SANS GSEC Security Certification

Gary Wallin
Version 1.4b

# The Keep Within the Castle Walls – An Experiment in Home Network Intrusion Detection

## ABSTRACT

There are a number of security measures that can be implemented to protect a network. One of the key components that will assist in determining whether a system is being attacked is a network-based intrusion detection system (NIDS). A wonderful and free NIDS is snort. The GSEC course discusses how to set up snort on a Windows-based system. I will discuss how to set up snort 1.9.1 – the latest version – on a virtual Linux machine. First, the “before” scenario will describe the situation before this security improvement is enacted. Second, I will asses the risk, discuss why someone should consider network intrusion detection, talk about snort, VMware, and Linux, and investigate configuration options. I’ll conclude with some implementation notes, enhancements and the “after” scenario. The appendices provide brief installation instructions and resources for further information.

## ASSUMPTIONS AND NOTES

This paper makes the following assumptions about the reader

1. The reader is familiar with common PC, network, and security terms
2. The reader is familiar with Internet protocols and the OSI model
3. The reader is familiar with Windows 2000 and Linux (or a UNIX flavor)

In this paper, the words “hack” and “hacker” always refer to the “up-to-no-good, trying-to-break-into-your-system, blackhat hacker.” I apologize to all of the whitehats out there that are offended by the use of this term to mean someone attacking a system. I want to acknowledge in advance that not all hackers are bad.

This paper discusses using the latest version of Red Hat – 8.0 or “Psyche”. Psyche is freely available to download at RedHat’s website. Any discussion about other flavors of Linux and Unix are outside the scope of this document.

The version of VMware discussed is VMware workstation. VMware Workstation costs about $300 at the time of this writing. There are other products in the VMware line, GSX Server and ESX Server. In my opinion there would be no point in spending any more money for GSX or ESX because workstation does everything I need it to do and the cost differential is several orders of magnitude.

I'm going to discuss Linux, security and VMware in this paper. I would not consider myself an expert or guru on ANY of these topics. I will frequently refer to simplified implementations recognizing that there are more complex installations that could be appropriate. Hopefully simplifying scenarios, installations and configurations will help provide clarity without leaving out important details.

## BEFORE

The before setup of this case study is a simple home network connected to the Internet so that friends and family can access a modest web site to view pictures, share files and such. The web server runs Windows 2000 with IIS 5.0. The system is hardened by updating OS and application patches, reducing the number of services running, restricting anonymous account access, removing sample applications and implementing various other security best practices. The network is sitting behind a hardware-based firewall updated with the latest firmware. The firewall allows traffic on only a few ports. It does not respond to pings from the Internet. This helps deter some people from assaulting the system because many of the automated hacking tools ping a machine before attempting an attack or exploit.

However, there is currently nothing implemented that would assist in determining if the system was hacked. A system compromise is likely if someone is determined enough. According to most security professionals the only real way to secure a system is to fill it with concrete and bury it six feet in the ground. Even then someone can only say "maybe" it's secure. Currently, if someone breaks into the Windows 2000 server no one would know unless the perpetrator is careless or inept and leaves evidence of the attempt.

## DURING

### *Risk assessment*

To begin this discussion, why try to figure out if someone hacked into the system? Is there anything on this system worth protecting? Yes, there definitely is. Here are a few examples of what is worth protecting:

1. Personal information – People have personal information on their home computers such as passwords to personal web accounts, credit card information, e-mail, and other sensitive information. Quite often people can use this information to steal someone's identity. Check out http://www.consumer.gov/idtheft/ for information on identity theft.

2. Connection to corporate LAN – Because of Virtual Private Networks (VPN), people can connect from their home PCs to the corporate office. If a hacker is able to gain access on a personal computer VPN'd in to a corporate network, they have carte blanche access to that network and all of the perimeter security firewalls and routers on which the business spends money won't stop them. Imagine being the person responsible for the accidental indulgence of a trade secret or critical sales data. This is probably more of a risk now in the more sluggish economy when people are looking for that edge that will push their company over the top. Unscrupulous people may resort to dirty tricks to do whatever it takes to get ahead.

3. System could become a tool in an attack - Your computer could be 1) a jumping point to attack other sites and/or 2) a "zombie" in a denial of service attack. Frequently, hackers use systems that they've compromised to attack other sites in distributed denial of service (DDOS) attacks or other malicious activities. It is no secret that people will cover their tracks by using your system in a long string of compromised machines to make tracing the real hacker to his/her geographical location very difficult. Additionally, to execute a DDOS attack, thousands of machines are compromised and turned into "zombies" – machines under the compete control of the hacker – and launch abnormal packets at a target with the single intent of overwhelming it and taking it off of the network completely. Probably one of the most famous of these attacks was the classic GRC.COM – see http://grc.com/dos/grcdos.htm. Depending on the damages caused, it is possible that the victim could sue the owner of the compromised machine. At the very least it's a real mess to clean up especially if the computer has been rooted.

*Where is this network still vulnerable?*

The Windows 2000 server in this network is updated with the latest patches (as of the time of this writing) for the operating system and applications running, unused services are disabled, accounts use passwords that are complex and at least eight characters, account access is limited to "least privilege", auditing is turned on, and system management security best practices are, for the most part, followed. It also helps that there aren't any named users of the system so this network administrator doesn't have to worry about other people using weak passwords, people with too much access, people giving away their passwords to others and social engineering.

What else could be done to secure this network? Could anyone even know if the system was being attacked or was in fact successfully compromised? SANS teaches defense in depth – attack countermeasures should be established for protecting the information, application(s), host(s) and network(s) in a series of layers. So, the host system(s) and application(s) are fairly well secured, but there is nothing watching the network. Unless the perpetrator was either clumsy or ignorant enough to leave tell-tale evidence of their presence, no one would know they were there. One application that will alert about nefarious activity on a network is a Network Intrusion Detection System (NIDS).

