

AC 2009-318: INTEGRATING VIRTUALIZATION TECHNOLOGY INTO REMOTE LABS: A THREE-YEAR EXPERIENCE

Peng Li, East Carolina University

Integrating Virtualization Technology into Remote Lab: A Three-Year Experience

1. Introduction and Background

Many colleges and universities are facing rising enrollments while the budgets have not been increased proportionally. In our program, enrollment has gone up significantly in the past few years, especially in the distance education (DE) section. How to use the limited resources to deliver quality education effectively and efficiently becomes a real challenge. To deal with this challenge, we have experimented with different virtualization options in the past three years. This paper is a follow-up to the studies^{1,2} we presented at the previous ASEE conferences and summarizes our three-year experience.

Virtualization technology allows multiple guest virtual machines to run simultaneously on a physical computer. The technology was first developed³ at IBM for mainframe systems in the 1960s. The first x86 virtualization product, “VMware Virtual Platform” was released by VMware Inc. in 1999⁴. The year of 2009 is the 10th anniversary of this breakthrough. The x86 virtualization technology has gradually become more mature and stable since early 2000s⁵. Virtual machines were reportedly used for educational purposes in college computer labs as early as 2002^{6,7,8}. However, the technology has not been adopted broadly until recent years due to a few factors: 1) the early virtualization products were buggy and not very reliable; 2) the virtualization software packages were resource-demanding and costly. Therefore they were usually hosted on high-end machines in centralized on-campus computer labs.

Members from our faculty have been using VMware since 2002. In 2006, VMware Server was released free of charge for personal use. Microsoft released Virtual Server R2 for free download. Then we decided to migrate from physical computers to virtual machines in selected labs. The cost of creating and maintaining a virtual lab was much lower than that of building and maintaining a physical lab. It was easier to deploy new projects in virtual environments. We believed that virtualization technology had become mature enough and it would help us deliver certain laboratory courses efficiently and effectively.

2. Virtualization Software Selection

Currently there are many different virtualization software packages available, notably VMware Workstation, Server, and Fusion (for Mac), Microsoft Virtual PC, Virtual Server and Hyper-V, Sun xVM VirtualBox, Parallels Workstation and Desktop (for Mac), QEMU and Xen.

QEMU and Xen are open source projects and are released under GNU General Public License (GPL). However, they are not as user friendly as some other virtualization applications. Xen can only run under Linux/UNIX systems with modified kernels.

Microsoft Virtual Server and Virtual PC are free for personal use. They support Microsoft Windows guest operating systems well but have inadequate support for other guest operating systems such as Linux and FreeBSD.

VMware Player and VMware Server are also free for personal use. VMware player can be used to run a single virtual machine (VM) but cannot be used to create virtual machines. VMware Server can be used to create and run virtual machines; nevertheless it is only officially supported under Windows Server and Linux, not under Windows XP or Vista. Faculty and students can use VMware Workstation and some other VMware products free of charge with certain limitations if the department is a member of the VMware Academic Program (VMAP).

Sun xVM VirtualBox is free for personal and educational use. The license is less restrictive than that of VMware. An open-source edition of VirtualBox is available under GNU General Public License.

Out of all virtualization products we tested, VMware and VirtualBox are better choices for us. Both VMware and VirtualBox are user friendly and run under Microsoft Windows - the operating system used by most of our students. Mac users can run the virtual machines using VMware Fusion or VirtualBox for Mac OS X. As shown in Table 1⁹, both VMware and VirtualBox support a variety of guest operating systems, including Windows, Linux, FreeBSD and Solaris, which is appealing for instructors who plan to build a virtual lab with diversified platforms.

VMware, with its 10-year history, is a proven product that is stable and reliable. VirtualBox is a new comer in the virtualization world and is supposedly buggier. We had minor issues with both VMware and VirtualBox but generally they performed well in most labs.

VirtualBox is more lightweight than VMware. The size of the current version (2.1.2) of the VirtualBox installation file is 36 MB (32-bit version) or 40 MB (64 bit version) while the size of the VMware Workstation 6.5.1 installation file is 500 MB (32/64 bit version).

