Messin' with BackTrack 3.0

Learning Through Installing Common Tools and Creating Custom ISO images

by: Gene Bransfield Jr.

garconcoffee <at> gmail

"If it's stupid but works, it's not stupid."

Murphy's Laws of Combat, #2
www.nightstalkers.com/mythology/murphy.html

Table of Contents

Introduction:

BackTrack is a suite of penetration testing/vulnerability assessment tools installed on a Linux operating system, all wrapped-up on a bootable CDrom. You can put the BackTrack CD in your CDrom drive, boot from the CD and pentest to your heart's content. I like BackTrack a lot because it has all the tools (well, almost all the tools) that anyone (well, almost anyone) could need to do a pen test.

I thought that BackTrack was a wonderful thing, but just needed a few more things added to it to make it the ultimate distro for my needs. So I installed the CD to my hard drive and installed the stuff I thought it was missing. After I customized it, I thought it would be great to create a custom CD that reflected my hard drive installation. I set out to do just that. So as I searched high and low throughout the Internet attempting to figure stuff out, I thought: "Wouldn't it be great if there was a one-stop source of this information?" So I created one.

Disclaimer:

The following instructions worked for my particular situation. These steps are particular to my VMware situation and my physical hardware. I may provide certain advice one way or the other, but please do some research with regard to your particular environment and make decisions independently where warranted.

I know a good bit about Linux, but am by no means an expert. I am sure that there are sexier ways to complete some of these steps in this guide. However, I subscribe to Murphy's Laws of Combat, Law #2 (MLOC2) which states "If its stupid but works, its not stupid" (note: nightstalkers web link no longer works; don't know why). These steps resulted in a working LiveCD distro that I utilize for penetration testing and vulnerability assessment, so I'm happy. I'm just trying to share my new-found knowledge with all those who care to know about it.

This guide is intended for people who are interested in the topics discussed herein, and is written for an audience that probably has never done this before. I am attempting to make a guide that is useful to all, and I am not trying to insult anyone's intelligence. Please do not be offended if I seem to explain certain commands or concepts in too much detail. Some of us are still learning.

What you will need:

-- VMware server installed
-- Either VMware server console of VMware client (to manage the VMware server)
-- 12GB of hard drive space on the host machine (could probably get by with 10, but I recommend 12).
-- Internet Access (or all of the necessary downloads on CD)
-- DVD Burner (this process creates an ISO image that is just under 1GB in size. CD won't do that).

INSTALL BACKTRACK IN VMWARE FROM ISO:

****This section contains a modified version of "BackTrack Hard Drive Installation" by jabra [at] remote-exploit [dot] org. We thank him for his contribution.

First, download the BackTrack 3.0 iso image. Save the image as **bt3.iso**.

Install VMware Server and connect using a VMware Client

At the VMware console, select "Create a new virtual machine," and the 'New Virtual Machine Wizard' will appear. Click 'Next.'

-- Custom
On the next page, select 'Custom' and click 'Next.'

--Linux
--Other Linux 2.6.x kernel
On the 'Select a Guest Operating System' page, select 'Linux' as your operating system, and at the 'Version:' drop-down menu box, select 'Other Linux 2.6.x kernel.' Click 'Next.'

--Name the VM
--<Edit path to VM as necessary>
On the 'Name the Virtual Machine' page, under the 'Name' field, enter the desired name for the machine

you are creating. This is the name by which VMware will reference your new creation. Standard practice is to call it "BackTrack," but we can get creative as well (i.e. "NinjasRule," "PiratesRule," "NinjasSuck," etc). I'll call mine 'Test.' Review the Location field and make sure VMware is putting your machines in the directory that you desire. Click "Next."

> Editor's note: commas are placed within the quotations because that's how I learned how to do it in school. It might not be a good idea to include commas in your naming convention.

--Select 'One'

On the 'Processor Configuration' screen, select as appropriate. If you've got a Pentium Core Duo, I'd recommend choosing 'One.' Click 'Next.'

--Make this virtual machine private.

On the 'Set Access Rights' page, it would probably be a good idea to select 'Make this virtual machine private' unless you have a good reason not to do so. Click 'Next.'

--Use half of available memory (i.e. if you've got 1GB memory, set your VM memory to 512M)

On the 'Memory for the Virtual Machine' page, I recommend choosing a memory size of exactly half of the available memory on your box (i.e. if you have 1GB of memory, choose 512 MB). If you want to change it to be something else, be my guest. Click 'Next.'

--Use Bridged Networking

On the 'Network Connection' page, I recommend choosing 'Use bridged networking.' This will allow the Virtual Machine to have an independent IP address on the network and act like a real host. If you want to change it to NAT, that will work, too -- but I haven't tested that configuration. The other two choices will not allow you to use your virtual machine to directly connect to the network. Click "Next."

--LSI Logic

On the 'Select I/O Adapter Types' page, my configuration chooses 'LSI Logic' by default, so I just leave that alone. If you feel you need to change it, then by all means do so. Click 'Next.'

--Create New Virtual Disk

On the 'Select a Disk' page, choose 'Create a new virtual disk' unless you have reason to do something else. Click 'Next.'

--IDE

On the 'Select a Disk Type' page, choose 'IDE.' BackTrack has trouble seeing virtual SCSI disks (i.e. VMware disks of type SCSI). Since we're using IDE instead of SCSI, all disk types will be hda.