The SANS course discusses how to work with snort for Windows but another option is to build the snort NIDS for Linux. This process isn't for the technologically shy – hence the assumptions about the reader at the beginning of the paper – but it's not overly difficult. Also very useful, is the concept of a virtual machine. There are multiple companies that work with virtual machines. Among them are VMware and Connectix. I decided to use VMware to create a virtual Linux machine to run the latest version of snort (1.9.1 at the time of this writing)

*So, why have a NIDS for a home machine?*

A NIDS makes sense for a corporation because they have the resources to spend on the hardware and software. They can support the staff required to set up, maintain, and monitor a NIDS but why bother with one at home? Here are the top five reasons

1. It provides a great deal of information about what is happening on a network and systems.

A NIDS is a good component to the defense in depth strategy. Something watching the network is a valuable tool to have in an anti-hacker arsenal if people are concerned about the risks listed in the section above.

2. Using snort is free; all that is required is the time for setup

The snort NIDS, thanks to the generosity of Marty Roesch and the rest of the snort team, is free and works well with very little effort. There are already precompiled snort binaries that can provide what is needed to get up and running. There is automatic rule management with another free add-on called oinkmaster that essentially will allow setup and configuration in a cron job with little maintenance. As new exploits are found, the snort rule writers include new rules that detect those exploits. With oinkmaster, the updates are downloaded automatically. I will discuss more about oinkmaster later.

3. *Noisy* people are found right away. If someone isn't careful and is trying to cause trouble it will probably show up on the IDS.

If someone is trying to get into the system and they're not sophisticated there's a good chance the IDS will detect them. Usually, assailants like to do reconnaissance on a network before attacking. When they're looking for known exploits or port scanning, the IDS can detect this activity and alert you to it before they start the attack. One caveat is that a particularly sophisticated individual will know how to avoid being detected by a NIDS, but it's still a key part of a security strategy and will help catch hackers.

I experienced this situation once at a company where a cache appliance was misconfigured and some people were using it as a proxy server to mask their IP addresses while surfing on the web for pornography. Until we had the IDS in place, no one had any idea this activity was going on. As soon as the IDS was in place and watching the network, it illuminated the problem. Pages and pages of alerts displayed on the console – we used DEMARC at the time – and we couldn't help but notice something was wrong. We stopped the system from being a proxy to the world shortly thereafter.

4. It's great for network cleanup!

From a security perspective configuration mistakes could lead to exploitable vulnerabilities which always lead to compromised systems, it's just a matter of time. So much goes on behind the scenes of even the smallest network. A NIDS will also help you figure out these configuration problems. If you want to know what's misconfigured in your network, install a NIDS at a major choke point, enable all of the rule sets, and watch the alerts pile up. We installed snort on our corporate network and within a day it registered a few thousand alerts. When we dug into the issues we had discovered routers were misconfigured, firewall ports weren't open that needed to be, firewall ports were open that should have been closed, etc. Now, you may think "that's really sloppy, you should be fired!" But, the fact is people make mistakes and no one has endless time to address every single issue on their network. If something points out the flaws with alerts and flashing icons the increased activity and the annoyance of having to sift through all of the alerts tends to cause people to prioritize fixing those problems.

Snort picked up on all kinds of other useful information about what was happening on the network – problems we could fix easily. We probably would have caught those issues running a sniffer on the network segment but a NIDS made it crystal clear. It was like a virtual neon sign pointing out the problems. On the home network, when I installed snort I saw alerts that directed me to investigate my DNS settings. I had forgotten that I had made some changes recently and my network was suffering because of it. I didn't know at the time because they weren't big problems that prevented me from doing the work I needed to do.

4. It has signatures for popular malware, viruses, and Trojan horses

If you have been attacked by any of the recent malware or ubiquitous worms such as the SQL Slammer, snort can help identify systems that have been affected. This is different than running a tool to detect which systems have the

SQL slammer virus and are currently affecting your network. Those tools only produce a snapshot – what is happening on the machines at this very moment. Snort, however, once the signature is enabled that looks for specific criteria about the virus/worm/malware, will alert on current *and* future instances

*Why a virtual machine?*

As an experiment, I decided to try and use a VMware virtual machine for building a simple NIDS. After installing, configuring, and using this virtual NIDS configuration for a while I became convinced that this setup could be useful for people for the following reasons:

1. The virtual machine disks can be run in undoable mode. From the VMware documentation "Undoable mode lets you decide when you power off the virtual machine whether you want to keep or discard the changes made since the virtual machine was powered on" This is an advantage for an intrusion detection system because if the machine ever becomes compromised or is overloaded by a denial of service, all that is required to restore it to a working, hack-free state is to power it off and on and then discard the changes. The system is instantly restored to the point from the last time changes were saved to the undoable disk. This is useful if I make a mistake or want to explore different configuration options.
2. There isn't any extra hardware to maintain and the amount of resources that the VM uses is minimal. Today's hardware has enough processing power for serving web pages and hosting a virtual machine. Additionally, with a home system behind a firewall, there isn't that much traffic involved so the load on the NIDS is light.
3. A VM has the same hardware configuration no matter what physical hardware the VM is on. In order to move a VM it is as simple as powering it off and moving the files associated with it to another physical piece of hardware. This would be very helpful if the hardware the VM is running on crashes and won't boot, but the hard drive is still intact. In this scenario all that you have to do is move the VM files to another system with VMware and power the VM back on. You the reader can duplicate my experience in configuring a VM and get exactly what I created because of this feature. In fact, if someone created an IDS virtual machine and made it available to others, people wouldn't have to do any work at all except download that VM and run it in their environment!

