# Adding Topics From Computer Security 101 To A Traditional Operating Systems Course

**Felix F. Dreher**
**Computer Science-Information Systems**
**Pittsburg State University**
**ffdreher@pittstate.edu**

## Abstract

The need to teach computer security topics has been emphasized in the press and in many curricula workshops. While we recently introduced an elective course in computer security, we could not guarantee that our computer science graduates would be exposed to a formal treatment of even the most basic computer security topics. This paper discusses how we added some fundamental computer security concepts and activities to our Operating Systems course. The Operating Systems course was selected since it is a core course and because Operating System textbooks typically cover material that lends itself to some basic, but fundamental computer security activities. We required our students to install and use two different operating systems during the first four weeks. We assigned activities involving these systems to introduce basic security topics early in the class, and we continued to inject security topics into most of the required exercises and reports.

## Background and History

The Computer Science-Information Systems department, CSIS, was established in 1975 in the Kelce College of Business at Pittsburg State University. We have both a traditional BS degree in Computer Science and a BBA degree in Information Systems. We provide computer literacy courses, a Management Information Systems course for the BBA degree programs, and introductory programming classes for degrees in mathematics, the sciences, and the College of Technology engineering technology programs.

The CSIS department progressed from a 'Mainframe' based, batch oriented instructional environment in the 1970's to a 'minicomputer' based, time-sharing instructional environment in the 1980's and then to a microcomputer based instructional environment using a college-wide computer lab for most courses. In the 1990's, almost all of the instructional activities moved to microcomputers running some version of Microsoft Windows© operating systems. In 1998, a small, departmental lab using personal computers was created. This lab is maintained by two members of the CSIS faculty. The lab employs a Windows 2000 Server domain to provide a centralized file server and networked laser printer for students to use. It provides access to various software products required in upper-division classes. We use individual users accounts to limit access to the client workstations and the software in this lab to students taking upper-division courses in our department.

As one of the faculty responsible for maintaining this small, departmental network, we gained a new perspective on the role of security for computer systems. We realized that the background required to provide basic operational level support for a network is quite different from that required to write operating system code or do software development using a particular operating system.

As we observed the placement of recent graduates, we began to see a trend that may well be duplicated in most parts of the country. As a regional institutional university, we attract many students from the immediate area and many of these students wish to remain in the area. During the past decade, our university employed several of our graduates as system administrators. Many local businesses and local governmental agencies have employed graduates as network administrators, systems programmers, end-user support personnel, and technical consultants. We have several graduates who started businesses in computer retailing, networking consulting, or Internet access and local Web Hosting as local ISP's. Students currently enrolled in our program often find positions in end-user support at the University or in local businesses. Other students serve as technical support staff for local telephone and Internet providers. Thus, it is becoming clear that our students are just as likely to be system administrators or technical support personnel as they are to be C++ , Java, or COBOL programmers. Because of this, we saw a need to cover some basic system administration topics and to insure that they get exposure to some very basic security related activities and procedures.

One way to provide coverage of security topics is to teach one or more courses dealing with the area. Four members of CSIS faculty taught an introductory, elective course in

*Information Assurance and Computer Security* during Fall 2003 using a team teaching approach. The course covered four areas: (1) An Overview and the Managerial Viewpoint of Security; (2) Network Security, Firewalls; (3) Encryption and it Role in Computer Security; and (4) Access Control, Host Security, and TCP/IP Overview. As the faculty member who covered the access control and host security area, we saw that we could easily blend some of these basic but very important topics into the traditional Operating System course. By doing this, we can better prepare all of our computer science majors to appreciate the critical issue of computer security and expand the coverage of computer security beyond a set of specialized, elective courses.

## Computer Security 101: Activities For Basic Computer Security

While Trojan Horses, Viruses, Worms, Denial of Service Attacks, and similar network based security issues are widespread and widely publicized, many security concerns were present before the 'Internet' and 'e-commerce' became popular. Panko [1] notes that "the unauthorized access by insiders category is difficult to discuss because it represents a very broad spectrum of transgressions" and that "it is clear that unauthorized insider access to computer systems is fairly common and a credible threat." Patrick Totty, who is a computer security consultant for Credit Unions [2], notes that "perhaps the most dangerous hackers are internal: employees who know the system so well that their intrusions and thefts are virtually undetectable."

