Thomas' TinyCore Browser Virtual Machine How-To

Author: Dr. Thomas Redelberger redetho@gmx.de

Version 1.0 2016-09-14

Contents

1. Legal Notices

Copyright

Suggestions for amendments and improvements are welcome.

Disclaimer

This information is provided as-is. I do not take any responsibility and do not grant any warranty nor do I imply any fitness for any purpose or kind, nor shall I be liable for any issues when using it.

Translations

The original version of this document was in English language. I might do a German version as this is my mother tongue. People are invited to do translations to other languages. But they shall properly cite and credit me and attribute back to this original.

2. Introduction

As there is malicious content on the web I wanted to create a browser environment isolated from my Windows environment. I chose TinyCore Linux to run in a virtual machine hosted by desktop Hyper-V under Windows 10 Pro 64 Bit.

I had tried various "light weight" Linux distributions. I found that TinyCore suits my requirement best and I thank the TinyCore team for providing it to the community.

3. How-To

I downloaded TinyCore-6.4.1.iso from http://www.tinycorelinux.net/downloads.html. I used version 6.4.1 because I was not able to get the then current version 7.1 to work. It hung when formatting the virtual hard-disk during the install process.

Creating the VM in Hyper-V

I created a 1 GB VHD using the Hyper-V-Manager. This size choice is not critical. All in all, TinyCore and the apps I needed took about 150 MB. The rest of the VHD could be used for file downloads because I opted them need to go onto the VHD at least temporarily.

I created the VM with the following properties in Hyper-V:

- Generation 1
- Deleted the SCSI Adapter
- Attached the ISO to the DVD drive
- Used the VHD I created in the previous step
- Attached my existing virtual switch to the virtual network card. This needs to be an "external" type virtual switch to be able to browse the web
- 1024 MB RAM, which could float between 512 MB and a much higher amount (default). It might work with less RAM
- Initially I switched of all "Integration Services".

Installing TinyCore under Hyper-V

After starting and connecting to the VM I could see it boot into the TinyCore desktop.

There were two mouse cursors: an arrow, which moved fast and was the "Linux" cursor, and a small square which was not accelerated. This was not only ugly but also annoying as sometimes you cannot get with the mouse close to the border of the window.

Before I could rectify this I needed to set-up how changes to this Linux system are persistent i.e. how changes survive a re-boot.

As TinyCore is bare bones - as opposed to CorePlus - I first needed to download the TC-Install tool. I started the "Apps" tool, which

- first checks for the fastest mirror.
- In Apps, I chose "Cloud (Remote), Browse"
- I typed *tc-install* into the text box and hit RETURN
- tc-install.tcz appeared in the left panel
- After selecting it with the mouse, a summary appeared in the right panel
- Hitting "Go" next to the "Onboot" button triggered the download
- The TC_Install icon was added to the apps on the desktop

I ran TC_Install and selected:

- "Whole Disk" which displayed fd0, sda, sr0
- I selected sda, which is the VHD I created earlier
- The next panel is "Formatting Options"
- I selected "ext2" rather than the "ext4" default. I had read somewhere on the web that in a VM we do not need the ext4 features and ext2 would cause less disk writes which helps VM performance
- The next panel is "Boot Options". I settled with the following (my comments in brackets):
  - quiet (I do not want to see all kernel boot details)
  - psmouse.proto=imps (this enabled the wheel on my PS2 mouse)

- video=hyperv_fb:1280x1024 (this was my target screen resolution)
- showapps (I wanted to see which extensions are loading)
- home=sda1 (this is where the home directory gets stored on VHD)

I finished TC-Install.

I noted that TC-Install also added its own ideas of waitusb and tce to the boot options.

Next I shut down TinyCore. The "install" copied only the core system on the VHD. Extensions like the TC-Install were lost, but I did not need it any longer anyway. The Exit button dialog would show Backup Options "None".

The next step was to remove the ISO from the VM in Hyper-V manager and start/boot again. You should then see the same desktop as before but this time it came from the VHD rather than the ISO.