*Why Put Snort on Linux?*

1. Snort's native environment is UNIX since this is the platform where the developers choose to write the code. The snort port to windows is a good one but I like to work in the same environment that the developers of the software work in because I know they will have figured out the peculiarities

of that platform whereas they may not for a ported application – especially if there isn't any money involved in porting it

2. Linux uses very little system resources overall as a virtual machine because it is modular. There isn't as much operating system overhead as with a windows machine. You can pair down the number of services running to create a tiny running footprint.
3. If the IDS flags something, there are more free security, system and network-related UNIX tools available than Windows tools. There are thousands of open-source tools for UNIX flavors – check out freshmeat.net for 27,000+ programs

## Networking

The networking configuration discussed in this paper is simple. There are myriad modifications someone could make to the configuration to improve security. Virtual machines can connect to the host machine via a bridging protocol, NAT, or host only. When using the proprietary bridging protocol, the virtual machine can see everything on the network by putting the network adapter in promiscuous mode. When using the host-only type of connection, the only traffic the VM sees is from the host machine and vice versa. This provides some options:

*Option #1*

Configure a host machine with two network adapters. Adapter #1 could utilize the host-only type configuration and adapter #2 could utilize the bridging protocol. The second adapter could be connected to a hub that sits on the other side of the router before the web server. You could also go one step further and create a receive-only or "stealth" cable (see http://www.snort.org/docs/faq.html#3.1 for more information). With this stealth cable configuration, the VM is only accessible by the host machine.

*Option#2*

Configure the interface on the Linux VM without an IP address. This is the method I prefer to use. The steps to setup an interface without an IP address on RedHat 8 are:

In your favorite editor, create /etc/sysconfig/network-scripts/ifcfg-eth1(or whatever virtual interface you want to add) with the following information:

```
DEVICE=eth1
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
BOOTPROTO=static
PEERDNS=no
```

To activate the interface type "service network restart"
In this scenario, the VM will not be accessible from anywhere but the host machine. You will need to have your machine service DNS requests if you configure the Linux VM this way and want to access the Internet from it.

Something to note about both of these setups is that because the virtual machine network interface is running in a *receive-only* mode OR without an IP address it would be very difficult for someone who didn't have physical access to the console to hack that system. This *keep within the castle walls,* as far as I can tell, would only be accessible from the host console because with the Workstation version the virtual *console* is the VMware application. VMware runs in a user's session and is only accessible from the real console of the host machine. If you're REALLY paranoid you could also consider putting sensitive information on the virtual machine for this reason. Two caveats are 1) the infiltrator could crash the virtual machine to render it useless; 2) the attacker could exploit remote control software on the host machine if it exists or he/she could load something that gives access to the host console.

*Installing VMware*

The VMware installation is essentially a default installation. This paper isn't going to include screenshots of each of the installation steps because VMware has documentation that can do a better job than I can. Here are a few notes about the install process:

1. When prompted, choose to disable autorun.
2. When prompted do not search for virtual disk files unless you know you have them.
3. Make sure that at least one of the virtual machine's networking connections is installed as "bridged".

*Configuring Linux Virtual Machine using RedHat 8.0*

Note: You will need to stick to text mode (init 3) until the VMware tools are installed. When the tools are installed, they configure a video driver and set X Windows settings appropriately. If you want to use the virtual machine as just an IDS, there really isn't any reason to use the GUI mode anyways so you can leave the init level at 3 (full multi-user mode). There are many other better papers on configuring and securing Linux

The ideal IDS configuration is outside of the scope of this paper but some thoughts on the Linux configuration are:

1. Use disk druid to configure the partitions. There are better ways of configuring a system than what the automatic partitioning scheme provides. Here is a suggested configuration:

    a. /boot = 50MB (must be within first 1024 cylinders)
    b. / = 500 MB
    c. /tmp = 500 MB
    d. /usr = 500 MB
    e. /var = 1 GB – this is where snort is going to write all of the alerts
    f. swap space (based on RedHat recommendations)
2. Use sudo.
    a. Check to make sure the sudo package is installed – rpm –q sudo
    b. If not, it's located on the first disk of RedHat 8.0 (Psyche) at /RedHat/RPMS/sudo-1.6.6-1.i386.rpm
    c. Install with rpm –Uvh sudo-1.6.6-1.i386.rpm
    d. Edit /etc/sudoers – visudo
    e. Locate the section that says "user privilege specification"
    f. Add "<your username> ALL=(ALL) ALL" to give your account the ability to run UNIX programs as root. This is safer than logging in as root because the actions are all recorded in the syslog file
3. Do all of the standard hardening practices. Check out the security-related HOW-TOs at the Linux Documentation Project
    a. Disable unused accounts
    b. Stop and disable unused services(daemons)
    c. Enable logging
    d. Make sure the system is patched

*Installing VMware Tools*

If you want to run GUI mode on Linux you should install the VMware Tools. You can find the instructions in the VMware help under the title "Installing VMware Tools in a Linux Guest Operating System".

*Saving the Changes*

Now that the virtual machine is configured and the OS is installed, you might want to make a backup of your virtual machine. This is as simple as copying the files in the virtual machine folder to another location. At this point you also might want to switch the disk mode to "undoable" mode. This will give you the ability to either save the changes you make to the system when configuring snort or to discard them if you make a mistake and start over without having to worry about figuring out if and how to reverse all changes – a feature that saves time.