--Disk size 12GB

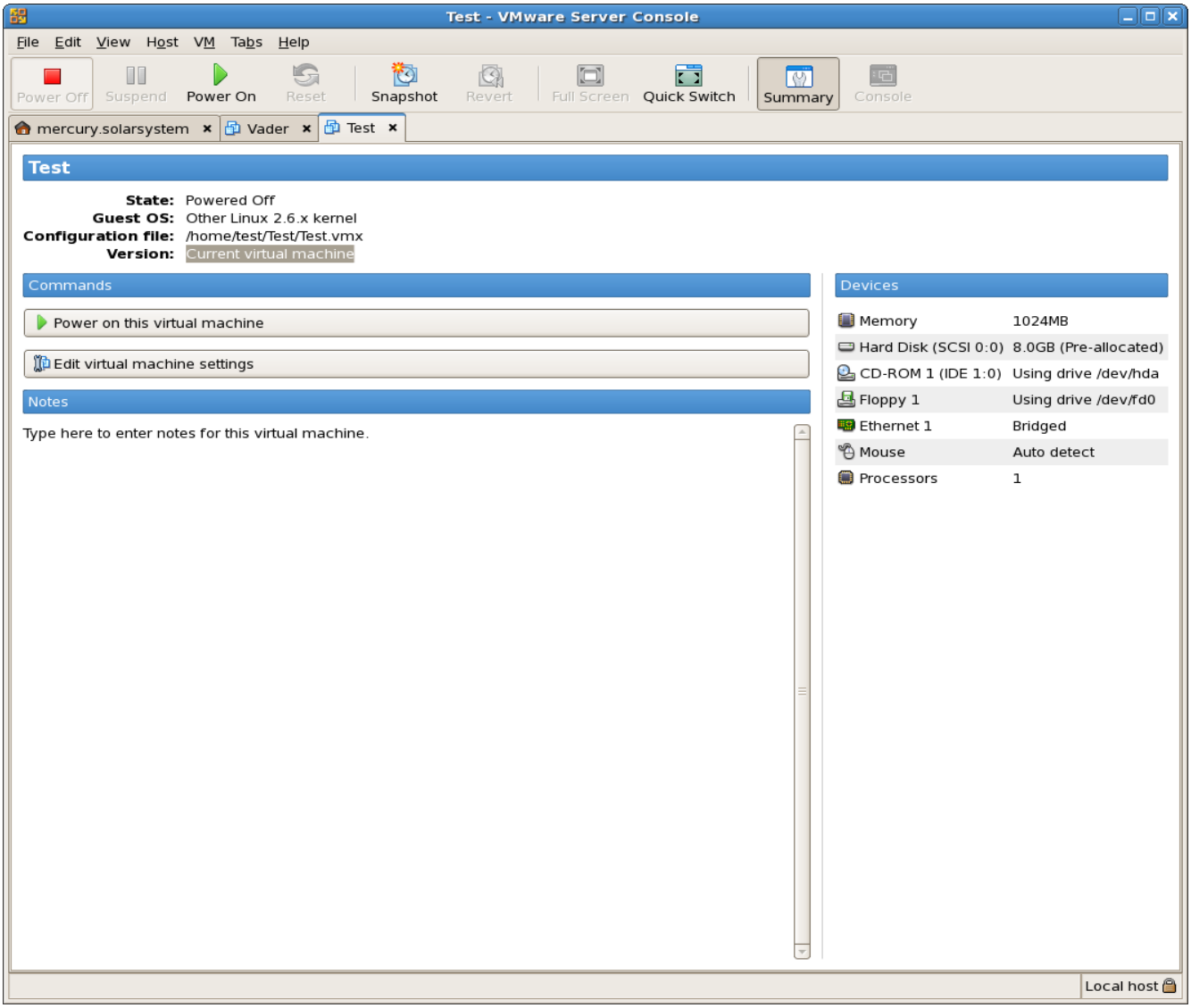
--Allocate all disk space now

--Split disk into 2GB files

On the 'Specify Disk Capacity' page, choose a disk size of at least 10GB (but I will recommend 12GB). Creating the ISO from the running operating system will require some extra space. Also, choose 'Allocate all disk space now' and 'Split disk into 2GB files.' Click 'Next.'

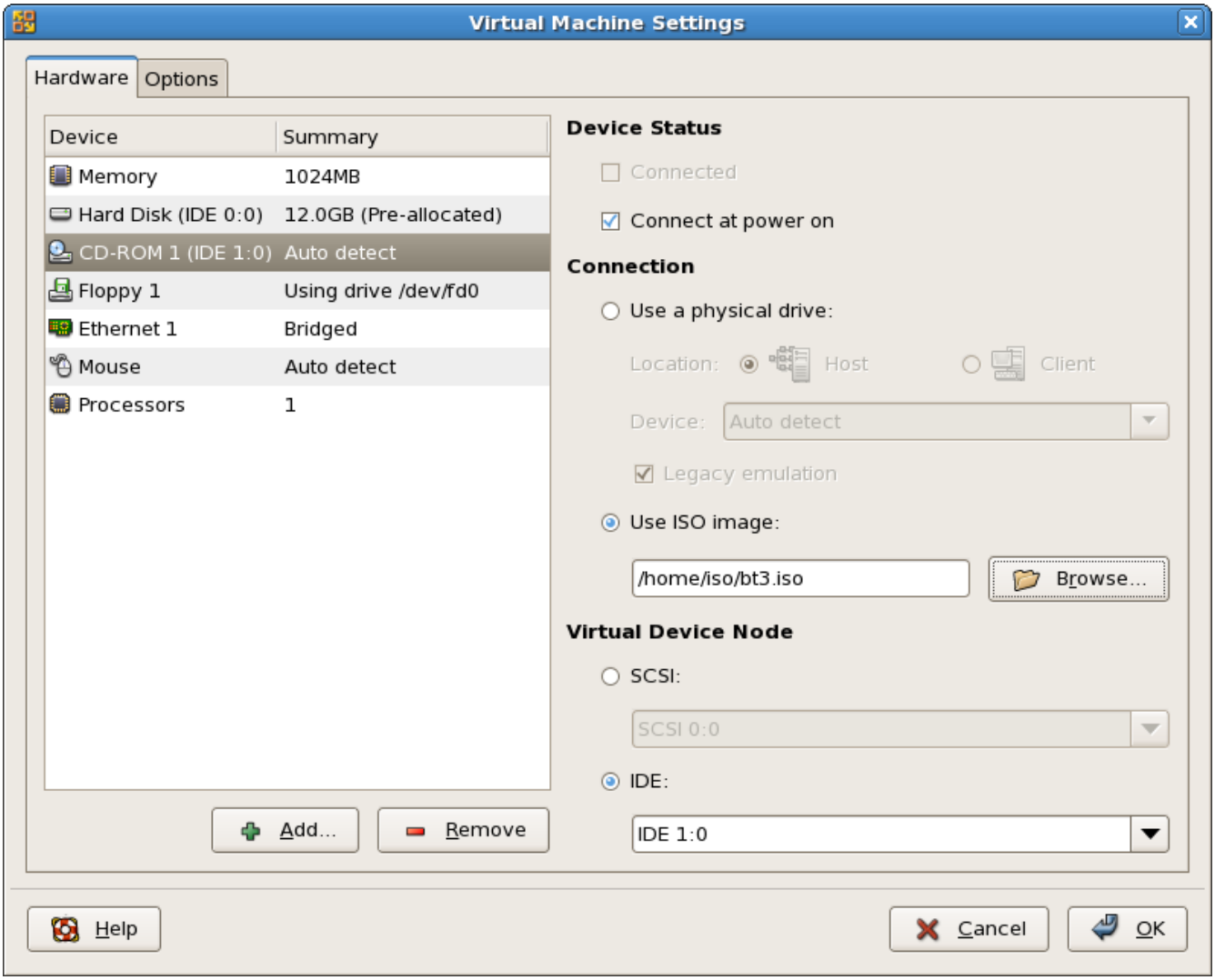
--Accept default setting (i.e. <YourVMName>.vmdk is fine, so hit [enter])

On the 'Specify Disk File' page, accept the default. This will place the .vmdk file into the directory specified in the 'Naming the virtual machine' page. Click 'Finish' and wait for the disk to be Once the disk is created, VMware should put you into a screen like this:



--Double-click CD-ROM 1 and set it to the bt3.iso

On the right-hand side of the screen, you'll see a column labeled 'Devices:'; the third device down is labeled 'CD-ROM 1.' Double click on that device. This produces another dialog box. On the right-hand side of this dialog box, choose 'Use ISO Image' and click on the 'Browse' button. Now browse to wherever it was that you downloaded the BackTrack iso image. This configuration will allow you to boot from the BackTrack iso image that you downloaded.



--Now, power on the virtual machine.

The BackTrack 3 CD boots directly into Xwindows. However, if you find yourself at a login prompt (which you will after installation) login username is root, password is toor.

****The following instructions are a modified version of nokia's tutorial posted at http://www.tutorials.thetazzone.com/viewtutorial.php?p=78361. You can use these instructions to install to a hard drive as well :-).

First, we need to partition drives to prepare for the install. Pop open a terminal from the menu bar.

bt ~ # fdisk /dev/hda

Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.