Insiders can browse private records, disclose confidential information obtained from a computer system, and engage in many different types of financial fraud, the theft of trade secrets, and even sabotage. Common examples of sabotage include deleting files, destroying equipment, and crashing systems. External threats such as 'hacking' and 'denial-of-service' attacks use both a computer system and a network. The wide spread use of e-mail, web browsing, file transfers, e-commence, and web page servers places an increased emphasis on the design, implementation, and management of the security components for a networked computer system.

There are many different sources for the various activities that can increase computer system's security. One can start with a computer security textbook such as Panko's [1], read readily available documentation for Microsoft and Linux systems for very detailed lists of concrete activities, or obtain material from professional societies, auditing firms, and regulatory agencies. The following list of activities is a good starting point.

- Use strong passwords on all accounts.
- Use access control to limit access to files and folders
- Eliminate unneeded services and processes
- Install Virus detectors, patches, fixes, and secure replacement for compromised services
- Create and audit system logs
- Harden network interfaces by installing firewalls, replacing compromised network services, limit resources available to root users and check for rootkits.

## Security In The Traditional Operating System Course

The traditional coverage of security in an Operating Systems text will be near the end of the textbook, just before the case studies that have more recent material that can to be interleaved with the earlier core chapters. The Computing Curricula 2001 Computer Science model curriculum [3] lists security as an elective component of the Operating System course rather than as a core requirement. It gives the following learning objectives for a computer security component of an Operating System.

> "1. Defend the need for protection and security, and the role of ethical considerations in computer use.
> 2. Summarize the features and limitations of an operating system used to provide protection and security.
> 3. Compare and contrast current methods for implementing security
> 4. Compare and contrast the strengths and weaknesses of two or more currently popular operating systems with respect to security.
> 5. Compare and contrast the strengths and weaknesses of two or more currently popular operating systems with respect to recovery management."

We tried to address these five objectives through out the class. We also introduced a set of activities that could support two of the practical *capabilities and skills* deemed desirable for computer science graduates [3].

> "*Risk assessment.* Identify any risks or safety aspects that may be involved in the operation of computing equipment within a given context. …
>
> *Operation:* Operate computing equipment and software systems effectively."

Panko [1] defines *access control* as "*the policy-driven limitation of access to systems, data, and dialogs*". He listed the following steps that are required to provide for computer security.

> " Determining what resources need to be controlled; determine a policy on access for each resource; identifying who should have access [and what type of access] to which resources; determining how the access control(s) to a particular resource will be implemented, and setting up procedures to insure that the access control policies have actually been implemented."

Many of the fundamental concepts related to the implementation of resource access control are covered in the chapter(s) on security in an operating systems course. The material related to risk assessment and the setting of policy and organizational procedures to insure security of systems are usually not covered. By using the exercises, reports, and text lectures as springboards, we try to provide some coverage of these less technical, but equally important areas.

## Reordering The Topics List For An Operating Systems Course

One goal we added for our Operating System course was to introduce security topics early and often. A second goal was to have students set up and use a new Operating System that could be used for security related activities. We began with the installation of Microsoft Windows 2000 followed by the installation of the Linux Redhat 8 operating system to insure that our graduates had experience with two different types of operating systems. We required them to add user's accounts and set up user directories. We introduced, via several exercise, simple system commands that are needed to use the system to do elementary tasks within the first half of the course. To keep these exercises in step with the lecture topics, we rearranged the order in which we covered the material in our standard text, Stallings [4].