Table 1. VirtualBox vs. VMware⁹

Feature	VirtualBox	VMware Server / Workstation/ Fusion
Supported host operating systems	Windows 2000, XP, 2003, Vista, Linux, Mac OS X, Solaris 10U5, OpenSolaris, FreeBSD (under development)	Windows 2000, XP, 2003, Vista, Linux (32bit and 64bit), Mac OS X
Supported guest operating systems	DOS, Windows 3.1, 95, 98, NT, 2000, XP, Vista, Linux, OpenBSD, FreeBSD, OS/2, Solaris, OpenSolaris, others	DOS, Windows 3.1, 95, 98, NT, 2000, XP, Vista, Linux, FreeBSD, Solaris
64bit host OS support	yes	yes
64bit guest OS support	no (planned)	yes
License costs	Free	Workstation and Fusion free for VMware Academic Program members, Server free for end users, not redistributable

For centralized remote labs hosted on campus, VMware is a better choice because VMware Infrastructure provides a reliable solution with centralized management capabilities. VirtualBox has a smaller footprint and runs faster, especially on relatively low-end computers with less RAM. So VirtualBox is probably a better choice for decentralized virtual labs hosted on students' personal computers. By default, VirtualBox does not install virtual network adapters on the host. The virtual machine can still access Internet through Network Address Translation (NAT) but the internal network is isolated from the host.

Our selection of VMware and VirtualBox does not imply that any other virtualization software package is inferior. For example, Xen is a well-known high-performance hypervisor using paravirtualization technology. Xen is suitable for server virtualization but is difficult to set up on personal computers. We consider VMware and VirtualBox as good options for educational use.

3. The Three Year Experience (2006-2008)

Virtual machines were deployed in Intrusion Detection Technologies, a 3-credit hour undergraduate course (2-hour lecture and 2-hour lab). The course was offered in fall 2006, fall 2007 and fall 2008. A decentralized virtual lab approach was implemented: The pre-built virtual machines were downloaded to and installed on students' personal computers. Students performed the hands-on labs on the virtual machines on their own computers. No centralized on-campus server was involved except that students could download the lab files from a class ftp server. If some students did not have broadband Internet access, the lab files could be distributed to them on a CD-R/DVD-R.

Table 2: Virtualization software used in the three semesters

	Fall 2006	Fall 2007	Fall 2008
Virtualization Software	VMware Server 1.0.x or VMware Player 1.x	VMware Workstation 6.0.x	VirtualBox 1.6.6 and VCL
Minimal hardware requirement	Pentium III CPU, 512MB RAM, 10GB free space	Pentium IV CPU, 512 MB RAM, 10 GB free space	Pentium IV CPU, 768 MB RAM, 10 GB free space
Pre-built VM1	Fedora Core 5	CentOS 5.0	CentOS 5.2
Pre-built VM2	None	Debian 3.1	Debian 4.0
Total size of all lab files	~ 910 MB	~960 MB	578 MB

As shown in Table 2, in fall 2006, some students used VMware Player and other students used VMware Server to run the virtual machines. Only one pre-built virtual machine (VM) was provided due to the concern that some students' computers might not be powerful enough to run more than one VM. The single VM acted as the server or the target, on which the students set up intrusion detection systems such as Snort and Bro. The host computer acted as the client or the attacker, on which students ran Nmap to scan the virtual machine or ran Wireshark to sniff the traffic on the virtual network. In 2006, virtualization was still a relatively new concept to many students. However, our students adapted well and the initial trial was a success¹.

In fall 2007, a second virtual machine was added in selected projects. VMware Workstation was used to run the virtual machines. The two VMs formed a private virtual network. VM1 acted as the server or the target and VM2 acted as the client or the attacker. Labs requiring multiple hosts could be carried out more easily. For example, we added a lab in which a Snort sensor was installed on VM2 and the central BASE console was installed on VM1. The communications between the sensor and the console were encrypted using stunnel. The student feedback about the virtual lab remained positive. However, many students also complained that VMware took up too much resources on their computers in terms of CPU and memory².

To make the virtual lab more portable and easier to run, VMware Workstation was replaced by more lightweight VirtualBox in fall 2008. On the server VM1, a stripped down version of YUM repository and a local ftp repository of lab files (including network traces and source codes of open source security tools) were pre-installed. As demonstrated in Table 2, the total size of all lab files was reduced to 578 MB. If a student did not have broadband Internet access, the lab files could be delivered on a CD-R to her/him by mail. Then the student could install the VMs on her/his personal computer and complete all labs.

Meanwhile, we tried out two bonus lab projects with Virtual Computing Lab (VCL)¹⁰. VCL was a remote access service and image management system developed at North Carolina State University. Through the VCL provisioning front-end, blade servers could be dynamically installed on demand with pre-built operating system images. The user could reserve a particular image, connect to and use the blade server running the image. After the reserved session ends, all changes made to the image by the user would be lost. The blade server would be reloaded with a fresh image. Thus computing resources could be shared more efficiently. Although the resources (mostly blade servers) behind the web front-end can be located at different places, generally speaking VCL is a centrally managed service.