The number of cylinders for this disk is set to 26630.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software fro other OS's
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):**n [enter]**

Command action
   e extended
   p primary partition (1-4)
p [enter]
 Partition number (1-4): **1[enter]**
 First cylinder (1-26630, default 1):**[enter]**
 Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-26630 default 26630): **+10M [enter]**
        --Note: This partition is for your boot sector

Command (m for help):**n [enter]**
 Command action
    e extended
    p primary partition (1-4)

p [enter]

Partition number (1-4): **2 [enter]**

First cylinder (23-26630, default 8):**[enter]**

Using default value 23

Last cylinder or +size or +sizeM or +sizeK (23-26630, default 26630): **+2048M [enter]**

--Note: Standard philosophy is that your SWAP space should be twice your RAM size.

Creating the ISO uses a lot of processor and memory, so a good SWAP size only helps

Command (m for help): **n [enter]**

Command action

e extended

p primary partition (1-4)

p [enter]

Partition number (1-4): **3 [enter]**

First cylinder (4257-26630, default 4257):**[enter]**

Using default value 4257

Last cylinder or +size or +sizeM or +sizeK (71-456, default 456): **+6144M**

Note: That's a 6GB drive (1024*6=6144). This will be the root partition.

Command (m for help): **t [enter]**

Partition number (1-4): **2 [enter]**

Hex code (type L to list codes): **82**

-Creates partition 2 as a Linux SWAP partition.

Command (m for help): **p [enter]**

Disk /dev/hda: 3758 MB, 3758096384 bytes

255 heads, 63 sectors/track, 456 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		1	104	10363+	83	Linux
/dev/hda2		23	4256	2000565+	82	Linux swap

/dev/hda3 4257 16955 60002777+ 83 Linux

Command (m for help): **w [enter]**

-Writes the changes to the partition table.

The partition table has been altered!

Calling ioctl() to re-read partition table.

Syncing disks.

bt~# mkfs.ext3 /dev/hda1

Now that we've created partitions, format drives /dev/hda1 and /dev/hda3 with Linux file system:

[bunch of output that ends with: This filesystem will be automatically checked every 25 mounts or 180 days, whichever comes first. Use tune2fs -c or -i to override.]

bt~# mkreiserfs /dev/hda3

It will ask you if it's ok that all data on /dev/hda3 will be lost. There's no data on /dev/hda3, so this is fine. Select yes.

[bunch of output that ends with: `ReiserFS is successfully created on /dev/hda3.`]

Now create the swap space:

bt~# mkswap /dev/hda2

Setting up swapspace version 1, size = 512380 kB

no label, UUID=156e48d1-bc52-445d-9a78-72072928a83f

bt~# swapon /dev/hda2

Now that we've created directories, let's install BackTrack 3!

bt~# mkdir /mnt/backtrack

creates the point to which we will mount the physical device

bt~# mount /dev/hda3 /mnt/backtrack

syntax for mount is "mount <what you want to mount> <where you want it mounted>

bt~# mkdir /mnt/backtrack/boot

bt~# mount /dev/hda1 /mnt/backtrack/boot

Just to make sure everything worked right:

bt~# df

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
tmpfs	618344	4476	613868	1%	/
/dev/hda3	6000068	32840	5967228	1%	/mnt/backtrack
/dev/hda1	10027	1124	8385	12%	/mnt/backtrack/boot

Make sure everything is mounted where it should be. Otherwise, things will go wrong very quickly.

For the remainder of these commands, PAY ATTENTION to the all '/' as they are very important

bt~# cp --preserve -R /{bin,dev,home,pentest,root,usr,etc,lib,opt,sbin,var} /mnt/backtrack

--preserve maintains all permissions associated with the files; -R makes the command recursive (copies directories and all associated files).

We're copying all necessary operating system files from the CDROM over to the hard drive. This will take some time (up to 30 minutes depending on system configuration), so don't poweroff your system until it is done.

That was fairly straight-forward, but now things get a tad-bit confusing. Stay with me!

bt~# mkdir /mnt/backtrack/{mnt,proc,sys,temp}

making those directories using one command vice a series of commands.

bt~# mount --bind /dev/ /mnt/backtrack/dev/

remounts the directory /dev (which holds the addresses for all physical devices) to our new mount point.

bt~# mount -t proc proc /mnt/backtrack/proc/

remounts proc to the specified mount point -- this will provide an interface with the system kernel

bt~# cp /boot/vmlinuz /mnt/backtrack/boot/vmlinuz

copies vmlinuz (necessary for boot) to the harddrive

bt~# chroot /mnt/backtrack/ /bin/bash

changes the root path to the hard drive /mnt/backtrack

/dev/pts/1: No such file or directory

Ignore this error; you're fine.

To boot the system properly, we're going to use lilo. You can either cut/paste this file into lilo or edit your current lilo to look like the file below. Either way, I HIGHLY recommend you 'mv /etc/lilo.conf /etc/lilo.conf.bak' first!!!

bt~# vi /etc/lilo.conf

```
# LILO configuration file
# generated by 'liloconfig'
#
# Start LILO global section
lba32 # Allow booting past 1024th cylinder with a recent BIOS
```

boot = /dev/hda

#message = /boot/boot_message.txt

prompt

timeout = 1200

Override dangerous defaults that rewrite the partition table:

change-rules

reset

VESA framebuffer console @ 1024x768x256

vga = 773

Normal VGA console

vga = normal

VESA framebuffer console @ 1024x768x64k

vga=791

VESA framebuffer console @ 1024x768x32k

vga=790

VESA framebuffer console @ 1024x768x256

vga=773

VESA framebuffer console @ 800x600x64k

vga=788

VESA framebuffer console @ 800x600x32k

vga=787

VESA framebuffer console @ 800x600x256

vga=771

VESA framebuffer console @ 640x480x64k

vga=785

VESA framebuffer console @ 640x480x32k

vga=784

VESA framebuffer console @ 640x480x256

vga=769

End LILO global section

Linux boot partition

image = /boot/vmlinuz
root = **/dev/hda3**
label = **<your label here>**
read-only
End Linux Boot partition

Save the file and run lilo from the command prompt. You should receive no errors.

bt~# lilo -v

LILO version 22.7.1, Copyright (C) 1992-1998 Werner Almesberger
Development beyond version 21 Copyright (C) 1999-2005 John Coffman
Released 17-Sep-2005 and compiled at 00:33:53 on Aug 8 2006.

Reading boot sector from /dev/hda
Backup copy of master disk volume ID record in /boot/boot.0300
Using MENU secondary loader
Calling map_insert_data

Boot image: /boot/vmlinuz
Added <your label here>

Writing boot sector
/boot/boot.0300 exists - no boot sector backup copy made.