Our syllabus lists the topics to be covered as follows:

- Computer Systems Overview
- Operating Systems Overview
- Computer Security
    - User's Accounts
- Processes
    - Process Overview
    - Basic Linux System Processes
    - Creating and Managing User Processes
- I/O Management and Scheduling
- File Systems Management
- Computer Security
    - Access Rights For File Systems
- CPU Scheduling
- Memory Management
- Multiprocessors and Distributed Computers
    - TCP/IP Overview
    - Sockets, Ports, IP Addressing
- Security
    - Network Based Attacks
    - Firewalls and Similar Defenses
- Threads and Multi-threaded Environments
- Concurrency Issues and Deadlock

This is a rather ambitious schedule, and we are quite rushed by the time we get to Concurrency and Deadlock issues. After many years of teaching these topics, we find that covering these topics much later in the course allows us to present a more realistic set of examples to define the problems and a richer background in discussing the various algorithms typically covered. At this stage of the course, we will have already introduced threads that share memory resources, SMP systems that share access to free

lists and other system data structures, and the role of producer and consumer processes as a model for threaded implementation of file servers, etc.

## The Exercises and Their Relationship To Covering Security 101 Topics

We were able to use most of the ten *exercises* and three written *reports* assigned in this class to add material directly related to security requirements. A *Report* is a short written paper that answers five to seven specific questions about a particular operating system topic. They are designed to have students read contemporary, professional level material, such as a white paper or a portion of the system documentation, which provides a detailed and specific discussion of a small, well-defined topic.

The exercises are hands-on activities done on a network of eight mature computers [Pentium II, 300 KHz, 128 Mbytes RAM] in a dedicated departmental lab. The lab is an open lab. The exercises are designed to be straightforward with *very detailed specifications* much like one would use with a computer literacy course such as *Computer Applications 101*. The exercises reinforce the discussions from the lectures and the textbook and provide a springboard for discussions related to policy, procedures and, in some cases, practical systems management issues. The exercises help to cover a specific set of operating systems concepts, activities, and software tools that we would hope that every one of our computer science graduates would have mastered. In particular, we tried to provide a fairly complete coverage of the foundations of basic security procedures that could apply to most computer systems.

### Exercise Set I: Setting Up Working Systems

The first three exercises and the first report deal with installing the two different operating systems on a computer. They are done as the classroom presentations cover an overview of computer systems, the role and structure of operating systems, and the system startup procedures. Our aim is to have the exercises reinforce the computer overview topics such a Interrupt Requests, CMOS and BIOS memories, I/O Ports, file systems, boot records, and very simple networking topics.

### Report 1: What is Linux?

Our first report deals with Linux licensing compared to other operating systems, what a typical Linux Distribution provides, and the basic system resources needed to support a particular distribution. It also requires a listing of the basic hardware component data that would be needed to install Linux on a PC and a discussion of the options for performing an installation. The material for this report is from the Linux Newbie Administrator Guide [5]. No security issues are involved, but the ethical issue of licenses is raised.

**Exercise 1: Install Windows 2000**

In this exercise, the students install Microsoft Windows 2000. Because of the limited number of computer systems available and the wide range of experiences of our students, we made this a group assignment. Randomly chosen groups of three students were assigned a 300 MHz, Pentium II based computer that had Redhat Linux 7.3 installed with the boot sequence set to boot from the 'C' drive first. The exercise required the team to alter the CMOS Boot Sequence so as to boot from the installation CD, then delete the three existing Linux partitions, create one [or more] partitions, and choose and format the file system assigned to the partition(s) on the Disk Drive. Since Windows 2000 allows a choice of file systems to use, we required the use of NTFS rather than a FAT file system. Students did not install a CMOS password or a loader password. They were given an administrator password and told to set up the networking option for a Workgroup.

This exercise provides several examples of how to address simple policy issues in an Operating Systems context. First, we considered the use of CMOS passwords. Students see how lack of a CMOS password allows one to change CMOS settings that permit changes to the Operating System, the file systems, the disk partitions, etc. We note that simply disconnecting the CMOS battery will defeat a CMOS password. With respect to policies concerning the use of Loader passwords, we bring up the problems they create for remote system reboots, control of password distribution, and how much effort is required to overcome the lack of knowledge of the boot password. Note that *policy* issues, rather that *implementation* issues, can easily be presented in the lectures in a minimal amount of time.