We created a VCL image with Windows XP as the host operating system. On the image, VirtualBox, 7-Zip, SSH Secure Shell client, Xming X Server, UltraVNC Server and the two prebuilt virtual machines were installed. The students reserved the image and connected to a VCL computer through remote desktop to complete the labs. If needed, the instructor could connect to the VCL computer using VNC to provide real-time assistance. Students could connect to the VCL computer from a hotel, from an airport, from a public library and from almost anywhere with high-speed Internet access to do the hands-on exercises. No additional software was needed since the remote desktop client program was already available under Windows XP and Windows Vista.

4. Evaluation

4.1 Student Performance

During the semester the course was offered, usually a lab manual was distributed with an answer sheet each week. The lab instructions were detailed considering 1) the hands-on labs were performed in a decentralized environment without the direct presence of the instructor; 2) some students had little Linux/UNIX experience. (We have since adjusted our curriculum and listed Linux as a prerequisite). In addition to the instructions, the lab manual contained two types of

questions: the observation questions and the discussion questions. To answer the observation questions, the student needed to perform certain tasks and record the output. To answer the discussion questions, the student needed to use her/his own words to compare topics or explain results. After the hands-on lab was finished, the student submitted the completed answer sheet with supporting materials (log files, network traces etc.) electronically to Digital Dropbox on Blackboard.

Twelve required weekly labs and multiple optional bonus projects were assigned each semester. The score averages of all required labs were listed in Table 3. The full score of a lab was 100. Bonus projects were not included when the averages were calculated because they were different each year. Table 3 shows that the student performance has been quite consistent. There is no clear correlation between the lab score average and the virtualization software used by the students. In other words, VMware is as effective as VirtualBox for virtual labs.

Table 3. The score averages of all required labs

	Fall 2006 VMware	Fall 2007 VMware	Fall 2008 VirtualBox
Lab score average	89.97	90.85	90.93

4.2 Student Feedback

Near the end of each semester, an anonymous, discretionary survey was conducted online. In fall 2006, 24 out of 40 students (18 face-to-face students and 22 DE students) participated in the survey. In fall 2007, 24 out of 67 students (24 face-to-face students and 43 DE students) replied to the survey. 48 out of 61 students (20 face-to-face students and 41 DE students) responded in fall 2008.

In fall 2006 and fall 2007, VMware was used to run the virtual machines. In fall 2008, VirtualBox was used instead. Students were asked what they liked about using VMware/VirtualBox. The results shown in Table 4 and additional comments from students revealed that most of them were positive about using VMware/VirtualBox to do the labs.

Table 4. Responses to the questions “What do you like about using VMware in the lab?” (2006, 2007) or “What do you like about using VirtualBox in the lab?” (2008)

	Fall 2006 VMware	Fall 2007 VMware	Fall 2008 VirtualBox
I could study at my own pace.	63%	55%	70%
I could learn Linux and other new stuff.	79%	96%	80%
The OS could be easily installed and reinstalled.	79%	77%	80%
I could experiment with the security tools inside a virtual network.	79%	82%	80%
Other	21%	5%	4%

Students were also asked what they did not like about using VMware/VirtualBox. The results were listed in Table 5. In fall 2006, only one VM was used. The next year, a second VM was added. It was not surprising that the percentage of students complaining that VMware used too much CPU/memory/space rose from 38% in 2006 to 50% in 2007. To deal with this issue, VMware was replaced by VirtualBox in fall 2008 to run the virtual machines. The percentage of students complaining about the problem dropped to 22%. The user experience indicated that VirtualBox was less resource-demanding. It was also worth noting that 57% of student had no complaint about VirtualBox in fall 2008 and 32% had no complaint about VMware in fall 2007.

Table 5. Responses to the questions “What don’t you like about using VMware in the lab?” (2006, 2007) or “What don’t you like about using VirtualBox in the lab?” (2008)

	Fall 2006 VMware	Fall 2007 VMware	Fall 2008 VirtualBox
It is difficult to use.	0%	5%	0%
It is time-consuming.	4%	9%	13%
It is too slow.	25%	5%	9%
It uses too much CPU/memory/space.	38%	50%	22%
Other	8%	9%	9%
None		32%	57%

In the fall 2008 survey, we also asked students which approach they preferred to do the hands-on labs. The results are shown in Table 6. According to a pre-class survey, 91% of students had used VMware and 14% of students had used VirtualBox before taking this course. At the end of the semester, 67.4% of students preferred to use VirtualBox and 21.7% preferred to use VMware to perform the labs on their own computers. A student pointed out that she/he liked VirtualBox but preferred VMware because VMware was more widely recognized and used in industry. Only 2.2% preferred to do the labs in a traditional physical computer lab and 6.5% preferred to use a centralized remote lab such as VCL to do the hands-on exercises.