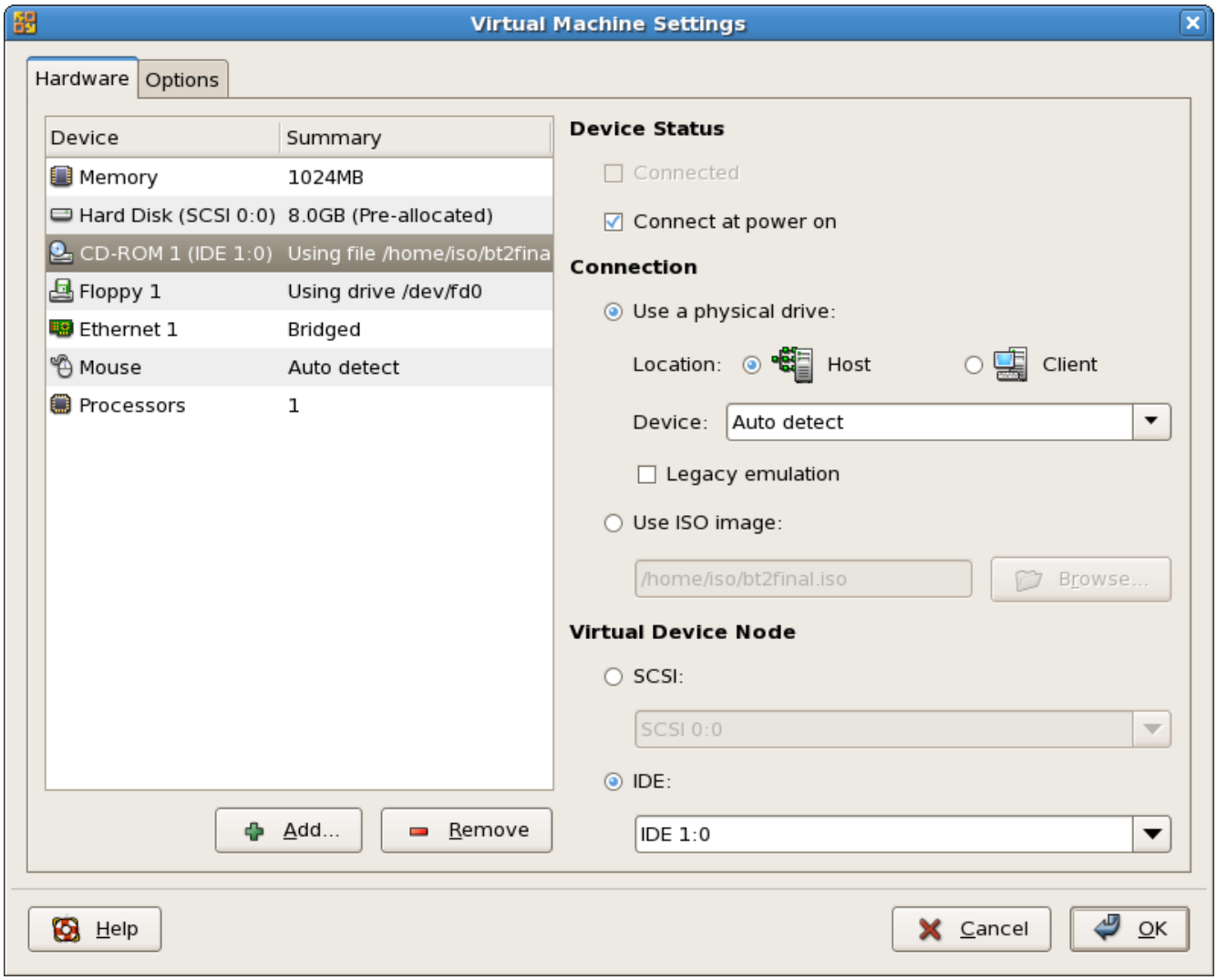
--Once again, make sure there are NO errors here AT ALL. If there are errors, troubleshoot and resolve them BEFORE rebooting.

--Once your lilo runs cleanly, it's time to take the plunge and reboot you box!

bt~# poweroff.

--Change CD-ROM back from bt3.iso to Host

Once BackTrack has shut down, you'll be back at the VMware console again. Double-click on the CD-ROM device and change the connection type back to 'Use Physical drive:' -- choose 'Host.'



Now, turn the power back on on the virtual machine. BackTrack should boot from the virtual disk.

The first thing you'll see is the lilo boot stuff. Select your BackTrack image (should be default) and press return.

Next thing you'll see is the login prompt. As previously stated, login is root, password is toor.

start Xwindows (the pretty graphical stuff) by typing :

bt~# startx

We're going to need to get to the Internet to download our stuff, so first step is to get an IP address. There are many ways to do this:

command line way: **bt~# ifconfig <interface> <your ip address> netmask <your netmask>**
bt~# route add default gw <ip of your default gateway>
(now you have to edit /etc/resolv.conf with your DNS information)
DHCP: **bt~# dhcpcd**

Uber-lazy way: **Kmenu --> Internet --> Set IP Address**

Personally, I like the Uber-lazy method myself (MLOC2), but the hard-core command line guys like doing things the hard way (and perhaps X windows won't work and you need that information someday :-)).
Anyway, now we have an IP address and we may surf the net. Please resist the urge to go to porn sites for now.

Congratulations! You now have a working copy of BackTrack in VMware. **BUT...** before you go watch the Matrix for the 1024th time, let's customize it!

Some basic things that I like (and are actually quite useful) are the following:

Allow BackTrack to boot directly into Xwindows:

bt ~ # vi /etc/inittab
change **id:3:initdefault** to be **id:4:initdefault**

Make sure your xorg.config is working before you do this :-).

Turn On Numlock by default:

Kmenu --> System --> Settings --> Hardware --> Keyboard

Under 'Numlock on KDE Startup' select 'Turn on'

Customize the toolbar and Wallpaper:

BackTrack runs KDE by default, so anything that you can do with KDE, you can do with BackTrack. Right-click on the wallpaper, and choose 'Configure Desktop.' You can also right-click on the KDE Panel (aka the toolbar) and choose 'Configure Panel.' Make your BackTrack your own!

Congratulations! You have successfully installed BackTrack in VMware!

BUT WAIT!!!!! BEFORE you do anything else, save your hard work and make sure that even YOU can't mess things up to a point that you have to reinstall anything (which may or may not have happened to me on more than one occasion).

TAKE A SNAPSHOT!!!

On your VMware Server Console, you'll notice that at the top there are several menus, one of which contains icons. Those icons are labeled "Power Off," "Suspend," "Power On," "Reset," "Snapshot," etc.

SNAPSHOT IS YOUR FRIEND. Snapshot saves the state of your current virtual machine. So let's say you wanted to recompile the kernel. You take a snapshot of your current VM as it is now, and then try to recompile the kernel. Let's say you fail miserably and your machine is hosed and it won't even boot. You try several different hacks, but your VM simply laughs at you and refuses to come up. Hours of reading, research and effort are all for naught -- and your data is gone. Normally, its time to curse, throw things, kick the dog, kill the cat, cry, etc. However, since you hit that Snapshot button, you saved your machine at a known good state. The icon next to Snapshot is "Revert." If you hit that

Revert button, the VM magically restores itself to the point at which you took the Snapshot. USE SNAPSHOT!!!! The weekend you save may be your own.

OTHER IMPORTANT STUFF:

VMWARE TOOLS PROBLEM

If you load VMware Tools, it will create a new X11 config file for you. This is good if you want to have complete interaction between your guest operating system and your host operating system; but it is very bad when you want to boot from your CD/DVD/USB and your Xwindows hangs. If you like VMware Tools that much, then by all means use it. Just remember -- before you create the ISO image it may behoove you to go into your /etc/X11 directory and change the xorg.conf-backup file back into your xorg.conf file

VMWARE USB PROBLEM

This may not apply to you, but it does to me. My VM installed without native support between VMware and the USB drives. This is a problem because I want to be able to use USB on my VMware machines. Fortunately, this problem is easily handled.

HOW DO TELL IF I HAVE THIS PROBLEM?