Using the Apps tool, I downloaded and installed then:

- mc.tgz Midnight commander as a simple "Explorer" and editor, which also works on the command line if needed
- kmaps.tgz to accommodate my keyboard layout
- bftpd.tgz a simple FTP server to be able to exchange files between the VM and Windows
- firefox_getLatest.tgz to be able to install Firefox.
- Xorg-7.7.tcz to solve the mouse integration issue.

I ran firefox_getLatest right away. There was no icon for it on the desktop. I needed to right click on the desktop to find a link in the pop-up box. It loads more extensions from the web and ends with the message:

"Remember to add firefox.tcz to your onboot.list"

Hence I went to

SystemTools, Apps, Onboot Maintenance

looked for firefox.tcz, selected it and clicked "Add Item".

I then amended the boot options. For this I opened a terminal and typed

$sudo mc

because to change that file requires root privileges. Alternatively, I could have typed

$sudo editor

which would have fired up the standard GUI editor which comes with TinyCore.

I navigated to

/mnt/sda1/boot/extlinux

and edited extlinux.conf by hitting F4 in mc.

I amended

waitusb=5:UUID=... to waitusb=15:UUID...

because the VHD seems to need some more time to be available sometimes.

Configuring for a non-US keyboard

I added also

kmap=/qwertz/de-latin1-nodeadkeys

because I use a German keyboard.

Finally, I hit F2 to save and F10 to exit.

I rebooted. The Exit button dialog did now show Backup Options "Backup" and the target path below showed sda1/tce.

Now the bigger desktop 1280x1024 appeared and mouse integration was very good then.

However, the kmap.tcz extension with the related entry in extlinux.conf only works on the text console and Xvesa and Xfbdev servers, not the Xorg which was active now.

For Xorg to recognise my keyboard, I had to add the following lines:

```
Section "InputClass"
    Identifier "Keyboard Defaults"
    MatchIsKeyboard "yes"
    Option "XkbVariant" "nodeadkeys"
    Option "XkbLayout" "de"
    Option "XkbOptions" "terminate:ctrl_alt_bksp"
EndSection
```

as a file "mykybd.conf" in the directory

/usr/local/share/X11/xorg.conf.d

Seems the exact file name is not relevant, but it might need to end in .conf. After that I did Exit "To Prompt" from the desktop icon and then

$startx

to get back to X again and get the other keyboard layout.

Exchanging files with the TinyCore VM

The bftpd server was already loaded in memory but was not running yet. I used *mc* to copy

/usr/local/etc/bftpd.conf

which is kind of a template to

/opt/bftpd.conf

I made the following changes to /opt/bftpd.conf :

ALLOWCOMMAND_DELE="yes"

such that files in the "Downloads" folder could also be deleted from Windows.

Under "user ftp {" I changed

ROOTDIR="/home/tc/Downloads"

because this is the default Firefox download directory.

I put "#" before DENY_LOGIN

In the file

/opt/bootlocal.sh

I added a line

bftpd -d -c /opt/bftpd.conf &

which starts the server some point in the boot process. Finally, I created the directory "Downloads" in

/home/tc

as this is the default download directory in Firefox.

To be able to access the VM reliably I put a fixed IP from my local LAN using the Control Panel "Network" Applet. This created a file

/opt/eth0.sh

and created an entry in

/opt/bootlocal.sh

to start /opt/eth0.sh

I then added the boot option

nodhcp

in the file extlinux.conf to prevent that an IP address is requested in the first place and decrease boot time a bit.

I needed to amend the file

/opt/.filetool.lst

because it contained an entry to both store the directory opt and the file opt/eth.sh, so I deleted the latter entry.

I did need to add a line to store

/usr/local/share/X11/xorg.conf.d/mykybd.conf

so my keyboard configuration would survive a reboot. I learned you do need to omit the leading slash of the path.

The file /var/log/Xorg.0.log might be useful to find out issues with Xorg but I did not need to dig into that.

Polishing

When checking

Apps, Maintenance, Onboot Maintenance

I saw that Xvesa.tcz and firefox_getLatest.tcz get loaded on boot. As these were not used anymore, I removed these entries.

You might want to edit

~/.setbackground

to have a solid desktop background with the colour of your choice and no or a different logo on the desktop.

