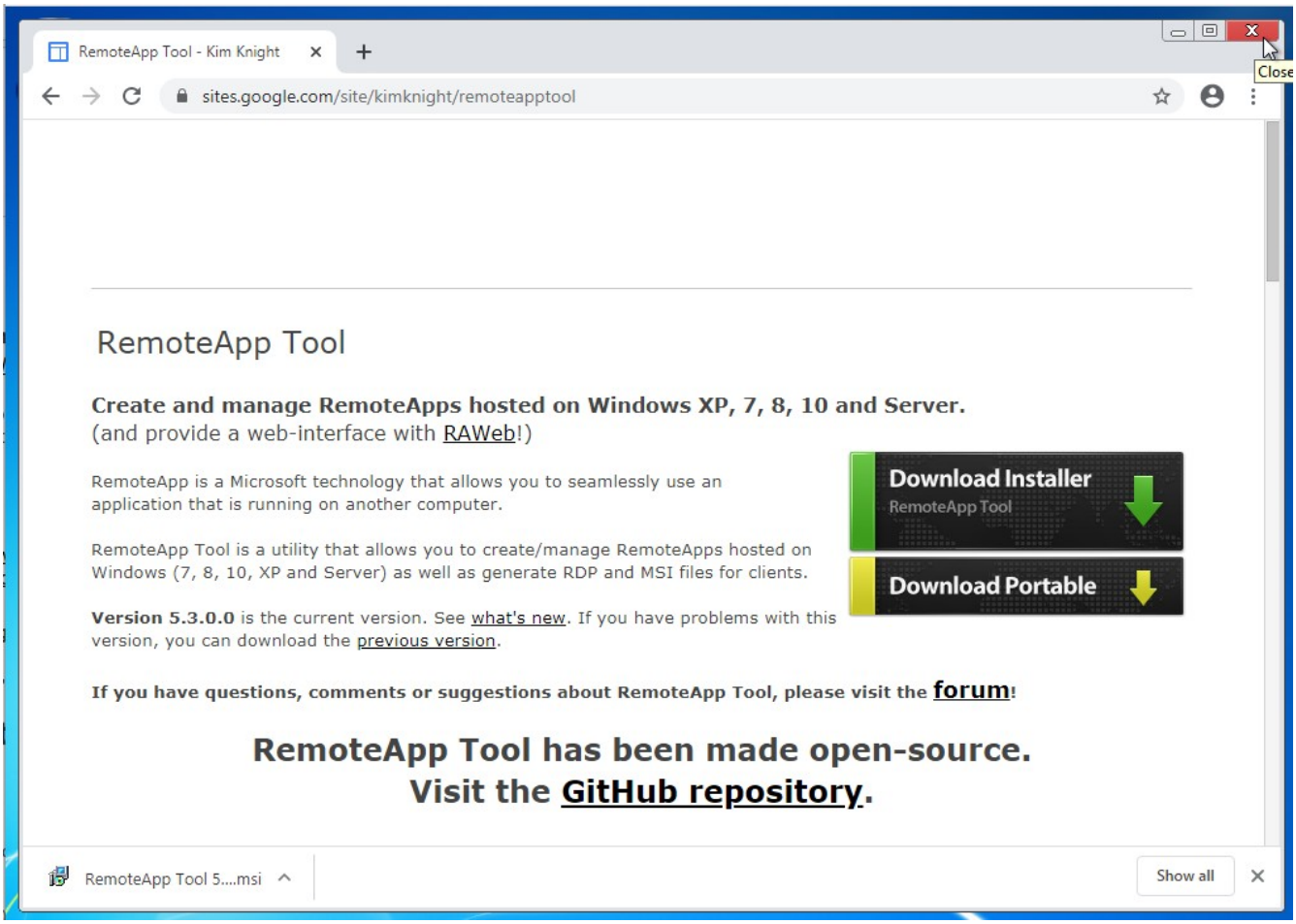
When done Reboot Windows then login again using:

```
$ lxc console win7 --type=vga
```