This exercise also gives us the first opportunity to discussion the role of file system support for access rights. We contrast the way Windows 95/98/ME's FAT and FAT32 file systems handle access rights with the Windows NT/2000/XP NTFS file system support for access control. We quickly cover the need for backup and recovery techniques and introduce the value of multiple user partitions on very large disks as a technique to allow the use of selective backups for critical files systems. Again the discussion of designing for security as an implementation step is stressed rather that the details of using FDISK or similar disk partitioning tool.

**Exercise 2: Examine Hardware Resources on System**

The object of this exercise is to establish a non-administrative user account and to examine many of the hardware components of the system. Each student was provided a list of specific hardware devices to examine. Students used the *Task Manager* to view all of the active user and system processes.

This exercise followed the computer system overview material that introduced the processor, interrupts, memory hierarchy, and I/O and communications components. By using the *Windows Hardware Manager,* the students could determine the I/O Addresses, Interrupt Request Lines, Memory Ranges, DMA Channels, and similar data used by the

selected devices. The *Task Manager* is used to view active application [user] and system processes and the allocation of memory to the active processes.

Observations concerning the use of Task Manger to terminate processes and observe rogue processes that may be active on the system allow us to tie this into a security topic. We covered the Microsoft Windows XP *services management tool* in the classroom as we covered processes. Policies related to which system services should be allowed on different types of systems are discussed. We can easily cite examples of services such as FTP and Mail servers, and Telnet that have potential problems.

**Exercise 3: Install Redhat Linux 8.0**

This was a team exercise in which the teams assigned in Exercise 1 will install a copy of Redhat Linux 8.0 on their assigned system.

The object of this exercise is to install an operating system to be used for the remaining exercises. For many of the students, this will be their first encounter with a non-Microsoft Operating System. The Linux distribution requires the students to select an installation that can be configured to support several different types of applications. Our particular distribution allowed a workstation, server, desktop, and custom installation. We required a custom installation be done, and we provided a complete list of packages to install. We used the traditional LILO loader with no password and required that it be installed on the Master Boot Record.

Students used the hardware information from exercise 2 to verify that the hardware probing of Linux 8.0 is very good for mainstream systems. Disk partitioning was done by use of the *Auto-Partition* option. In future classes, we may want to specify a set of system partitions, Root, Boot, and Swap, and then add several user-defined partitions to reinforce the use of partitions for enhanced security and backup and recovery operations.

**Set II: Using The Operating System To Perform Useful Tasks**

The next four exercises covered some of the basic activities required to actually use an operating system to do something useful. Only one exercise used a GUI desktop, KDE, to perform the activities. Basic text based commands are introduced in the remaining exercises to provided a foundation for discussions of system functions and the use of shell scripts that are often used with minimal systems such as routers, Domain Name Servers, and Web Servers. Each exercise is done by each student.

**Exercise 4: Users, Groups, Passwords, Files, and Directories**

The object of this exercise is to use the GUI tools provided by the KDE desktop to create a set of resources that can be used by a group of users. Each user was asked to create a

set of accounts for a small group of co-workers. Each user was to have a personal account, a private home directory, access to a group directory, and personal subdirectories within the group directory.

The lecture material will be covering user accounts, passwords, and Linux style access control for Directories and Files. The distinction between the root user and ordinary user is also discussed.

**Report 2: FreeBSD Accounts**

This report covers the types of users accounts available in FreeBSD and the operations related to account management. Disk quotas and the use of localization [to support multi-national environments] are covered. Material for this report comes from the FreeBSD Handbook [6]. The need to select a kernel option to enforce disk quota provides additional insight into the structure of a kernel and the ability to customize a kernel.

**Exercise 5: Common Linux Terminal Commands Including Mount and Umount**

This exercise deals with very basic, text based commands using the bash shell for Linux. It introduces the *mount* and *umount* commands for use with removable devices. Students are exposed to the use of pipes, I/O redirection, and script files to capture the terminal output. They compile a *c* program and a *C++* program using the text mode commands for GNU *gcc*. The resulting executable files are used with by a Perl script.