4. Integrating with Windows

Using the VM from a non-privileged Windows account

Most Hyper-V administration tasks can be done from a non-privileged account, provided that the user is added to the group of "Hyper-V Administrators". You might want to use the User Manager tool *C:\Windows\System32\lusrmgr.msc* for this. However, it appeared that some aspects of manipulating virtual hard disks in Hyper-V could still only be done with administrator privileges.

Rather than using the Hyper-V Manager, you could also access the VM via a desktop shortcut

C:\Windows\System32\vmconnect.exe localhost "TinyCore 6.4.1"

where the string between the double quotes is the name you had given to the VM.

Having the browser just available when needed

Linux *does* have some support for Hyper-V built in. Hence in the Hyper-V manager settings of the VM, under "Integration Services", I finally switched on "Operating System Shutdown".

This allows to automatically shut down the VM when you shut down Windows. To enable this, I went to settings "Automatic Stop Action" and selected "Shut down guest". Linux will then perform a proper shutdown.

Under settings "Automatic Start Action" I have the VM always automatically start when Windows starts. It will just sit there running but hidden until needed, until I open the window using the desktop shortcut mentioned above. Likewise, I just close the VM window when I do not need the browser. A double-click on the desktop short-cut brings it back immediately.

It appears that such a VM starts independently from any user logged on to Windows. When you log off and somebody else logs in, he will find the VM as you left it. Be mindful of privacy! Of cause that other user needs to be part of the Hyper-V administrators group to be able to connect to the VM.

I did not dare to switch on any other integration services in Hyper-V for this VM. I am happy to receive advice which of the other options are supported in Linux and what they are good for.

Protecting the VM

After I had set-up the VM to my preferences, I switched on a "Standard Check Point". I chose that over the "Production Check Point" because I am unsure whether Linux supports the latter which seems more advanced.

"Applying" the check point will bring the VM back to that state, wiping all impact that browsing the web had. The VM needs to be shut-down to do this.

I read that TinyCore itself can be installed in such a way that it can be made very robust. But I find this VM check-point solution quite easy to do.

Load on the host machine

I found that such a TinyCore VM eats few host CPU while idle.

Copy/Paste Issues

Clipboard copy and paste between Windows and the TinyCore VM did not work.

To be able to bring a Windows clipboard content into the virtual machine I wrote a small *vbs* script to transfer the clipboard text content into the VM using FTP. I do not support anything else beyond unformatted text though.

Building on that I wrote a script that runs in TinyCore as a small demon to wait for a file containing an URL. If the file is present it will read the URL and fire up a Firefox browser window.

5. Appendix

Remarks about my extension choices

mc.tgz

I use the Midnight Commander (mc) as a simple file explorer and editor, which also works on the command line. I prefer to see a directory structure visually but I do not want to install one of those X file explorers.

TinyCore comes with an editor running on the X desktop called "editor". But occasionally I *did* have the need to edit files while X was not running and the built-in editor of mc is sufficient for me. I did not want to deal with these classical "vi" or so tools.

bftpd.tgz

Rather than using the usual SMB to exchange files over the network with the host Windows machine, I opted for a simple FTP server to be able to exchange files between the VM and

Windows. I had copied this FTP solution from "BrowserLinux", an earlier Puppy Linux based solution for a secure browsing environment. I like to give credit to its creator but I could not find that info back.

I think exposing an FTP server and let Windows access it is less intrusive than having SMB logging on to Windows to get access to Windows shares.

Firefox

I had tried Chromium but some web sites complained about an "outdated" browser thus I needed to go for Firefox

Xorg

I learned that there are basically three software choices to provide the X-windows platform for the GUI: Xvesa, Xfbdev and Xorg. TinyCore comes with Xvesa built-in. Both Xvesa and Xfbdev are "light weight" compared to Xorg.

But I could not resolve the mouse integration issue with both Xvesa and Xfbdev. Even worse: Xvesa and Hyper-V would only work reliably together for a 1152x786x24 resolution. Other resolutions would yield a corrupted window when closing and opening again the VM window. This is especially relevant when you have the VM starting when booting the host PC, which I wanted to do.

Hence I needed to settle for Xorg which was easier to set-up than I thought.

* * *