*Configuring snort*

1. Check to see if libpcap is installed by typing "rpm –q libpcap".
2. If not, locate the following library on the RedHat CDs – libpcap 0.6-2-16 or later
3. Install by typing "rpm –Uvh libpcap 0.6-2.16.rpm"

4. There are two options for snort installation. I suggest using the RPMs because they not only provide a compiled binary but they install a snort daemon in /etc/init.d which you can set up to start automatically when the machine is powered on. Another option is to compile the source yourself. To do this, download the snort binaries from www.snort.org and follow the instructions in the INSTALL file to configure, compile, and install the binary. It is really not much more than typing "./configure", then "make", then "make install". The automatic configuration script makes this relatively painless. Snort writes to the /var/log/snort directory to a file called "alert" and this option is configured in the snortd script added by the RPM install. That's the configuration we're going to use to keep this simple. Other configuration options are discussed briefly later in this paper.
5. Edit the snort configuration file at /etc/snort.conf if you installed with RPMs
    a. Change the HOME_NET variable to match your network and subnet mask.
       ```
       var HOME_NET 192.168.1.0/24
       ```
    b. Change the EXTERNAL_NET variable from any (which I think causes a LOT of false alerts) to !$HOME_NET.
       ```
       var EXTERNAL_NET !$HOME_NET
       ```
    c. Change the RULE_PATH to /etc/snort
       ```
       var RULE_PATH /etc/snort
       ```
6. If you installed using the RPMs then there is a snortd script in /etc/init.d that you can use chkconfig to add to the services that start automatically. That set of commands are
    a. Edit the INTERFACE variable in the /etc/snortd script to the correct interface e.g. eth0, eth1, etc.
    b. chkconfig –add snortd
    c. chkconfig –levels 2345 snortd on

   Now when you power on the virtual machine the snort daemon will start automatically.
7. Download the oinkmaster tar file from www.snort.org. This script will allow signatures to be updated automatically if you schedule a cron job. This is where you will gain the benefit of automatic signature updates. This team is FAST at coming up with a new snort rule that detects the latest bug, exploit, or hacker tool. RedHat installs tar, gzip, and wget so you should have everything you need to support oinkmaster.
8. Once oinkmaster is installed create a cron job to run daily.
    a. In the /etc/cron.daily directory create a file called oinkmaster.cron that contains the following information and save it
    ```
    #!/usr/bin/bash
    oinkmaster.pl –C /etc/oinkmaster.conf –c –o /etc/snort
    ```
    b. Make sure it is executable by typing chmod 755 oinkmaster.cron

*Running Snort*

This configuration will cause snort to write intrusion detection events to the file /var/log/snort/alert. Review those logs on a regular basis. If you want you can configure snort to write to the system log instead of the alerts log. To do that, change the snortd script to include the –s switch or edit the snort.conf file. If you want to use windows you can get a windows syslog daemon such as the awesome Kiwi Tools' syslog daemon for Windows (get it for free at http://www.kiwitools.com/ ). Tell snort to send the alerts to the syslog and then configure the syslog to forward to the IP address of the machine that is running the Windows syslog daemon. Just make sure you put the syslog daemon on a server the VM can access. The line in snort.conf for sending output to the syslog looks like this. . .

```
#output alert_syslog: LOG_AUTH LOG_ALERT
```

*Starting Snort*

Type "service snortd start"

You should see something similar to the following in the syslog

```
<timestamp> localhost snort: Initializing Output Plugins!
<timestamp> localhost kernel: eth1: Promiscuous mode enabled.
<timestamp> localhost kernel: device eth1 entered promiscuous mode
<timestamp> localhost snort: WARNING: OpenPcap() device eth1 network
lookup:  ^Ieth1: no IPv4 address assigned
<timestamp> localhost snort: Initializing daemon mode
<timestamp> localhost snortd: snort startup succeeded
<timestamp> localhost snort: PID path stat checked out ok, PID path set to
/var/run/
<timestamp> localhost snort: Writing PID "6856" to file "/var/run//snort_eth1.pid"
<timestamp> localhost snort: http_decode arguments:
<timestamp> localhost snort:     Unicode decoding
<timestamp> localhost snort:     IIS alternate Unicode decoding
<timestamp> localhost snort:     IIS double encoding vuln
<timestamp> localhost snort:     Flip backslash to slash
<timestamp> localhost snort:     Include additional whitespace separators
<timestamp> localhost snort:     Ports to decode http on: 80
<timestamp> localhost snort: rpc_decode arguments:
<timestamp> localhost snort:     Ports to decode RPC on: 111 32771
<timestamp> localhost snort: telnet_decode arguments:
<timestamp> localhost snort:     Ports to decode telnet on: 21 23 25 119
<timestamp> localhost snort: Conversation Config:
<timestamp> localhost snort:    KeepStats: 0
<timestamp> localhost snort:    Conv Count: 32000
```

```
<timestamp> localhost snort:    Timeout   : 60
<timestamp> localhost snort:    Alert Odd?: 0
<timestamp> localhost snort:    Allowed IP Protocols:
<timestamp> localhost snort:  All
<timestamp> localhost snort:
<timestamp> localhost snort: Portscan2 config:
<timestamp> localhost snort:     log: /var/log/snort/scan.log
<timestamp> localhost snort:     scanners_max: 3200
<timestamp> localhost snort:     targets_max: 5000
<timestamp> localhost snort:     target_limit: 5
<timestamp> localhost snort:     port_limit: 20
<timestamp> localhost snort:     timeout: 60
<timestamp> localhost snort: WARNING: command line overrides rules file alert
plugin!
<timestamp> localhost snort: Snort initialization completed successfully, Snort
running
```

Note that it complains that there isn't an IP address on the Ethernet interface that you want to watch on. Ignore that error, this is what you want. The other warning about overriding rules file alert plugin is because you specified what you want snort to do on the command line in the snortd script

Congratulations, you're up and running! Now as long as you review the logs you will know more about what's happening on the system.

*Other Options*

At some point, viewing the alerts in the syslog files could become tedious, so there are a number of other tools that someone can use to augment their snort implementation. The installation of each of these tools is beyond the scope of this document. I'll mention a couple that I think are particularly useful and you can check out the snort website for many more.

ACID – The Analysis Console for Intrusion Databases (ACID) is a very powerful reporting tool that does graphs and allows searches on just about everything you can imagine. It works with mysql, an open source database that has become very popular.  ACID is included in the snort distribution under the contrib. directory. ACID homepage is at http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html

Sguil – The Snort GUI for Lamerz (Sguil) is a snort front-end based on Tcl/Tk. Sguil can be found at http://www.satexas.com/~bamf/sguil/. It allows someone to manage the alerts as they occur.