- Bring up your virtual machine
- Insert a USB Device into your host operating system's USB port. Make sure the Host OS can see the USB device.
- On your VMware Console, select VM --> Removable Devices --> USB devices
- If your host OS can see the USB device, but your USB Devices menu is empty, you have this problem.

HOW DO I FIX THIS PROBLEM?

--Power Off the VM.

--On the host operating system, go to the directory that contains your VM image.

--Edit the .vmx file

-Between 'floppy0.filename' and 'Ethernet0.present' add the following lines:

usb.present = "TRUE"

usb.generic.autoconnect = "FALSE"

--Save the file.

--Restart the VM.

Now when you go to the VM --> Removable Devices --> USB Devices, you should see your USB device listed on the menu. Click the USB device, and BackTrack will autodetect the device.

Note: Autodetection of the USB device is not instantaneous; it may take up to 1.5 minutes for the VM to tell BackTrack that it has a USB device available. Be Patient.

Anyway, back to the fun.

FURTHER READING:

"BackTrack v2.0 -- Developer Notes for End Users" by the BackTrack Development Team

Perhaps most of it is OBE at this time, but some stuff is still useful.

BackTrack Wiki -- http://backtrack.offensive-security.com/index.php?title=Main_Page

INSTALLING NESSUS ON BACKTRACK

********The next section contains a modified version of Bodgan Dragomir's "How to install Nessus on BackTrack 2.0" found on professionalsecuritytesters.org. I thank him for his kind contribution.

Go to **www.nessus.org/download/**

Choose **Nessus 3.0.6 for Linux**. You will be required to fill out some paperwork, but that's fine. Tenable are good people ;-). Be sure to enter an authentic email address as Nessus will email you a registration number -- you will need this number later.

The version of Nessus you want is **Nessus-3.0.6-fc6.i386.rpm.**

Now go back to www.nessus.org/download/ and download **NessusClient 3.0.0 (a GUI for nessusd)**. Choose **NessusClient-3.0.0-fc6.i386.rpm**

Next, Google for and download **krb5-libs-1.6-6.i386.rpm; keyutils-libs-1.2-2.fc6.i386.rpm** and **openssl-0.9.8b-8.i386.rpm**

Copy the downloads into the /tmp directory, and change the .rpm extension to a .tgz extension.
bt~# cd /tmp
bt tmp# rpm2tgz *.rpm

Install the krb5 and openssl packages (if you begin to type the package name and then hit the TAB button, Linux will complete the name for you)

bt tmp # installpkg krb5-libs-1.6-6.i386.tgz
bt tmp # installpkg openssl-0.9.8b-8.i386.tgz
bt tmp # keyutils-libs-1.2-2.fc6.i386.tgz

Note: You may be wondering "How did he know to go and download those packages?" It's very

simple really: to run nessus you must first start the daemon. When I started the daemon, I got an error to the effect of "I can't start because I can't find 'something.c.ko.'" I therefore use my friend Google and search for 'something.c.ko.' I find an rpm package of some sort and install it. I then try to start the daemon again. Repeat until it starts without error or you get frustrated and give up.

Now, do the same for Nessus (don't do the client just yet).

bt tmp # installpkg Nessus-3.0.6-fc6.i386.tgz

Add an extra directory to your execution path:

bt tmp # ln -s /opt/nessus/sbin/nessusd /opt/nessus/bin/nessusd

Note: to perform other commands, nessusd must be in your path. However, for some reason the software doesn't install that way. This links nessusd from where it lives to where you need it to live. (FYI: you can link anything to anywhere. This is useful if you want to avoid copying files from one directory to another every time you update your software. The syntax is:
ln -s <where the file is> <where you want the file to be>)

--I'm sure there is a sexier way to do this, but... MLOC2!

Now copy the Nessus lib files over to the main operating system's lib files

bt tmp # cp -R /opt/nessus/lib/* /lib/

Now create the Nessus server certificate.

bt tmp # /opt/nessus/sbin/nessus-mkcert

Here's how I answered the questions:

CA Certificate life time in days [1460]: **[return]**

Server certificate life time in days [365]: **[return]**

Your country (two letter code) [FR]: **US**

Your state or province name [none]: **<your state>**

Your location (e.g. town) [Paris]: **<your town>**

Your organization [Nessus Users United]: **Gonk**

<bunch of output>

Press [ENTER] to exit

Next, start the nessus daemon

bt tmp # nessusd -D

Nessus will process some plugins and return you to a command prompt

Now we must create a user for Nessus. Since our end goal is to use a CD distribution version of BackTrack (and in my case other people are going to use it), I'll just use the standard root/toor login/password combination. Feel free to be more secure about things.

bt tmp # /opt/nessus/sbin/nessus-adduser

(Nessus will complain that it doesn't like to run in VMware. Ignore this for now)

Login : **root**

Authentication (pass/cert) [pass] : **pass**

-This is asking if you want to login using a password or a certificate. I told it to use a password.

Login password : **toor**

Login password (again) : **toor**

User rules

-I just hit **ctrl-D**, which is fine for my purposes. Feel free to be more secure.

Nessus will now show you the user you have created and ask if that's ok. Hit return to default to [yes].

Now we'll register our copy of Nessus and get the latest plugin set. Remember when you filled out paperwork before Nessus would let you download the server? You gave them an email address during that paperwork session. Well, they should have sent you an email by now. Within that email, you will find your registration number.

bt tmp # nessus-fetch --register <your registration number>

This may take a while, but you'll be all up to date when it's done.

Finally, install the Nessus Client

bt tmp # installpkg NessusClient-3.0.6-fc6.i386.tgz
bt tmp # NessusClient

The Nessus Client GUI should start.

OPTIONAL:

To make things easier on yourself in the future, it might be good to make an icon to launch Nessus. It might be better if we add this icon to the KDE Panel (aka the toolbar). Lets do that, shall we?

--**Right-click on the Kmenu and select 'Menu Editor.'**

-The menu editor window will appear

-- Choose File --> New Item

-Name your item Nessus; click 'OK'