Step 9

Using your Windows web browser go to Kim Night's RemoteAppTool web site and download the RemoteAppTool to your Windows VM:

https://sites.google.com/site/kimknight/remoteapptool

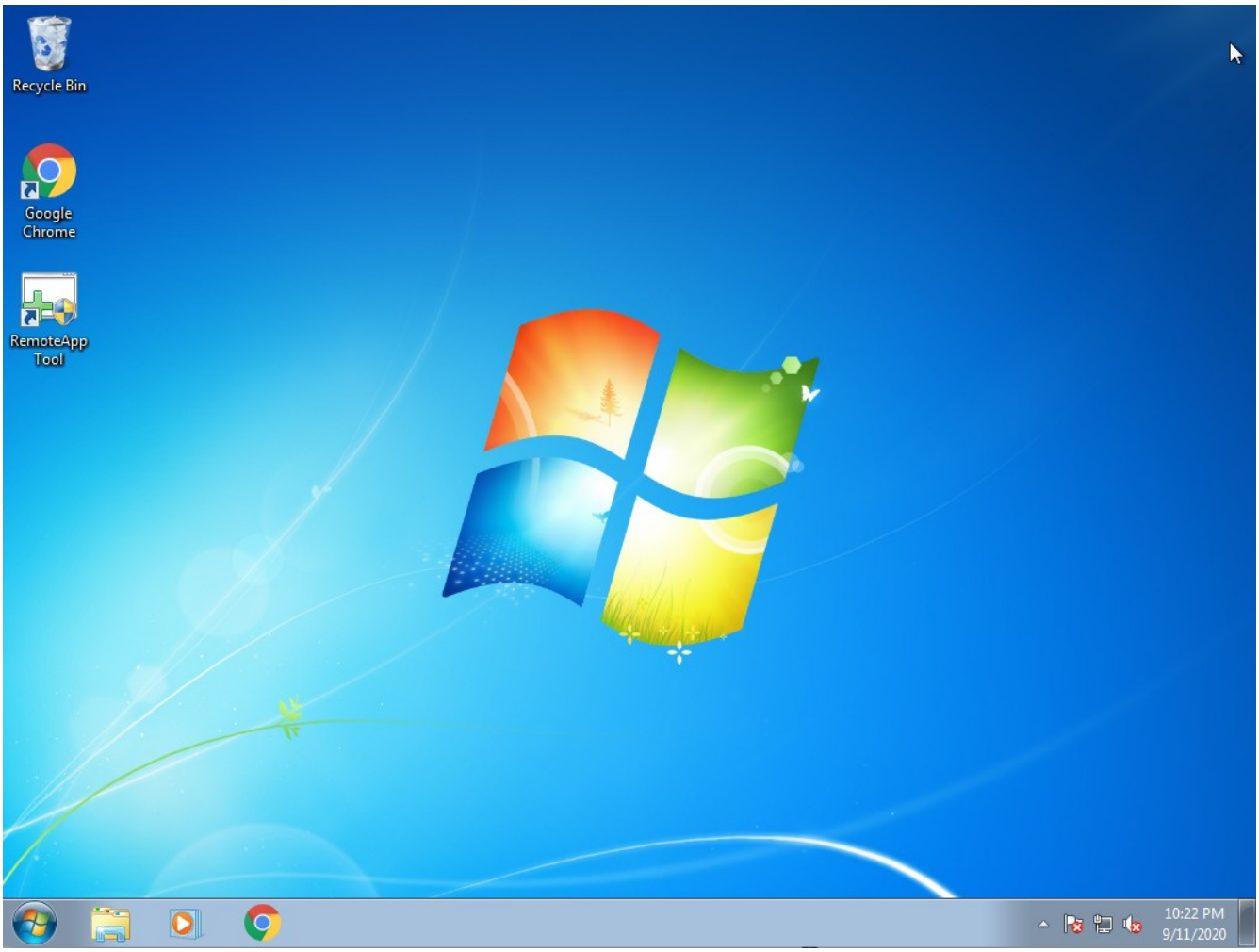