The policies related to mounting file systems in Linux are discussed as well as the ability of the root user to change the access rights of the removable devices to allow others to mount them. Such a policy would have an impact upon insider theft. The question of supporting similar restrictions in Microsoft Windows environments is raised. We used the "Help and Support Center" feature of XP to locate the access folders control options available for both networked Workgroups and Domains. The keywords "mount removable storage", gave us the command *rsm* that can be used in scripts dealing with Zip Disks and various CD devices. Displaying the related topics leads to mounting and dismounting of tape or disks. This provides an opportunity to contrast the approaches to backup and recovery used by two different types of operating systems.

**Exercise 6: Linux /proc File System**

This exercise has the students exploring the Linux */proc* file system that holds much of the system information used to manage the hardware resources of the system. The /proc file system also provides a directory for each active process. Using the *ps* commands to find the active Terminal Sessions PID and Process ID, the students can use the associated directory in /proc to determine the status, the environment, memory maps, etc. associated

with the process. This reinforces the concept of a process being more than code and data.

The most significant security issue for this exercise is the ability of a root user to write to the various files to change selected options for individual devices and processes. When accessed by an ordinary user, the access to various files is denied.

**Exercise 7: Scripts in Bash, Perl, and Tcl/Tk**

This exercise requires student to execute several bash and perl scripts and then to modify them by providing command line arguments to provide input and output files. Students were asked to write and test simple bash scripts that used file arguments as inputs. They had to test for file existences using simple *if statements* with file comparison operations. This test reinforces the need to prevent the unintentional overwriting of data files.

**Report 4: Managing Memory-Mapped Files in Win 32**

This report covers the role of memory-mapped files as a mechanism to access files that take advantage of the Virtual Memory system resources. The discussion of sharing a memory mapped file provides an opportunity to discuss the use of *security_attributes* and access rights associated with the file object and the role of inheritance with respect to child processes. Details of the memory-mapped files in windows are found in [8].

**Exercise 8: Installing Software In A Linux System**

This exercise deals with the initial groups of students assigned to an individual system. Each group is responsible for installing additional software on the system. Using a group account, each member of the group installs a software product using the Redhat Package Management command, rpm. Each member is then required to install a different software package provided by a compressed .tar file using the various commands. Some of the software packages that are available to install include the following items:

| **Package** | **Use** | **Format** |
|---|---|---|
| Pine | An e-mail client | rpm |
| Cheops_ng | Network monitoring and Mapping Program | rpm & tar |
| Lokkit [gnome] | Simple Filewall Package | rpm |
| Splint | Secure c Program Utility | rpm |
| Sendmail Fixes | Security Fixes For Sendmail package | rpm & src |
| Command_history | Perl Script Shows Commands Issued For Day | tar |
| Filemon | Monitor File System Activity | tar |
| Webmin | Web Based System Administration Utility | tar |
| AIDE | Advanced Intruder Detection Environment | tar |

The goal of this exercise is to provide students an opportunity to use the two principal techniques to install new software packages for a Linux system. The Cheops_ng, Lokkit, Splint, Filemon, Tripwire, AIDE, and Sendmail Fixes packages all relate to the software that is available for improving the security of a system. Webmin can be used to manage various network-based services such as Apache, DNS, DHCP, NFS.

**Set III: Distributed System Concepts and Security Issues**

The final set of activities use a simple networked environment. The network consists of the eight individual workstations combined to form an Ethernet network. Only the simplest form of networking is used and only basic operations are covered. Each system uses a *hosts* file to provide connectivity. The concepts of sockets, servers, IP addresses, TCP/IP Stacks, Medium Access Control addresses, Network Interface Cards, and standard TCP/IP services are covered in the lectures.

**Report 5: Security Issues For Redhat 8 Linux Systems**

Students are to outline workstation and server security issues and describe several different techniques that are recommended to enhance the security of a Redhat Linux 8 installation. Specific questions deal with passwords, services that are often used to attach a server, how to deal with services and the need to fix insecure services. A brief discussion of several different types of Firewalls currently in use, the role of Linux kernel netfilter module and the roles of iptables to restrict access to a server are required. The material introduces the chkconfig command that can be used to check the state of services on the system. It also shows how the simple command

*"echo "1" > /proc/sys/net/ipv4/ip_forward"*

is used to turn on packet forwarding to allow the system to provide a routing function.