Table 6. Responses to the question “The hands-on labs can be performed in various ways. Which way do you prefer as a student?” (2008)

	Percent Answered
Perform the labs on physical machines in a computer lab.	2.2%
Perform the labs using VirtualBox on my personal computer	67.4%
Perform the labs using VMware on my personal computer	21.7%
Perform the labs in a centralized remote lab (e.g. VCL) through Remote Desktop or VNC	6.5%
Unanswered	2.2%

The survey results were certainly affected by the facts that 1) VirtualBox was the only virtualization software package we used in fall 2008; 2) most labs were performed on students' own computers; 3) VCL was only required in two bonus lab projects so many students did not have enough exposure to it.

However, at least the survey results show that the decentralized virtual lab approach can be a viable choice for laboratory courses. A few students mentioned that their computers were powerful enough to run multiple virtual machines and they were glad that they did not need to depend on a centralized system which might not be available all the time.

Students were asked to leave comments and suggestions on the virtualization software they used and on the course overall. Most comments were positive. For instance, a student commented: "I have more VMWare experience, but for the purposes of this class, I found VirtualBox to be easier to use and more user friendly".

We also received suggestions and took corrective actions accordingly. In 2006, some students indicated that they had little experience with Linux. To deal with this problem, the instructor added review sessions on how to use Linux. The first few lab manuals were designed to be very detailed and step by step instructions were provided. The manuals gradually became less detailed and more challenging. The students were expected to use skills learned from previous labs. We recognized that that knowing Linux/UNIX was essential for future information technology professionals. A Linux course was added as prerequisite for this lab.

A few students commented that the virtual machine was slow. So we replaced VMware with VirtualBox in 2008 and customized the virtual machines further to reduce their footprints. Some students suggested more Microsoft Windows-related labs be added. We are considering it.

5. Conclusion

The last three year's experience shows that the decentralized virtual lab has made it possible for students to study anytime, anywhere and at their own pace. The student performance has been consistent. The student feedback has been positive. Using VMware/VirtualBox virtual machines not only provides a cost-effective way to offer remote hands-on laboratory courses but also exposes the students to virtualization technology, which is being adopted extensively in industry. Many students indicated that they intended to continue exploring virtual machine technology after finishing the course.

For educational purposes, VMware and VirtualBox are both viable choices with which a virtual lab can be built. VMware is reliable while VirtualBox is lightweight. They can be deployed with a centralized remote access service such as Virtual Computing Lab (VCL). They can be used on students' personal computers in a decentralized fashion. It is unlikely that the decentralized virtual lab approach can replace the physical computer lab approach or the centralized remote lab approach entirely because many devices have not been or cannot be virtualized. Instead, the decentralized virtual labs and the centralized remote labs can be complementary. Virtualization technology can be implemented in the centralized remote labs to reduce equipment and maintenance costs and to improve efficiency. Centralized remote labs are also more suitable if

some equipment cannot be virtualized or if the lab software has restrictive licenses. The decentralized virtual lab can provide 24x7 availability because a student does not need to share any equipment with others. Virtualization technology has helped improve and will continue helping enhance distance learning through remote labs.

Acknowledgments

This project is part of the ongoing effort to integrate virtualization technology into our curriculum. I would like to extend thanks to Dr. Tijjani Mohammed, Dr. Phil Lunsford, Dr. Chengcheng Li, and Mr. Lee Toderick for their help and support.