**AFTER**

Obviously there is a great deal more than can be done to secure the whole network but this next step will help identify when someone is attempting to do reconnaissance, someone is attacking a system, or someone has successfully compromised the system.

I review the logs regularly. In fact quite often I open an SSH session to the virtual machine and have it tail the alerts log, that way I can see what snort finds interesting. People try and attack web servers all of the time. Additionally, the Code Red and other IIS worms are relentlessly trying to break in to the server.

I had two major concerns about this experiment when I started

1. Would the virtual machine consume so much memory and CPU that I wouldn't be able to do much else?
2. Would the virtual network adapter run in promiscuous mode and if it did, would it be able to capture attacks?

*Memory and CPU Consumption*

The virtual machine uses about 50% of the available CPU (1.8GHz) when it's booting and about 5% CPU when it's scanning the network. The memory usage varies from 5MB to 60MB of the total 512MB on the system.  Both of these measures are acceptable. I don't notice a performance loss when running the NIDS VM.

*Virtual Network Adapter*

First, the virtual network adapter will run in promiscuous mode. Second, so far the network adapter hasn't had any problems capturing the packets. I ran a few hacking tools I found on the Internet against the network and snort alerted on each of them.

It would be foolhardy to assume that snort will catch everything so a next step for the security conscious could be to include a host-based IDS. Each individual can determine their own level of risk exposure and implement an appropriate solution.

## References

Danyliw, Roman. “Analysis Console for Intrusion Databases (ACID).” URL: http://www.andrew.cmu.edu/~rdanyliw/snort/snortacid.html (March 19, 2003).

“Desktop Products -- VMware Workstation 3.2.” URL: http://www.vmware.com/products/desktop/ws_features.html (March 15, 2003).

“freshmeat.net: Welcome to freshmeat.net.” URL:http://freshmeat.net (March 20, 2003).

Gibson, Steve. “The Attacks on GRC.COM.” Gibson Research Corporation. March 5, 2002. URL: http://grc.com/dos/grcdos.htm (March 5, 2003).

“Identity Theft.” February 21, 2003. URL:http://www.consumer.gov/idtheft/ (March 8, 2003).

O’stling, Andreas. “The Oinkmaster Snort rules updater.” URL:http://nitzer.dhs.org/oinkmaster/ (March 19, 2003).

“Redhat.com | RedHat Linux 8.0.” URL:http://www.redhat.com/software/linux/ (March 21, 2003).

“Single List of HOWTOs.” URL: http://www.redhat.com/mirrors/LDP/HOWTO/HOWTO-INDEX/howtos.html (March 21, 2003).

“Snort.org.” URL:http://www.snort.org/dl/ (March 3, 2003).

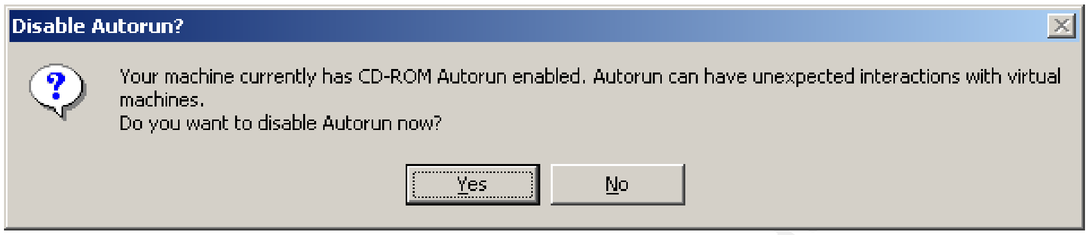
“Syslog Daemon for Windows, Free Syslog Server, Firewall logging, Kiwi Syslog Daemon.” URL:http://www.kiwisyslog.com/products.htm#syslog (February 15, 2003).

Visscher, Bamm. “Snort GUI for Lamerz [Sguil].” URL: http://www.satexas.com/~bamf/sguil/ (March 19, 2003).

VMware Workstation Help. VMware, Inc. 1998-2002.

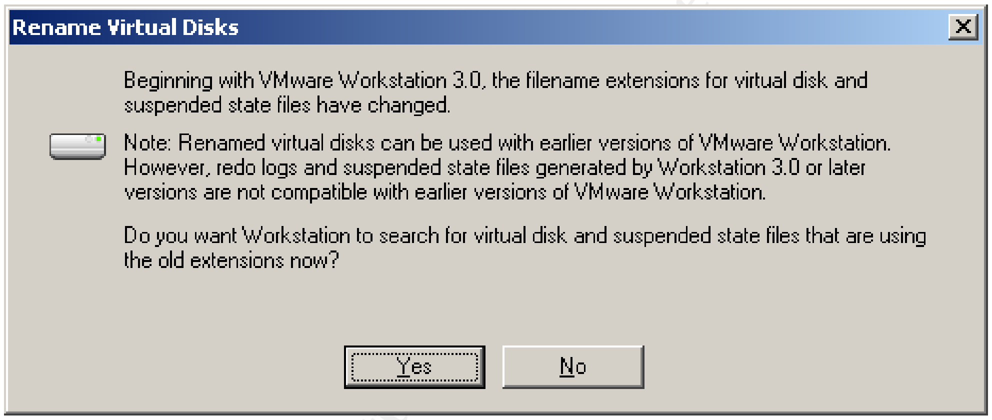
## Appendix A – VMware Installation

1. Start the installation
2. Agree to the license agreement
3. Choose a folder to install the software to
4. Click Install
5. When prompted to disable autorun choose yes



Your machine currently has CD-ROM Autorun enabled. Autorun can have unexpected interactions with virtual machines.
Do you want to disable Autorun now?