Click on **Download Installer**

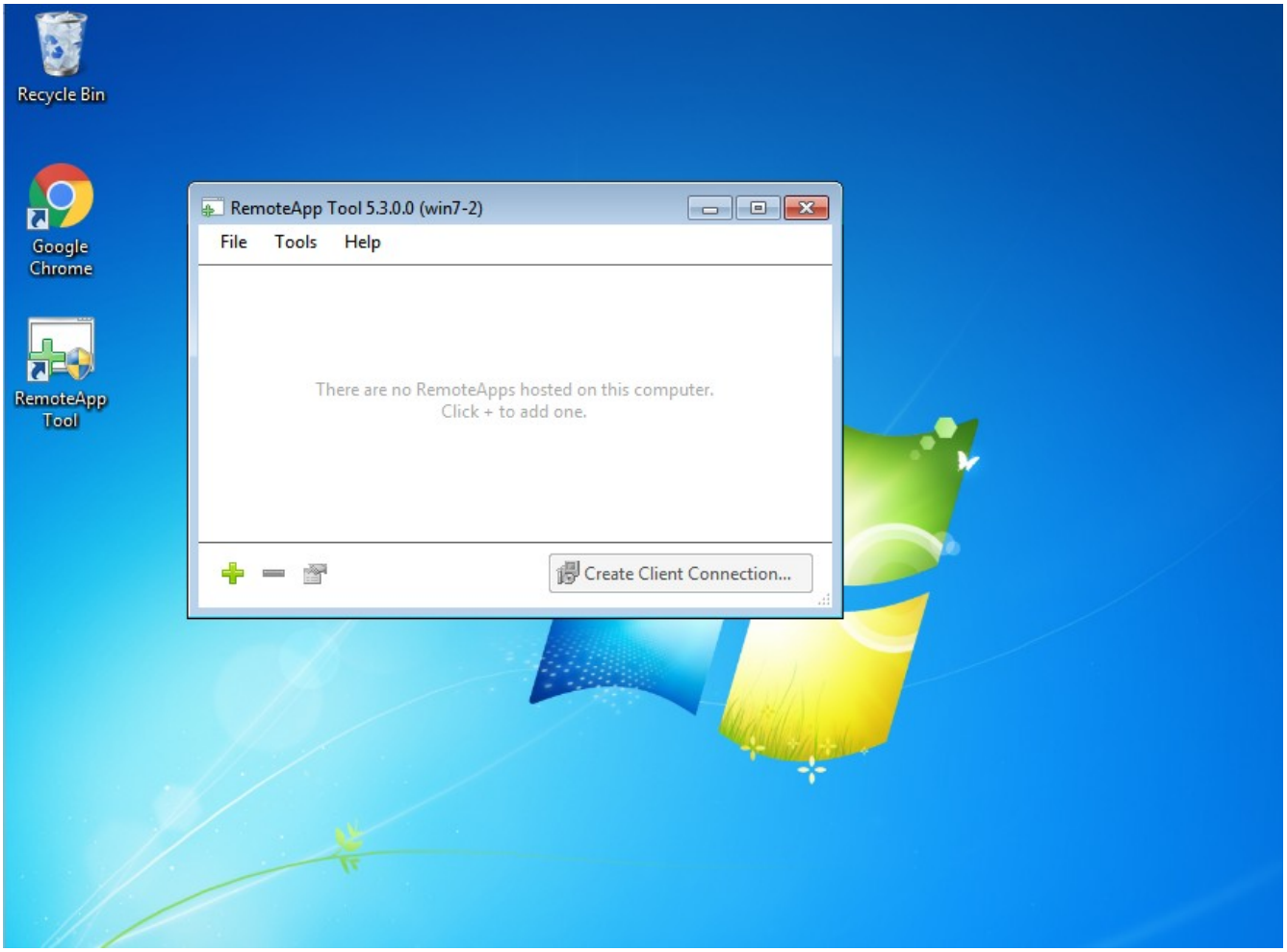
When its completed the download, **Double Click on "RemoteApp Tool...msi" (lower left) to install RemoteAppTool on your Windows VM**