Bibliography

1. P. Li, P. Lunsford, T. Mohammed, L. Toderick, and C. Li, "Using Virtual Machine Technology in an Undergraduate Intrusion Detection Lab", Proceedings of 2007 ASEE Annual Conference and Exposition, Honolulu, Hawaii, USA, June, 2007.
2. P. Li, T. Mohammed, L. Toderick, P. Lunsford, and C. Li, "A Portable Virtual Networking Lab for IT Security Instruction", Proceedings of 2008 ASEE Annual Conference and Exposition, Pittsburgh, PA, USA, June, 2008.
3. R. J. Creasy, "The Origin of the VM/370 Time-Sharing System. IBM Journal of Research and Development", 25, 5, 483-490 (1981)
4. http://en.wikipedia.org/wiki/X86_virtualization, retrieved February 3, 2009
5. M. Rosenblum, "The Reincarnation of Virtual Machines", ACM Queue. 2, 5 (July/August 2004).
6. M. Stockman, "Creating remotely accessible virtual networks on a single PC to teach computer networking and operating systems", Proceedings of the 4th Conference on information Technology Curriculum, Lafayette, Indiana, USA, 2003.
7. Y. Nakagawa, H. Suda, M. Ukigai, Y. Miida, "An innovative hands-on laboratory for teaching a networking course", Proceedings of the 33 rd ASEE/IEEE Frontiers in Education Conference, 14-20, Boulder, CO, USA. November 5-8, 2003.
8. G. Steffen G, "Teaching Local Area Networking in a Secure Virtual Environment", Proceedings of 2004 ASEE Annual Conference and Exposition, Salt Lake City, UT, USA, June 2004.
9. http://www.virtualbox.org/wiki/VBox_vs_Others, retrieved February 3, 2009
10. S. Averitt, M. Bugaev, A. Peeler, H. Shaffer, E. Sills, S. Stein, J. Thompson and M. Vouk, "Virtual Computing Laboratory (VCL)", Proceedings of the International Conference on the Virtual Computing Initiative, Research Triangle Park, North Carolina, USA, May 2007.

Appendix A: A Sample Lab Manual

ICTN 4201 Lab 6

Installing and Using Network-based Intrusion Detection System Snort

Requirements:

Please read the lecture notes before doing the lab. To complete the lab report, place answers in Lab 6 Answer Sheet.doc. The length of the answers is not restricted. The final submission should include the completed answer sheet and the required files *BinCapture.[timestamp]* and *alert* in one 7z file. Please submit the 7z file by 11:59 pm Wednesday October 8 to the Digital Dropbox on Blackboard. Before uploading, please rename the 7z file to “LastNameFirstNameLab6.7z”. For example, John Smith in should use the name “SmithJohnLab6.7z”. Keep a backup copy of your report. The submission received after the due date will get 20% deduction in grade each day. If you have a real emergency and need an extension, please inform the instructor at least 24 hours before the due date. I would suggest everyone finish and submit the report at least 24 hours before the due date. You should complete the report independently. The requirements for each lab could be different, please examine them carefully. If you have any question, please contact the instructor as early as possible.

Objectives:

After completion of this lab, you should be able to

1. install Snort from source
2. use Snort as a packet sniffer/logger
3. use Snort as a NIDS logging to the local host

Procedures:

A. Installing Snort from Source Code

1. We need to use fresh virtual machines for this lab. Before starting VirtualBox, delete the .VirtualBox folder and reinstall it from VMS.7z. Power on the virtual machines Sunflower and Camellia. Log onto Sunflower as **student** using the SSH client program.
2. In the SSH terminal, use wget to download the snort source code from **<http://www.snort.org/dl/old/snort-2.6.1.5.tar.gz>**
(Note: if the link is broken, use the backup link **<ftp://192.168.128.128/pub/bak/snort-2.6.1.5.tar.gz>**)
3. Unpack the source code by typing
tar zxvf snort-2.6.1.5.tar.gz
4. Go to the source code directory by typing
cd snort-2.6.1.5

5. Check all the options available for configuration by typing
./configure --help | more
6. Hit spacebar to scroll down.
7. By default, the program will be installed in what directory? Answer: (___A1___).
8. What feature is enabled by "--enable-flexresp2"? Answer: (___A2___)
9. What feature is enabled by "--enable-react"? Answer: (___A3___)
10. What feature is enabled by "--with-mysql"? Answer: (___A4___)
11. After reviewing the configuration options, run the configure script by typing
./configure --with-mysql --enable-dynamicplugin
(Note: One of the reasons we build Snort from source is that the prepackaged Snort rpm is not available in the CentOS 5 base repository.)
12. The configuration process will stop with an error "Libpcap library/headers not found".
13. To solve the dependency, the libpcap headers must be installed. Switch to the superuser mode by typing *su* (without Dash so you will remain in the current directory). Use yum to install libpcap-devel by typing
yum install libpcap-devel
14. Type *y* for any question.
15. Use the rpm command to find out the description of the package **libpcap-devel**. Record here (___A5___).
16. Exit from the superuser mode by typing *exit*.
17. Back to the source code directory /home/student/snort-2.6.1.5, run the configure script again by typing
./configure --with-mysql --enable-dynamicplugin
18. The configuration process will stop with an error "Libpcre header not found".
19. To solve the dependency, the Libpcre headers must be installed. Switch to the superuser mode by typing *su*. Use yum to install pcre-devel by typing
yum install pcre-devel
20. Type *y* for any question.
21. Use the rpm command to find out the description of the package **pcre**. Record here (___A6___).