6. Unless you know you have virtual disks choose "No" to searching for virtual disks. It's just a waste of time



Beginning with VMware Workstation 3.0, the filename extensions for virtual disk and suspended state files have changed.

Note: Renamed virtual disks can be used with earlier versions of VMware Workstation. However, redo logs and suspended state files generated by Workstation 3.0 or later versions are not compatible with earlier versions of VMware Workstation.

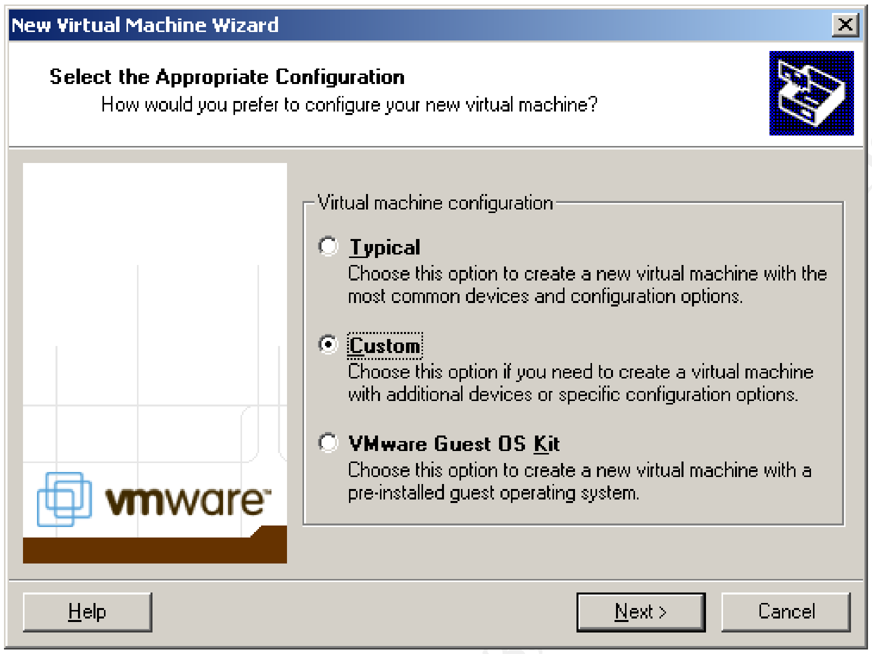
Do you want Workstation to search for virtual disk and suspended state files that are using the old extensions now?

7. Restart the system after installation has completed

## Appendix B – Create A Linux Virtual Machine

The following screen shots show the steps to create a Linux virtual machine

1. Choose a custom configuration



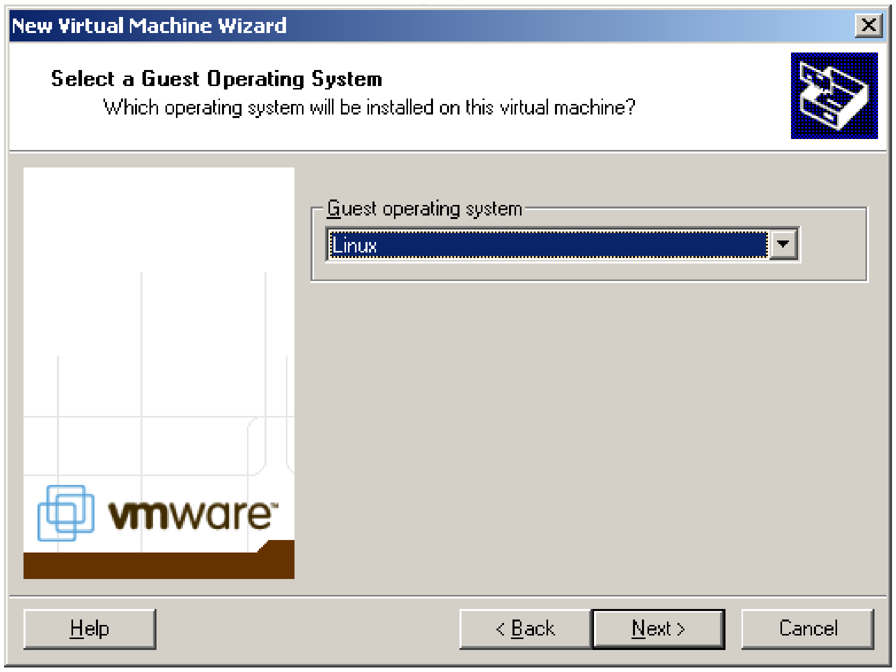
New Virtual Machine Wizard

**Select the Appropriate Configuration**
How would you prefer to configure your new virtual machine?

Virtual machine configuration

- ○ **Typical** Choose this option to create a new virtual machine with the most common devices and configuration options.
- ● **Custom** Choose this option if you need to create a virtual machine with additional devices or specific configuration options.
- ○ **VMware Guest OS Kit** Choose this option to create a new virtual machine with a pre-installed guest operating system.