**Exercise 9: Simple Networking in Linux**

In this exercise, the students use text based commands *ifconfig*, *ping*, *ifdown*, *ifup*, and *netstat* to view and control the network connections. They examine the *hosts*, *hosts.allow*, *hosts.deny*, and several other files associated with the network configuration. They use the *md5sum* command and I/O redirection to capture a checksum of a file. They use the file difference checker, *diff*, to check for differences between the files and checksum files. They use *last* to capture the recently used commands and use the superuser command, *su*, to gain access to various system logs owned by root.

The introduction of md5sum checksums provides an opportunity to explain how some packages such as Tripwire [10] or Aide [11] use checksums and hashing functions to create a database that can be used to detect changes to files within a file system.

**Exercise 10: Chkrootkit, chkconfig, Profiles, and Start Up Scripts**

The final exercise is quite short and simple. It introduces the Linux *chkconfig* command that is used to list and update the various run level services used at boot up with the Linux system. Students install the root kit checker, *chkrootkit* [12] and use it to check for a problem with the current root installation. Next, they examine the global profile setting files, /etc/profile and /etc/bashrc and the local profile files associated with their home directories, ./bashrc and ./bash_profiles. They examine the bootlog and rpmpages log.

This exercise exposes the grunt work of security. Checking configurations, active services, system and user profiles, and system logs to detect changes to the system.

## Conclusions and Observations

There are many different treatments of the standard Operating System course among the many Computer Science curriculum implemented at the undergraduate level. The background of the students in these classes is also quite varied. Some students have installed operating systems in computers they assembled from individual parts and others have never even seen the inside of a computer and have never installed a program on a computer system. Most of our graduates will not pursue an advanced degree and will not be employed in writing operating systems software. To address the needs of students who take this required course, we tried to strike a balance between the coverage of the basic theory and the coverage of materials that the students will consider useful. The strategy of adding a computer security emphasis is an attempt to achieve a better balance by adding material from concrete systems that overlaps with an area in which there is growing interest. Students engage in hands-on activities that add to their appreciation of the Operating System as complex sets of programs that work together to achieve many diverse goals, including that of protecting system resources. Very little of the material being presented is covered in earlier courses, as they are typically much too full. Our efforts to make a simple revision of the order of presentation of topics in the Operating System course coupled with some focused exercises seems to be a realistic way to insure that all of our computer science graduates are exposed to these important computer security concepts.

Many of the topics considered in our various exercises, reports, and even asides in class are well covered in the professional literature. Much of the material used is available from the Internet and most can be freely copied for distribution to classes. The software products used with both Microsoft Windows and Linux systems related to testing the security of systems are easily found and can form the basic for exercises similar to those used in this class.

## References


1. Panko, R. R. (2004). *Corporate Computer and Network Security*. Pearson.

2. Computing Curricula 2001: Computer Science (Final Report), The Joint Task Force on Computing Curricula, IEEE Computer Society, Association for Computing Machinery.

3. Terry, Patrick, "Keeping Hackers at Bay: Intrusion Detection Systems Reveal Server Vulnerabilities", Credit Union Magazine, Feb. 2004.

4. Stallings, William, Operating Systems, 4th Ed., (2001) Prentice-Hall.

5. Kliman, Peter and Stan, *Linux Newbie Administrator Guide,* http://www.Linux-Newbie.sunsite.dk/

6. FreeBSD Handbook (2003), http://www.freebsd.org

7. The Linux Systems Administrator's Guide, http://en.tldp.org/guides.html

8. Kath, Randy "Managing Memory-Mapped Files in Win32", Microsoft MSDN Library

9. Red Hat Linux 8.0: The Official Red Hat Linux Security Guide, http://www.redhat.com/docs/

10. Tripwire, http://www.tripwire.org

11. AIDE, http://sourceforge.net/projects/aide

12. chkrootkit, ftp://ftp.pangeia.com.br/pub/seg/pac/chkrootkit.tar.gz