22. Leave the superuser mode by typing *exit*.
23. Back to the source code directory `/home/student/snort-2.6.1.5`, run the configure script again by typing
./configure --with-mysql --enable-dynamicplugin
24. The configuration process will stop with an error “unable to find mysql headers (mysql.h)”. To solve the dependency, the MySQL header files must be installed. Switch to the superuser mode by typing *su*. Use yum to install mysql-devel by typing
yum install mysql-devel
25. Type *y* for any question.
26. Exit from the superuser mode by typing *exit*.
27. Back to the source code directory `/home/student/snort-2.6.1.5`, run the configure script again by typing
./configure --with-mysql --enable-dynamicplugin
28. The configure step finishes with no error, makefiles are created. Start compiling the source code by typing *make*
29. After the compilation process finishes with no error, switch to the superuser mode by typing *su*
(Note: No dash here. If *su* with dash, you will be switched to the `/root` directory; without dash, you remain in the current source code directory. All steps below should be done in the root mode).
30. Start the final installation by typing *make install*
31. Now the Snort program is installed in `/usr/local`. The executable *snort* is copied to `/usr/local/bin`.
32. Create the directory to store the Snort configuration file by typing
mkdir /etc/snort
33. Create the Snort log directory by typing
mkdir /var/log/snort
34. In the directory `/home/student/snort-2.6.1.5`, use *wget* to download the Snort rules `ftp://192.168.128.128/pub/bak/snortrules-snapshot-2.6.tar.gz`
35. Unpack the rules by typing
tar zxvf snortrules-snapshot-2.6.tar.gz

36. Copy the rule directory to `/etc/snort` by typing
`cp -R rules /etc/snort`
 (Note: Please remember Linux is case-sensitive.)
37. Go to the etc directory by typing
`cd /home/student/snort-2.6.1.5/etc`
38. Copy files `snort.conf`, `reference.config`, `classification.config`, `sid-msg.map` and `unicode.map` separately to `/etc/snort` by typing
`cp *.* /etc/snort`
39. Edit `snort.conf` by typing
`nano /etc/snort/snort.conf`
 (Note: Nano is a free clone of the Pico editor. In Nano, to search a string (e.g. `RULE_PATH`), press **Ctrl-W** and enter the string; to exit from Nano, press **Ctrl-X**, Nano will ask if you want to save the modified buffer, type **y** if you want to save the changes. A tutorial for the Pico editor can be found here:
<http://www.helpdesk.umd.edu/documents/4/4795/>)
40. In `snort.conf`, change the variable **`RULE_PATH`** from `../rules` to `/etc/snort/rules`

```

127.0.0.1 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
GNU nano 1.3.12 File: /etc/snort/snort.conf

# mail this for compatability with versions of snort that support port lists
portvar MAIL_PORTS [25,143,465,691]

# SSL Ports
portvar SSL_PORTS [25,443,465,636,993,995]

# Path to your rules files (this can be a relative path)
# Note for Windows users: You are advised to make this an absolute path,
# such as: c:\snort\rules
var RULE_PATH /etc/snort/rules

# Configure the snort decoder
# =====
#
# Snort's decoder will alert on lots of things such as header
# truncation or options of unusual length or infrequently used tcp options
#
# Stop generic decode events:

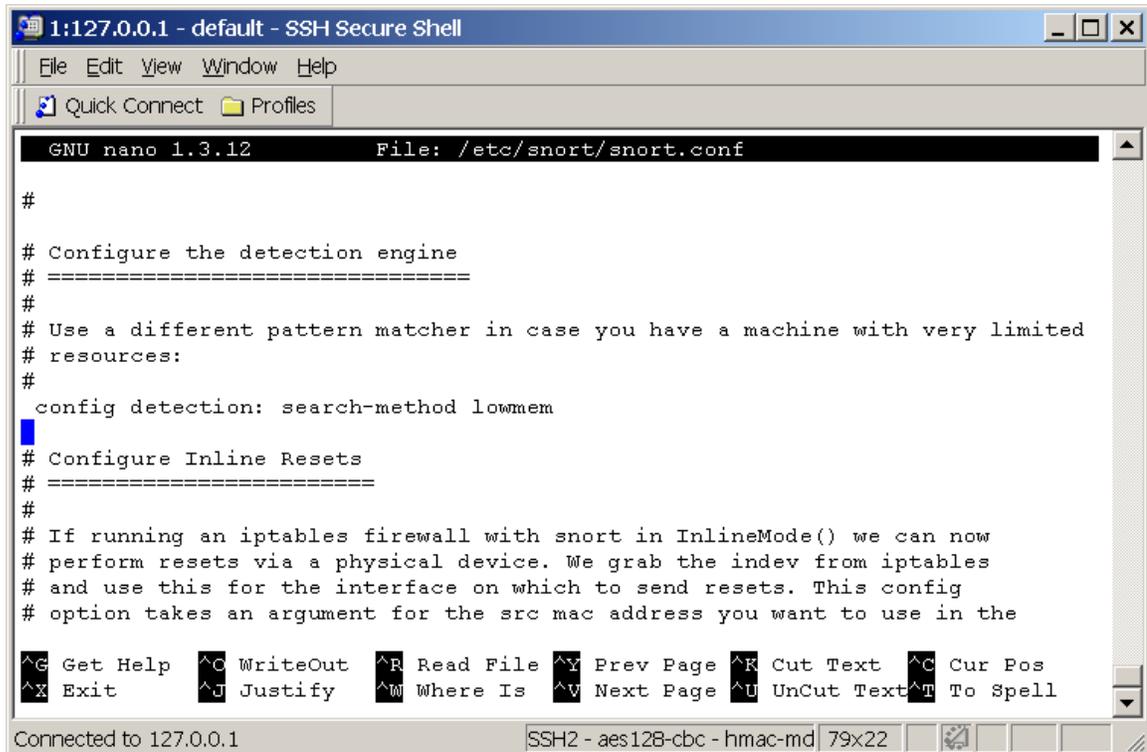
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^M Where Is ^V Next Page ^U UnCut Text ^T To Spell
Connected to 127.0.0.1 SSH2 - aes128-cbc - hmac-md 80x25