2. Choose Linux as the Guest Operating System

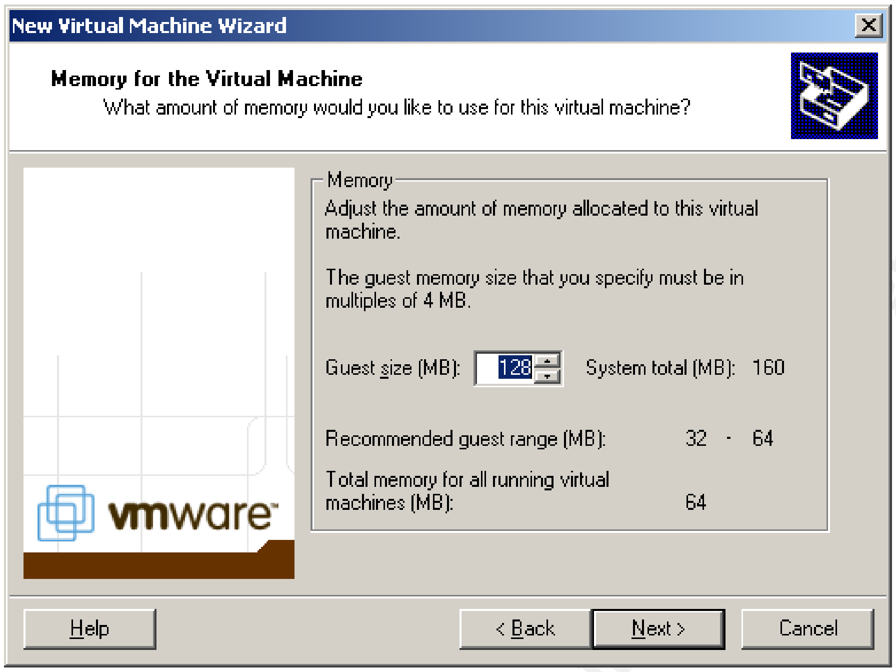


New Virtual Machine Wizard

**Select a Guest Operating System**
Which operating system will be installed on this virtual machine?

Guest operating system: Linux

3. Choose a memory size that works for your system. 128MB is probably a decent size for most systems


New Virtual Machine Wizard

**Memory for the Virtual Machine**
What amount of memory would you like to use for this virtual machine?

Memory
Adjust the amount of memory allocated to this virtual machine.

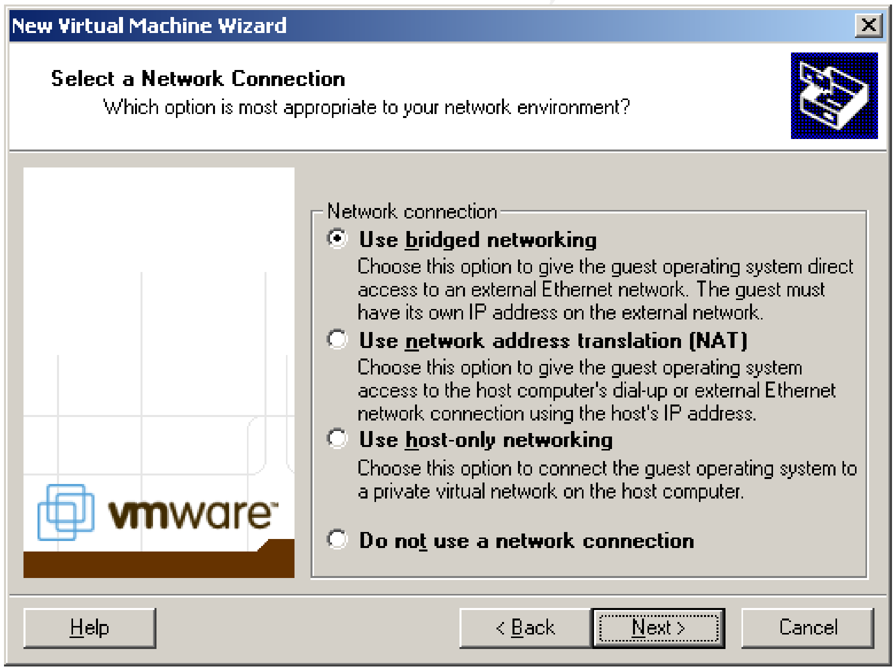
The guest memory size that you specify must be in multiples of 4 MB.

Guest size (MB): 128 System total (MB): 160

Recommended guest range (MB): 32 - 64

Total memory for all running virtual machines (MB): 64

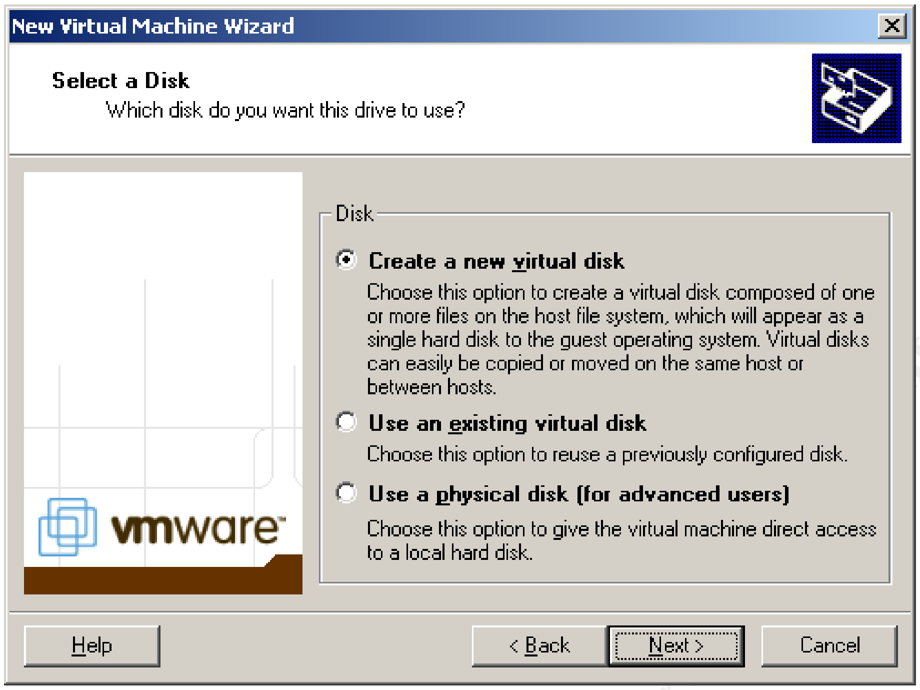
4. Choose "Use bridged networking"