Step 10

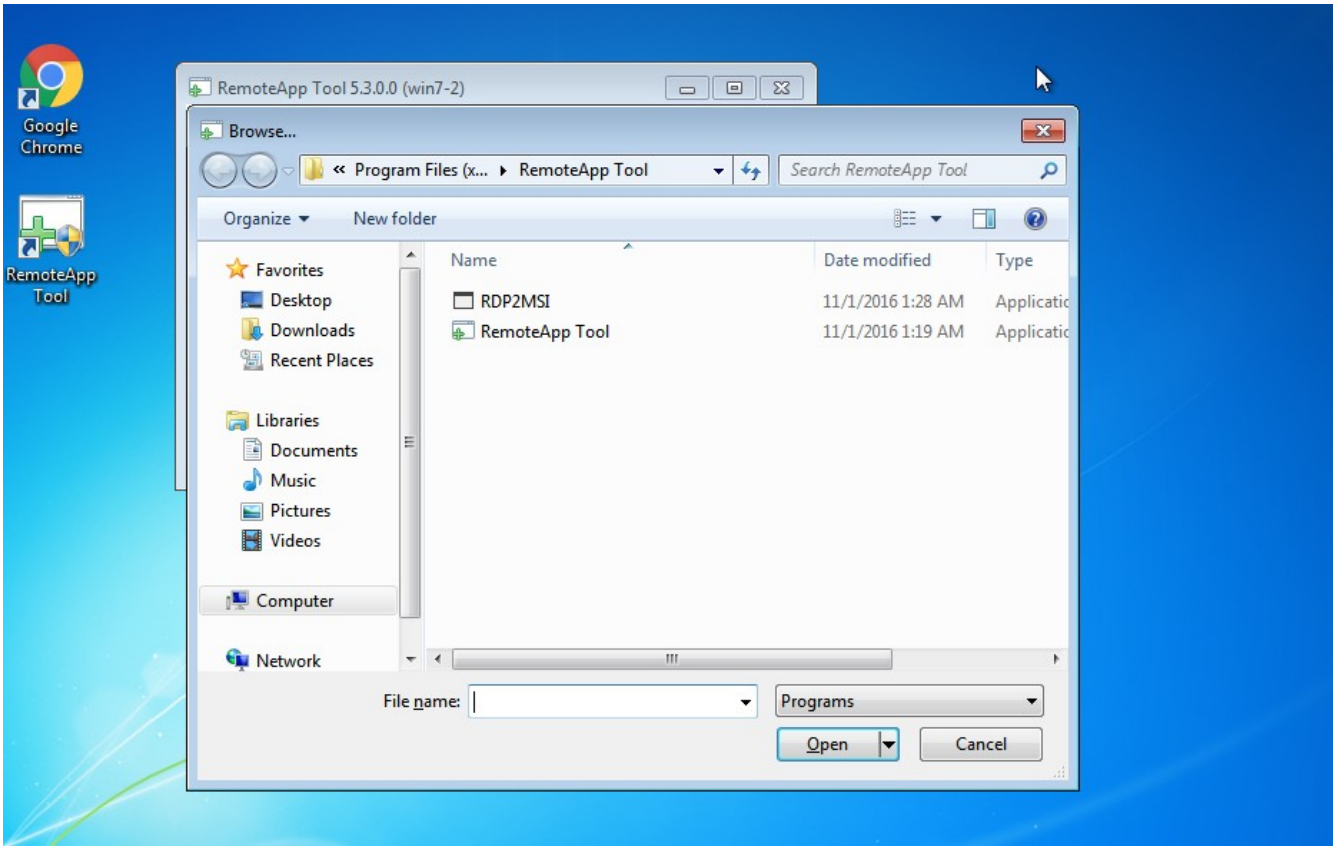
With RemoteAppTool installed you will see it on your Windows Desktop