```

41. Snort, especially the recent version, requires a lot of memory to run. Because our virtual machine has limited memory, we should change the search method in `snort.conf` to `lowmem` in order to reduce the potential memory usage. Find and uncomment the line

config detection: search-method lowmem

(Note: To find the string *lowmem*, press Ctrl+W and enter the string. To uncomment the line, remove the number sign(s) at the beginning of the line.)



```
1:127.0.0.1 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
GNU nano 1.3.12 File: /etc/snort/snort.conf
#
# Configure the detection engine
# =====
#
# Use a different pattern matcher in case you have a machine with very limited
# resources:
#
config detection: search-method lowmem
#
# Configure Inline Resets
# =====
#
# If running an iptables firewall with snort in InlineMode() we can now
# perform resets via a physical device. We grab the indev from iptables
# and use this for the interface on which to send resets. This config
# option takes an argument for the src mac address you want to use in the
#
^G Get Help ^C WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
Connected to 127.0.0.1 SSH2 - aes128-cbc - hmac-md 79x22
```

42. Save the modified snort.conf file. Snort should be ready for use now.

B. Using Snort as a Packet Sniffer

1. In the SSH terminal, go to the student home directory by typing *cd /home/student*
2. Run Snort as a packet sniffer by typing */usr/local/bin/snort -dev*
3. Captured packets will be displayed through the SSH terminal. Press **Ctrl-C** to stop sniffing, a summary of the packets will be shown at the end of the output.
4. Run Snort as a packet sniffer again by typing */usr/local/bin/snort -dev | more*
5. Press the spacebar a few times, select a packet randomly and examine it. Copy the content of the packet here (___B1___). Record the source IP address of the packet (___B2___). Record the destination IP address of the packet (___B3___). Record the source port of the packet (___B4___). Record the destination port of the packet (___B5___). What is the application layer protocol of this packet? Answer (___B6___). (Hint: Think about what service is running on the source or destination port.)

6. Press **Ctrl-C** to stop sniffing.
7. Check the manual about what the switches -d, -e, -v and -L mean by typing
man snort
8. Press spacebar to scroll down. Record what “-d” enables (___B7___). Record what “-e” enables (___B8___). Record what “-v” enables (___B9___). Record what “-b” enables (___B10___). Record what “-L” enables (___B11___). (Note: Please remember Linux is case-sensitive.)
9. Type *q* to exit from man page.

C. Using Snort as a Packet Logger

1. In the SSH terminal, capture and log packets in binary format by typing
/usr/local/bin/snort -b -L /home/student/BinCapture
(Note: If the log file is not specified, Snort will create binary logs in /var/log/snort with the name snort.log.[timestamp].)
2. Wait no shorter than 10 seconds and press **Ctrl-C** to stop sniffing.
3. Check the names of the generated file by typing
ls /home/student
4. The binary log file name looks like this: BinCapture.1160008555, in which 1160008555 is the timestamp. Your log file should have a different timestamp.
5. You can read the binary log file using Wireshark or Tcpcap. Type
/usr/sbin/tcpdump -r BinCapture.[timestamp]
6. Record the last three lines of the output (___C1___).
7. Change the permission of the log file so user student can read by typing
*chmod 644 BinCapture**
8. Download this binary log file to your PC desktop. Include it with the completed answer sheet in your final submission 7z file.