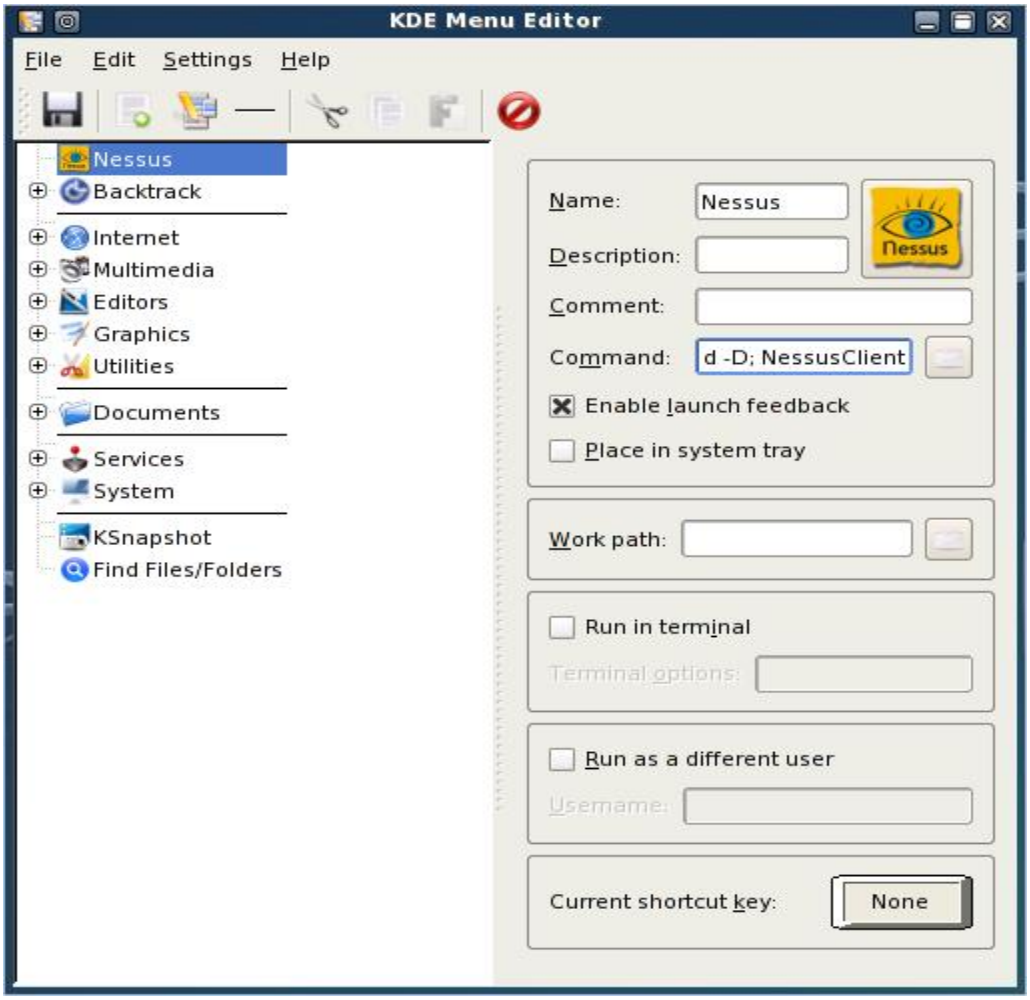
--Fill out the 'Description' and 'Comment' fields as desired.

-I do not desire to.

--In the 'Command' field, type: 'nessusd -D; NessusClient'

-This will complete the nessusd -D command (which starts the Nessus server daemon), and then launch the Nessus Client

--Next to the 'Name' and 'Description' field, there is a picture of a piece of paper. Click on that picture, and the "Select Icon" menu comes up. Under the "Icon Source" section, choose "Other Icons" and the NessusClient Icon is available. Feel free to choose your own icon from the list (or import your own Icon).



--Accept the defaults for all other settings, and select File --> Save.

Now run Nessus from the KDE panel using your new icon launcher.

Congratulations! Nessus has been successfully installed on your box. **BUT!!!!** Before you go have the celebratory tasty snack cake of your choice

---TAKE A SNAPSHOT!!!

INSTALL VMWARE PLAYER

I like BackTrack a lot because it has all the tools (well, almost all the tools) that anyone (well, almost anyone) could need to do a pen test. However, sometimes it would be nice to have a Windows box to do some Windows-specific stuff; or a Solaris Box to do some Solaris-specific stuff. Well, since large-capacity Flash Memory sticks are becoming affordable (I got a 4GB memory stick for $40 at Best Buy) it makes more sense to me to have a Windows VM on a memory stick and VMware Player on BackTrack. I discovered that it works like a charm (for me anyway).

A COUPLE OF THINGS OF NOTE:

--Loading the VM onto the OS and then creating the DVD did not work for me. My media was non-writable and VMware Player wants to write to a specific VM's log file when it boots that VM. I have not tried this configuration with writable media.
--Nor have I tried this on one Flash Memory stick (i.e. OS and VM on one Flash Stick). I might try doing that later.
--Booting a VM from a Flash Memory Stick takes quite a while to load. However, once it's up and running it runs like a champ.
--You cannot run a virtual machine from inside a virtual machine, so you'll have to wait until you create the ISO and boot from CD/DVD/Flash Drive to test functionality (trust me, it works).

First, go to the following website

http://www.vmware.com/download/player

VMware will ask you to fill out some paperwork as well, but we like VMware, so that's fine. Provide VMware with a valid email address as they will be emailing you your registration key. You'll need this key later.

Download VMware Player 2.0.2 (or latest version) for linux. Choose the **.tar** version and move it to your /tmp directory.

Now, unpack the VMware Player .tar.gz

bt ~ # cd /tmp

bt tmp # tar zxvf VMware-player-2.0.0-45731.i386.tar.gz

bt tmp # cd vmware-player-distrib

bt vmware-player-distrib # ./vmware-install.pl

VMware will prompt you with the following questions. Here's how I answered them:

In which directory do you want to install the binary files?
[/usr/bin] **[enter]**

What is the directory that contains the init directories
(rc0.d/ to rc6.d/)? **/etc/rc.d [enter]**

What is the directory that contains the init scripts?
[/etc/rc.d/init.d] **[enter]**

In which directory do you want to install the daemon files?
[/usr/sbin] **[enter]**

In which directory do you want to install the library files?
[/usr/lib/vmware]

The path "/usr/lib/vmware" does not exist currently. This program is going to create it, including needed parent directories. Is this what you want?
[yes] **[enter]**

In which directory do you want to install the documentation files?
[/usr/share/doc/vmware] **[enter]**

The path "/usr/share/doc/vmware" does not exist currently. This program is going to create it, including needed

parent directories. Is this what you want?
[yes] **[enter]**

Installation of VMware Player is complete <output truncated> if you want to uninstall the command is "/usr/bin/vmware-uninstall.pl"

Before running the VMware Player for the first time, you need to configure it by invoking the following command "/usr/bin/vmware-config.pl". Do you want the program to invoke the command for you now? [yes] **[enter]**

From this point on, I accept all defaults. If you wish to change anything, be my guest.

When installation is complete, run VMware-player from the command prompt.

bt vmware-player-distrib # cd [enter]

- simply typing "cd" and hitting [enter] automatically returns you to the current user's home directory.

bt ~ # vmplayer

D'oh! Error! Something about attempting to remove a filter function?

Yeah, that error was a pain to track down, but the high-level explanation is that the dbus-daemon is interfering with the execution of vmplayer.

So what do we do about that? We can stop the dbus-daemon, start vmplayer, and then restart the dbus-daemon. Probably not the optimal solution but MLOC2!

-- the optimal solution would be to find the problem code in the dbus-daemon, fix that code and recompile. However, I am not a kernel-level programmer so I'm going to have to let that one alone. :-)

dbus-daemon can be stopped by executing the '/etc/rc.d/rc.messagebus stop' command. It can be

started by executing the '/etc/rc.d/rc.messagebus start' command.

That's all well and good, but wouldn't it be better if there were something that could do that for us? Lucky for everyone, there is! It's called shell scripting; and it's almost too easy to do. Therefore, with some help from Google and a bit of trial and error, here's the script I came up with:

```
vi startvmplayer
#!/bin/sh
#
# This script will kick-off the VMplayer on this modified version of BackTrack
#
# description:    VMplayer is having a problem with dbus, so to effectively start
#                 VMPlayer, we must first stop the dbus-daemon; then kick-off
#                 VMPlayer, then restart the dbus-daemon.  If you're running it from
#                 a hard drive or within a VM, 'sleep 5' works well.  If you're running
#                 from a CD/DVD, 'sleep 8' is necessary.

#                 For some reason, I cannot successfully restart the dbus-daemon without
#                 invoking the 'rc.messagebus start' command twice.  I haven't been able to
#                 figure out why, but... MLOC2 applies.
#

/etc/rc.d/rc.messagebus stop
vmplayer &
sleep 5
/etc/rc.d/rc.messagebus start
/etc/rc.d/rc.messagebus start

#end script
```

Save this script to the /root directory (or any directory in your path, actually).