Click on it and in the menu that pops up Click on the Green Plus Sign



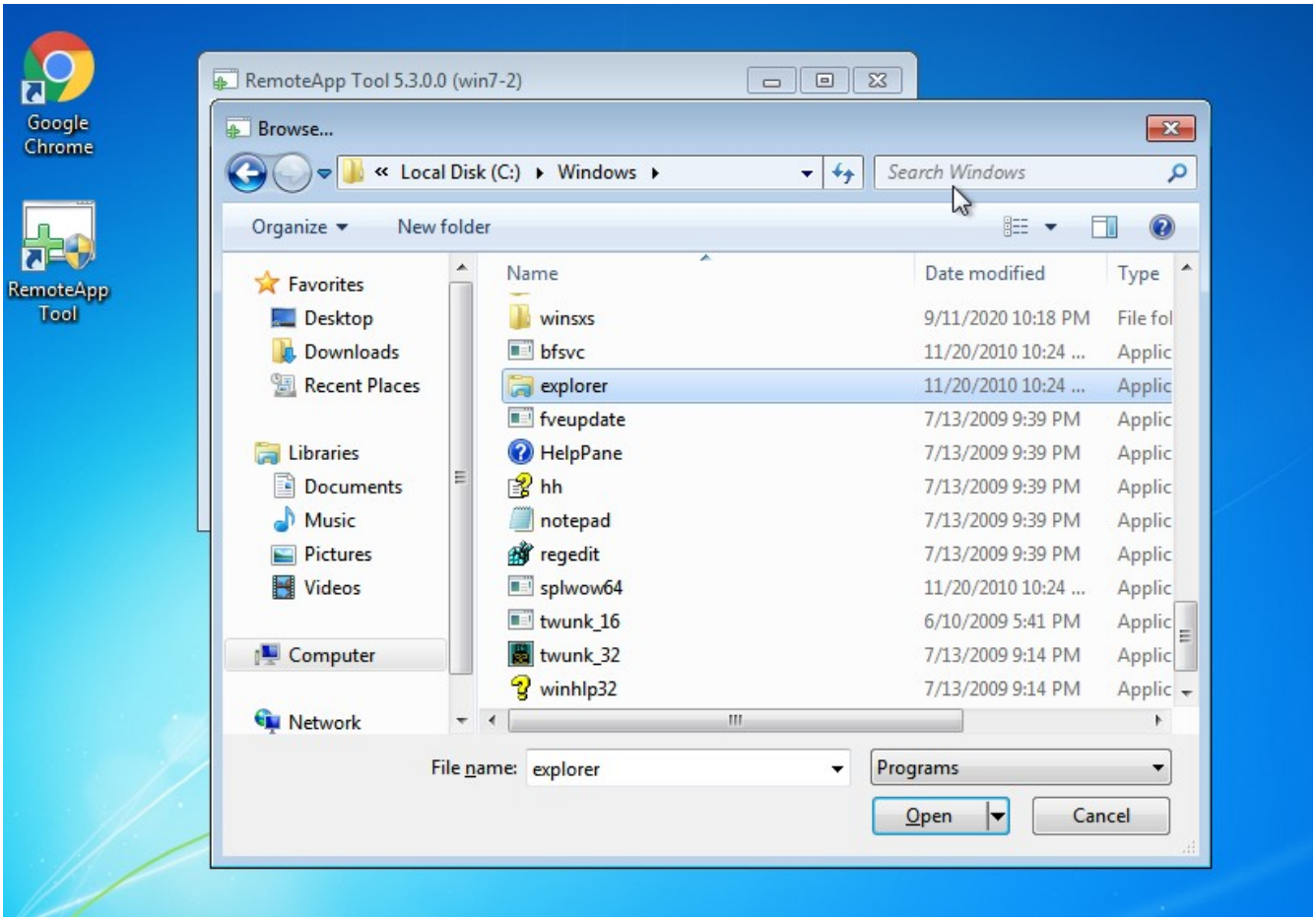
Click on Computer



Next Click on **“Local Disk (C:)**

Click on **Windows**

Scroll down and Click on (re highlight) Explorer



Click on "Open"

Now you should notice that Windows Explorer is listed as a RemoteApp in the RemoteAppTool !

Double Click on "Windows Explorer"

In the form that pops up, **ONLY change the NAME and FULL NAME** fields to just **"Explorer"**

Click "Save" and exit all the other prompt boxes **back to the Windows Desktop.**

Click on START

Click on the Arrow next to Shutdown and Select Logoff to log off of Windows then CLOSE the Windows "window" by clicking the X in the upper right hand corner.

At this point DO NOT worry the Windows VM will disappear again but it is still running !

As always you can get back to the Windows Login by:

```
$ lxc console win7 --type=vga
```

Next we will use our Ubuntu Desktop to start a RemoteApp !

Step 11

Finally, you can test out running a Windows RemoteApp from your Ubuntu Desktop.

First, create a “shared directory” that you can use to pass files back/forth between Windows and your Ubuntu Host.

$ mkdir ~/shared-folder

Next install xfreerdp

$ sudo apt install freerdp2-x11

IMPORTANT NOTE:

The following is all 1 line...

xfreerdp /cert-ignore /u:yourWinUserID p:yourWinPassword drive:shared-folder,~/shared-folder +clipboard /sound:sys:alsa /app:'%windir%\explorer.exe' /v:Win-VM-IP

You can get your Win-VM-IP IP address by doing:

$ lxc list

Next I would recommend just making a small Bash script, with an easy name to remember, that you can execute and it will execute the above for you. I called my script just “win7”

$ nano win7

!/bin/bash

again the following is all 1 line

xfreerdp /cert-ignore /u:yourWinUserID /p:yourWinPassword drive:shared-folder,~/shared-folder

+clipboard /sound:sys:alsa /app:'%windir%\explorer.exe' /v:Win-VM-IP

Obviously, put in our Windows UserID and Password

The statement “drive:shared-folder,~/shared-folder

Will set Windows up so it can access that ~/shared-folder directory you created above.

The statement “+clipboard” enables Cut & Paste to/from your Windows and our Ubuntu systems.

The statement “ sound:sys:alsa” enables Sound from your Windows system through your Ubuntu system’s speakers.

Save the bash script file.