New Virtual Machine Wizard

**Select a Network Connection**
Which option is most appropriate to your network environment?

Network connection
[✓ **Use bridged networking**]
Choose this option to give the guest operating system direct access to an external Ethernet network. The guest must have its own IP address on the external network.
**Use network address translation (NAT)**
Choose this option to give the guest operating system access to the host computer's dial-up or external Ethernet network connection using the host's IP address.
**Use host-only networking**
Choose this option to connect the guest operating system to a private virtual network on the host computer.
**Do not use a network connection**

5. Choose "Create a new virtual disk"



New Virtual Machine Wizard

**Select a Disk**
Which disk do you want this drive to use?

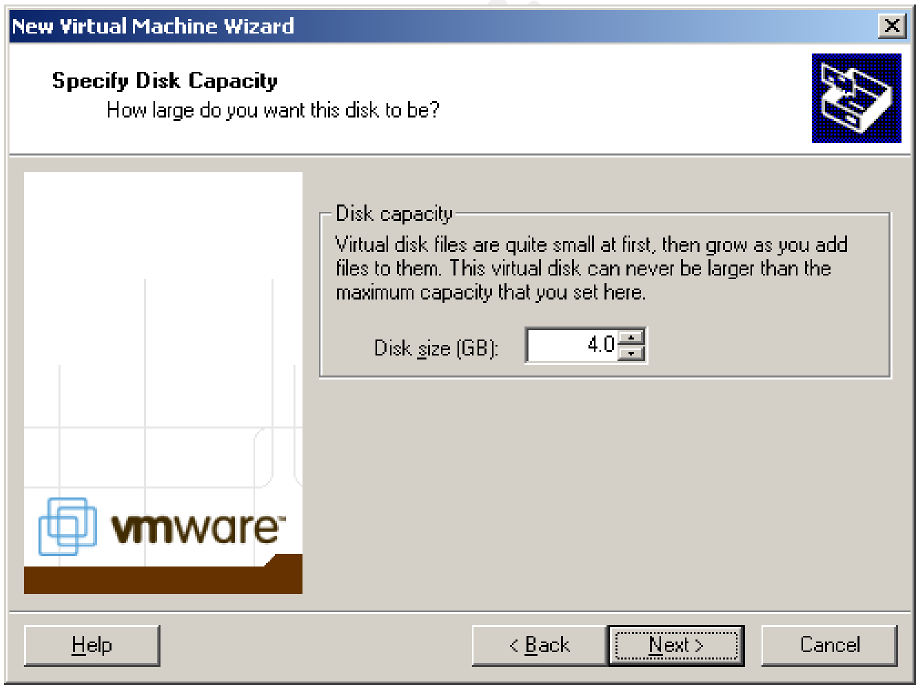
Disk

(●) **Create a new virtual disk**
Choose this option to create a virtual disk composed of one or more files on the host file system, which will appear as a single hard disk to the guest operating system. Virtual disks can easily be copied or moved on the same host or between hosts.

( ) **Use an existing virtual disk**
Choose this option to reuse a previously configured disk.

( ) **Use a physical disk (for advanced users)**
Choose this option to give the virtual machine direct access to a local hard disk.

6. Set the capacity to be at least a few GB. It's OK the virtual disk files don't start out taking that muc space, they grow as they are filled. The maximum limit though is what is defined here. The virtual disk will NEVER be able to grow larger than this



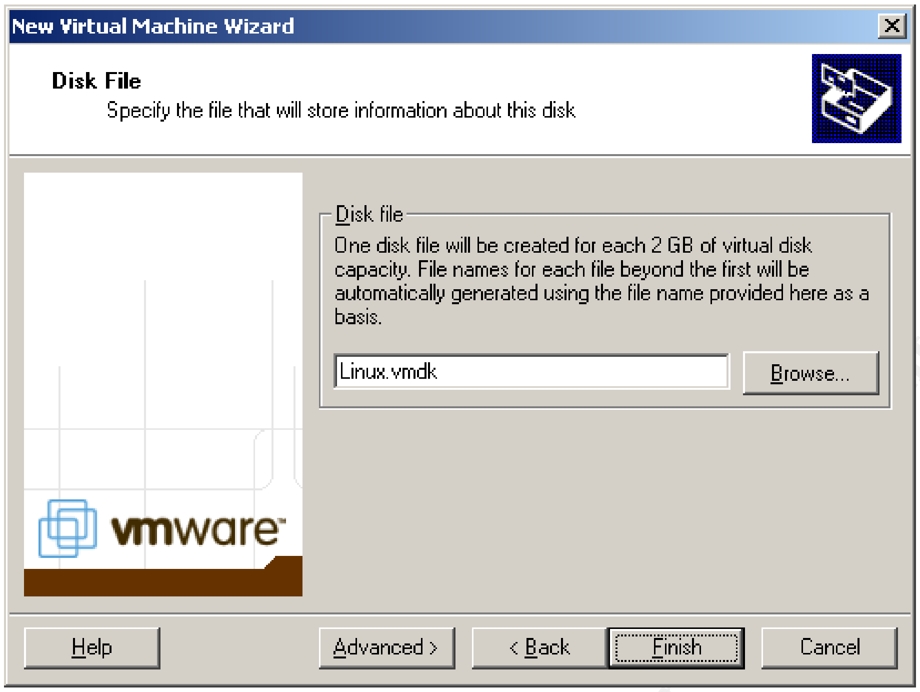
New Virtual Machine Wizard

**Specify Disk Capacity**
How large do you want this disk to be?

Disk capacity
Virtual disk files are quite small at first, then grow as you add files to them. This virtual disk can never be larger than the maximum capacity that you set here.

Disk size (GB): 4.0

7. Give the disk file a name. Using the default name that the installer provides is just fine


New Virtual Machine Wizard

**Disk File**

Specify the file that will store information about this disk

Disk file

One disk file will be created for each 2 GB of virtual disk capacity. File names for each file beyond the first will be automatically generated using the file name provided here as a basis.

Linux.vmdk