Now that we've constructed our vmplayer startup script -- MLOC2 though it may be -- let's execute it!

bt~# startvmplayer

VMware-Player should pop-up a license agreement. Accept it, and you should see the following:

Congratulations! You've installed VMware-Player. Now, let's make a launcher!

--Right-click on the Kmenu and select 'Menu Editor.'

-The menu editor window will appear

-- Choose File --> New Item

-Name your item VMware Player; click 'OK'

--Fill out the 'Description' and 'Comment' fields as desired.

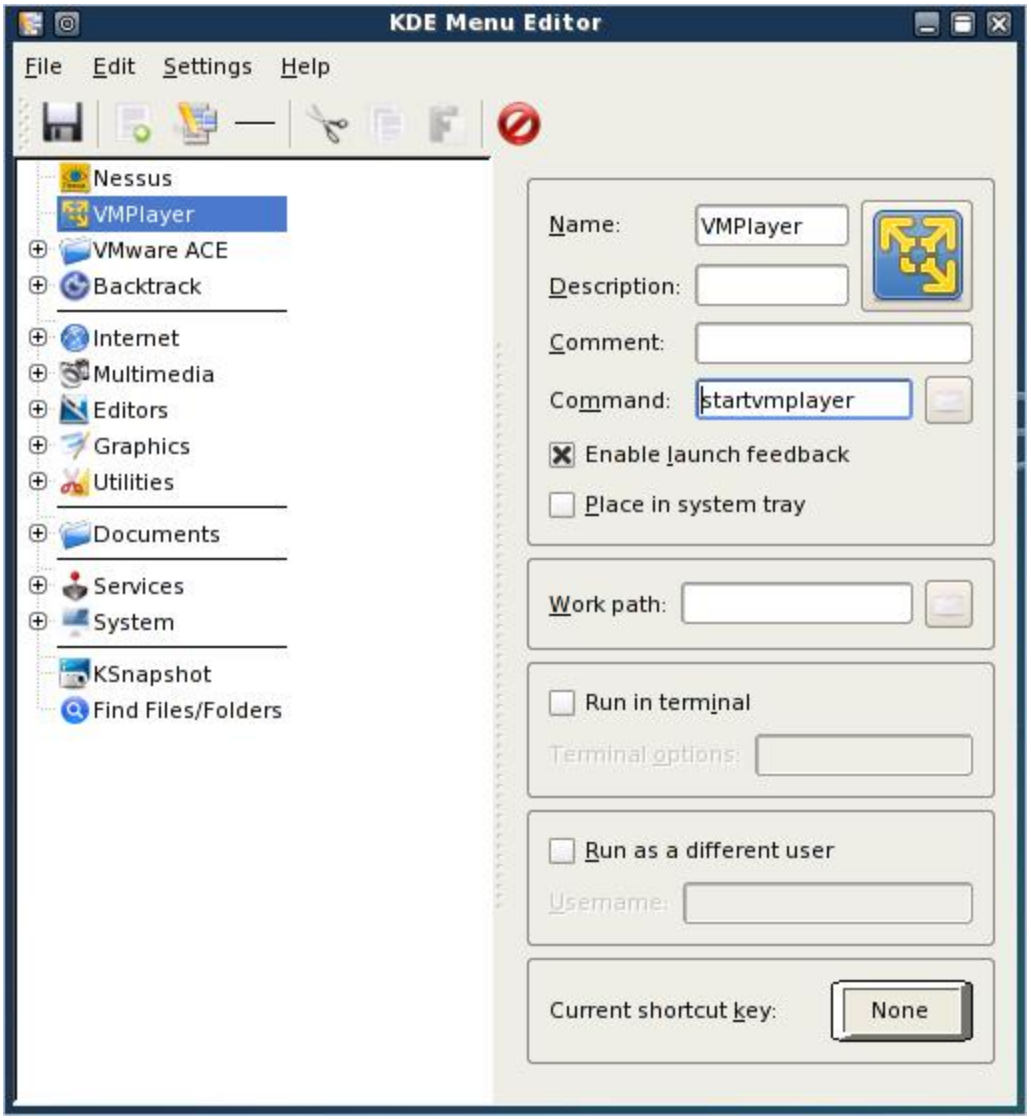
-I do not desire to.

--In the 'Command' field, type: 'startvmplayer'

-This will stop rc.messagebus, start vmware player, and then start rc.messagebus.

--Next to the 'Name' and 'Description' field, there is a picture of a piece of paper. Click on that picture, and choose your own icon from the list (or import your own and choose that).

-Under the "Icon Source" section, choose "Other Icons" and the VMware Player icon is available.



--Accept the defaults for all other settings, and select File --> Save.

Now run VMware Player from the KDE panel using your new icon launcher.

Congratulations! VMware Player has been successfully installed on your box. **BUT!!!!** Before you go have the celebratory venti double-shot latte...

---TAKE A SNAPSHOT!!!

INSTALLING TRUECRYPT

We all want to be safe and secure, don't we? Well, I do -- and I want you to be safe and secure as well; 'cause I love you. All emotional BS aside, let's install Truecrypt.

Actually... BackTrack 3.0 includes Truecrypt by default!

Kmenu --> BackTrack --> Miscellaneous --> Truecrypt

HOORAY!!!! Less work for me!

However, do you know how to use Truecrypt? If not, here's a brief tutorial:

BRIEF TRUECRYPT TUTORIAL:

So, if you've never used truecrypt before, you're probably asking "Now what?" I'm glad you asked. In short: truecrypt is awesome. It encrypts and decrypts on the fly; it can encrypt entire drives; it can create hidden partitions; it can create hidden partitions within encrypted drives; it is an incredibly powerful encryption tool. And you can't beat the price! So let's get you started with a few simple tasks:

Encrypt a USB Flash Drive:

Insert a USB Flash Drive into the host operating system. Make sure the host operating system can see the USB Flash Drive.

On your VMware Console, select VM --> Removable Devices --> USB Devices --> <your usb device>

bt ~ # truecrypt -c

Volume type:

1) Normal

2) Hidden

Select [1]: **1**

Enter file or device path for new volume: **/dev/sda1**

Note: To encrypt the USB drive, you have to enter a device name instead of the mount point. To determine the device name, enter the command "df -h" and note which /dev is associated with the mount point of your usb device.

WARNING: Data on device will be lost. Continue? [y/N]: **y**

Filesystem:

1) FAT

2) None

Select [1]: **1**

Hash algorythm:

1) RIPEMD-160

2) SHA-1

3) Whirlpool

Select [1]: **3**

Encryption algoritm:

1) AES

2) Blowfish

3) CAST5

4) Serpent

5) Tripple DES

6) Twofish

7) AES-Twofish

8) AES-Twofish-Serpent

9) Serpent-AES

10) Serpent-Twofish-AES

11) Twofish-Serpent

Select [1]: **1**

Enter password for new volume '/dev/sda1': **<your password>**

Re-enter password: **<your password>**

Enter keyfile path [none]: **[enter]**

TrueCrypt will now collect random data.