Make it executable but for safety you only make it read/write/executable by YOU

$ chmod +700 ./win7

and I'd move it to /usr/bin

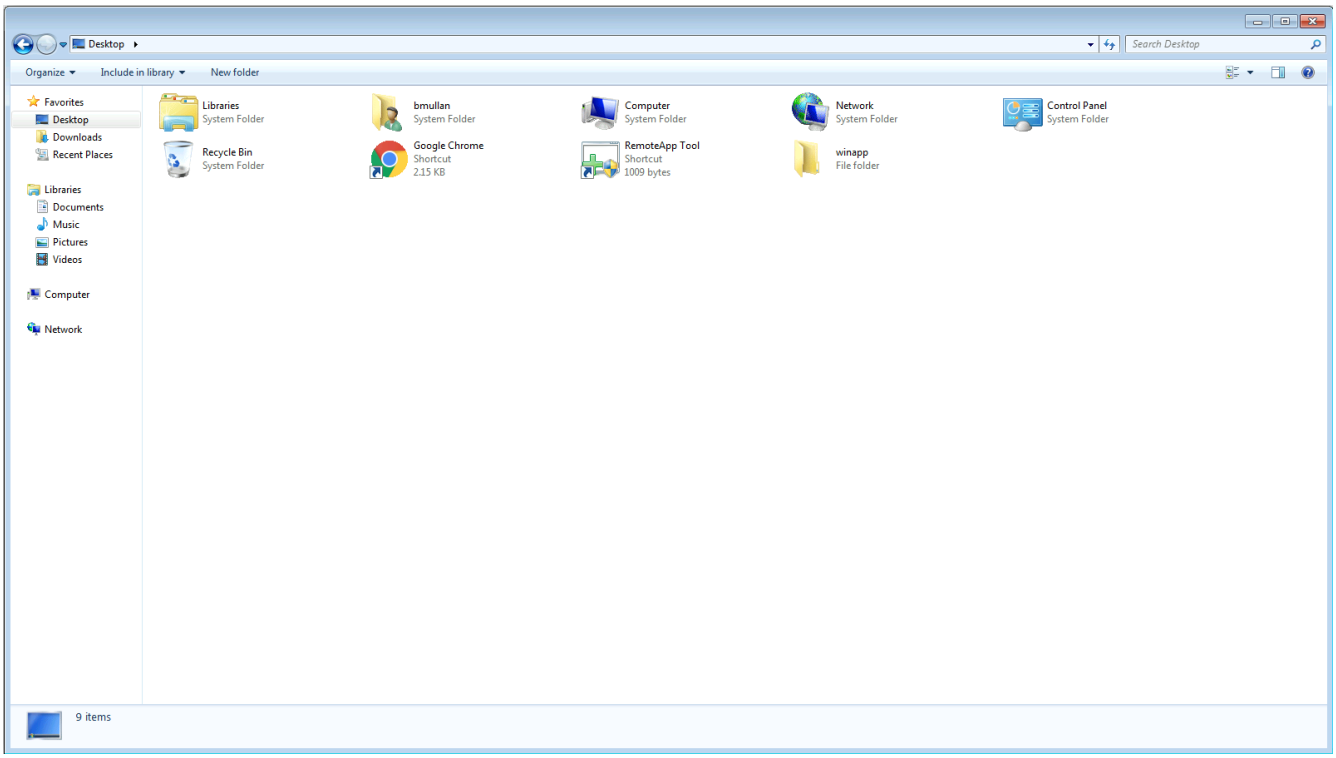
$ sudo move ./win7 /usr/bin

the above command will use xfreerdp to login to your Windows 7 using your Windows ID and Password

So now execute your win7 bash script

$ win7

and if you configured everything right you should see Windows Explorer pop up in its own Gnome Desktop Windows like below:



If you click on Desktop.. you will see your Windows 7 Desktop

Just to make things simpler for you…

When you download/install any Windows Applications and they create an Icon on your Windows Desktop

Right Click on that Icon and then ***Copy then Paste it into the "winapps"*** directory.

Later, when you execute your script you can then use the Windows Explorer window that pops up to browse to your Windows Desktop "winapps" folder and Click on any Application Icon you have there to run that Application as a RemoteApp in its own Gnome Window.

Summary

You may be wondering how this even works? After all you **only** made Windows Explorer a RemoteApp using the RemoteAppTool ***in Step 10*** !

How are all these other Windows applications also running as a RemoteApp on my Ubuntu Desktop?

Well, one unique fact that few people that have ever used Windows know is that Windows Explorer is special.

*Any program/application you click on using Windows Explorer "**inherits**" the capabilities of Windows Explorer itself !*

So if Windows Explorer was configured to be a Windows RemoteApp… then ANY program you click on with Windows Explorer is enabled as a RemoteApp also !

This is great to know because it keeps you from having to set up every single Windows Application you may want to run to individually be configured as a RemoteApp.

How cool is that right??

Brian Mullan

bmullan.mail@gmail.com