D. Using Snort as a NIDS

1. In the SSH terminal, in the student home directory, use wget to download the trace file of a single attack from
ftp://192.168.128.128/pub/lab6/attack.pcap

2. Delete any files in the default Snort log directory by typing
rm -f /var/log/snort/*
3. Run Snort as a NIDS to process this trace file by typing
/usr/local/bin/snort -r attack.pcap -c /etc/snort/snort.conf
4. Check what log files have been generated by Snort by typing
ls -l /var/log/snort
(Note: The file ***alert*** stores the alerts in plain text format; the file ***snort.log.[timestamp]*** stores the binary log in pcap format. If you do not see these two files or if the size of the alert file is zero, most likely something was done wrong between step A33 and step A42. You might want to copy and edit snort.conf again. Also check if there are rule files in /etc/snort/rules. Then redo step D3. If the problem persists, reinstall the virtual machines, start the lab from step A1. You can skip sections B and C after finishing section A if sections B and C have been completed without any problem.)
5. Examine the alert by typing
cat /var/log/snort/alert
6. Record the output (___D1___).
7. Examine the output, when did the attack occur? Answer (___D2___).
8. What is the classification of the attack? Answer (___D3___).
9. What is the destination port number? Answer (___D4___).
10. What service is usually listening on the destination port? Answer: (___D5___).
11. There are a few reference URLs in the output. Check the web sites to find out what vulnerability the attack exploited. Describe here (___D6___).
12. Delete any files in the default Snort log directory by typing
rm -f /var/log/snort/*
13. Start the Snort NIDS by typing
/usr/local/bin/snort -i eth1 -c /etc/snort/snort.conf -D
14. Check the man page of Snort. Find out what the argument “-D” in the command above enables. Answer (___D7___).
15. Log onto Camellia as root. Scan Sunflower by typing
nmap -A -F -T4 192.168.128.128

16. Wait until the scan is over. Examine the Nmap output. What port(s) is/are open? Answer (___D8___). Record the version(s) of the service(s) (___D9___).
17. On Sunflower. Make sure the current directory is /home/student. Copy the alert file by typing
cp /var/log/snort/alert /home/student
18. Examine the alert file by typing
more alert
19. In the alert file, there is one alert regarding “TCP Portscan:”, record the whole alert here (___D10___).
(Hint: Use nano to open the file, then use Ctrl-W to find the string. If you do not see this alert, type *ps aux|grep snort* to make sure snort is listening on Sunflower. Go to Camellia and run the scan again.).
20. Modify the permission of the alert file so user student can read it by typing
chmod 644 alert
21. Download the file *alert* to your PC desktop and include it with the completed answer sheet in your final submission 7z file.
22. Turn off the VMs by typing */sbin/poweroff*. Wait until both VMs are off, close VirtualBox. Find the .VirtualBox folder (where the VMs are located), right click the folder, select **7-Zip - Add to archive**, and click **OK**. The compression process may take more than 20 minutes. Rename the backup file *.VirtualBox.7z* as *VMS_Snort.7z*. Keep the backup (7z) file in a safe place. In the next lab, we will install Basic Analysis and Security Engine (BASE), a web front-end of Snort. We may need to reinstall the VMs from *VMS_Snort.7z*.

End of Lab 6

Appendix B: A Sample Answer Sheet

ICTN 4201 Lab 6 Answer Sheet

Student Name: ()

Email Address: ()

A. Installing Snort from Source Code

A1 ()

A2 ()

A3 ()

A4 ()

A5 ()

A6 ()

B. Using Snort as a Packet Sniffer

B1 ()

B2 ()

B3 ()

B4 ()

B5 ()

B6 ()

B7 ()

B8 ()

B9 ()

B10 ()

B11 ()

C. Using Snort as a Packet Logger

C1 ()

D. Using Snort as a NIDS

D1 ()

D2 ()

D3 ()

D4 ()

D5 ()

D6 ()

D7 ()

D8 ()

D9 ()

D10 ()

Please download *BinCapture.[timestamp]* and *alert* to your PC desktop, and include them with the completed answer sheet in the final submission 7z file. Do not forget to compress the .VirtualBox folder and save the backup file VMS_Snort.7z at a safe place.

Appendix C: The VirtualBox Virtual Machines Used in ICTN 4201 in Fall 2008