Is your mouse connected directly to computer where TrueCrypt is running? [Y/n]: **y**

Mouse data captured: **<move your mouse around until this value reaches 100%>**

TrueCrypt then creates the volume on the raw device.

MOUNT ENCRYPTED USB FLASH DRIVE:

Insert the Encrypted USB Flash drive into the host operating system's USB port. On the VMware Console, choose VM --> Removable Devices --> USB Devices --> <your usb drive>

Create a mount point:

bt ~ # mkdir /mnt/usb

Mount the encrypted drive (assumes physical device is located at /dev/sda1):

bt ~ # truecrypt /dev/sda1 /mnt/usb

Enter password for '/dev/sda1': **<your password>**

bt ~ #

Device is now mounted. You may transfer data into the device with normal commands (i.e. cp /home/file.txt /mnt/usb/)

UNMOUNT ENCRYPTED USB FLASH DRIVE:

Ensure your working directory is NOT the flash drive itself

bt ~ # truecrypt -d /mnt/usb

Drive is now unmounted.

More information on how to use TrueCrypt in Linux is available at **http://www.truecrypt.org/docs under the Linux Version hyperlink.**

CREATE AN ISO FROM THE OS:

Here's the meat of the project. Usually, if you're going to create a LiveCD from an operating system, you have to build and compile the aufs and squashfs (with lzma support) kernel modules from scratch. As much fun as it is to compile kernel modules from scratch and load them into the kernel, it is easier to screw things up and have to restore from backup, rebuild, or use VMware to Revert. HOWEVER, since BackTrack was built from a LiveCD distro, aufs and squashfs (with lzma support) kernel modules already exist! So we don't have to go through the pain of building them (and there was much rejoicing... <yea>).

Remember when we created our Virtual Disk of a size of 12GB? Well, we still have some empty space left on /dev/hda; so let's format the rest of that drive and mount it.

bt ~ # fdisk /dev/hda
<output truncated>

Command (m for help): **n [enter]**

Command action

e extended

p primary partition (1-4)

p [enter]

Selected partition 4

First cylinder (14771-26630, default 14771):**[enter]**

Using default value 14771

Last cylinder or +size or +sizeM or +sizeK (14771-26630, default 26630)**: [enter]**

Using default value 26630

Command (m for help): **w**

The partition table has been altered!

<output truncated>

The Kernel still uses the old table.

The new table will be used at next reboot

Syncing disks

bt ~ # init 6

Note: This will reboot the computer and thusly load the new partition table.

After BackTrack completes it's reboot:

bt ~ # mkfs.ext3 /dev/hda4

This formats the new partition using Linux filesystem ext3

<output omitted>

bt ~ # mkdir /mnt/hda4

This creates the mount point where the new partition will be mounted to

bt ~ # mount /dev/hda4 /mnt/hda4

This actually mounts the physical partition to the logical mount point.

Now we have a large amount of space in which to play. We're going to need quite a bit of it.

Next, go to **http://www.linux-live.org** and hit the **Download** icon

Grab the linux live 6.1.2 link, which will download the linux-live-6.1.2.tar.gz to your hard drive. Move the file to /mnt/hda4

bt hda4 # tar zxvf linux-live-6.1.5.tar.gz
bt hda4 # cd linux-live-6.1.5
bt linux-live-6.1.5 # vi .config

If you don't know how to use the vi editor, here's a great place to learn:
http://www.washington.edu/computing/unix/vi.html

The .config file controls various aspects of the build process. We must customize it to our needs.
Change **LIVECDNAME** to be whatever you want it to be.
Leave **RAM0SIZE** alone (or change at your own discretion).
Leave **KERNEL** alone
Edit **MKMOD** by

- removing **lib64** (unless you need it because you're on a 64-bit system, so then remove lib)
- removing **srv**
- adding **pentest**

Change **CDDATA='/tmp/live_data_$$'** to be **CDDATA='/mnt/hda4/live_data_$$'**
Accept all other defaults. and save the file.

Now, mount your physical boot partition to your boot directory:
bt linux-live-6.1.5 # mount /dev/hda1 /boot

Next, kick off a build:

bt linux-live-6.1.5 # ./build

-Accept all defaults (because you just edited the .config file) unless you have a good reason not to.

The build will take several HOURS, so now would be a good time to do something else.

RANDOM NOTE:

The build will create a folder called 'live_data_##' where ## is a randomly created random number. Once you burn your ISO image onto your CD/DVD/Flash Drive, you may discover certain aspects that you would like to change about your BackTrack. Before you kick off another build, I HIGHLY recommend you DELETE the previously created 'live_data_##' directory. Here's why: The funny thing about randomly created random numbers is that they are created randomly. Since they are created randomly, they are not in sequence, so one number being greater than another number provides no information relevant to the sequence in which those folders were created. This gets confusing very quickly and will bite you if you forget to do it. You will end up wasting time and CDs. HEED MY ADVICE.

When the build process is done, it will say:

----------done----------

* run /mnt/hda4/live_data_<randomNumber>/<YourBuildName>/make_iso.bat to create ISO image
* or copy content of /mnt/hda4/live_data_<randomNumber> to your USB device and run ./boot/bootinst.sh (from the device!) to setup boot sector

Now press Enter...

-Press Enter
-do **NOT** run make_iso.bat

Instead, perform the following:

bt linux-live-6.1.5 # cd /mnt/hda4/live_data_<randomNumber>/<YourBuildName>

bt <YourBuildName> # ./make_iso.sh

make_iso.sh works; make_iso.bat does not.

The script will prompt you for the name of your target image (which should be /mnt/hda4/<YourBuildName.iso). I recommend appending the current date to your build name, just so you can keep track of things (i.e. /mnt/hda4/<YourBuildName>_mmddyy.iso).

It will only take about 10 minutes to make the ISO image.
Once the ISO has been created, burn it!

I always transfer my ISO image to my host device first. I have SSH server enabled on my host device, and I simply type:

bt hda4 # scp <YourBuildName>.iso <user>@<hostIpAddress>:/home/<user>/.

OR... BackTrack came with a CD/DVD burner built in (but I haven't figured out how to use it in VMware)!

Go to **Kmenu --> Multimedia --> K3b**

Once the GUI comes up, go to **Tools --> Burn DVD ISO Image**

NOTE: You're about to burn a CD/DVD image from a .iso file (AKA an ISO image or ISO file). This is NOT the same thing as creating a data CD/DVD and making that CD/DVD bootable. The .iso image is a data representation of what a CD/DVD looks like when it is actually burned onto a CD/DVD. What you want to do is create a CD/DVD image from that data representation. If you create a data CD/DVD and make it bootable you will essentially be copying the .iso file (which is a data file) onto the

CD/DVD and then adding useless bootable "stuff" to that data CD/DVD. BIG DIFFERENCE. This has been known to bite a few people, so make SURE that your burning software supports the ability to create a CD/DVD from an ISO image, and then take advantage of that.

After the image gets burned; boot from it!

Congratulations! You have just learned how to modify BackTrack and create a Custom CD/DVD from it! Time to consume adult beverages in mass quantities! Don't drive drunk!

CONCLUSION

That's all folks! I hope this helped!

P.S. If you figure out something that would be helpful to the community at large, it might be cool if you contribute!