SUMMER WORKSHOP ON CYBER SECURITY

CYBER SECURITY EDUCATION

DR. AKBAR NAMIN

Texas Tech University
August 12-16, 2013

Supported by National Science Foundation
A HOLISTIC, MODULAR APPROACH TO INFUSE CYBERSECURITY INTO UNDERGRADUATE COMPUTING DEGREE PROGRAMS

TRUDY HOWLES, CAROL ROMANOWSKI, SUMITA MISHRA, AND RAJENDRA K. RAJ
In response to societal change and national educational objectives, a holistic, modular approach to Cybersecurity education is presented in this paper. This approach is characterized by a set of reusable, self-contained modules that can be embedded in existing classes in several computing disciplines.

The intent is to introduce these modules across computing disciplines, and throughout the undergraduate years to ensure a greater understanding of security issues among diverse computing majors.

The ultimate goal is to address the societal need for computing professionals who are educated and experienced in diverse aspects of computing security and information assurance.
I. INTRODUCTION

- With the growth and pervasiveness of cyber infrastructure in modern society, secure computing and communicating have become critically important. Applications with critical security requirements include e-commerce, voice/video communications, military operations, secure data management, and financial market transactions.

- In addition, there is a growing trend toward adding security at various points in the information technology infrastructure, such as embedding in disk drives, processors, trusted system boards, network switching elements, mobile devices, and sensors. All of these points require careful attention to algorithm choice and implementation method, and involve trade-offs between software and hardware.

- The development of such systems requires a population of developers who possess the necessary knowledge and design skills. Students must learn fundamental theory and gain practical implementation experience to understand security requirements holistically.
This paper presents an approach to develop and deliver a broad range of concepts and topics associated with security and software design through modules embedded in existing classes in multiple disciplines, including Computer Engineering, Computer Science, Software Engineering, and Cyber Informatics.

These course modules provide sufficient breadth and depth in information assurance concepts, security requirements elicitation and analysis, and specific techniques that are widely used in industry applications. The modules begin with introductory concepts and gradually introduce the necessary theoretical background.
II. A HOLISTIC, MODULAR APPROACH

- This proposed holistic, modular approach embeds cybersecurity education in computing courses, at the undergraduate level, vertically from freshman to senior year. The course modules in security and information assurance are designed to enhance cybersecurity instruction within the computing disciplines.

- Societal change is the impetus behind this approach; the rapid advance of mobile computing devices such as smartphones and tablets, along with the growth of cloud computing, has made cybersecurity a concern of everyone.

- Issues of cybersecurity and security breaches impact all of society, not just computing professionals [1]. These topics must be incorporated throughout the curriculum, so that students confront these concepts several times during their academic years.
The foundation of the approach is a set of self-contained instructional modules that can be “dropped in” to relevant classes in computing disciplines such as Computer Engineering, Software Engineering, Network Security, Information Technology, and Computer Science.

A module defines as a distinct unit of course materials, such as a lab or a teaching component, that can be incorporated into existing courses in the curriculum by an instructor without requiring any course or degree program changes and curricular approval.

Modules cover user, legal, ethical, and technical aspects of cybersecurity in appropriate detail—from introductory material to deeper, more technical concepts—and can be combined into a stand-alone course or sequence of courses in these topics.
The modules are created with a common style that is coordinated across module categories (technical, social, legal, etc.). This coordination extends to the format of exercises, assessments, and other instructional elements.

Module developers work with instructional designers to create modules that are self-contained, consistent and can be linked together to form larger blocks of instruction.

Each module contains a short introductory section presenting the broad picture of cybersecurity and where each module fits within that overview.

Fig. 1 shows example module categories and possible topics in each category. This figure depicts the long-term vision for how modules and courses can impact a wide cross-disciplinary audience.
Fig. 1. Module categories and possible topics (Note that topics may span multiple categories, but are shown under one category for convenience.)
Modules are a common pedagogical tool for computing and cybersecurity topics, but mostly used in a single course or set of courses within computing disciplines [2-7].

A recent panel [5] comments on the multidisciplinary nature of security, the need for all computer users to have some security knowledge, and the effectiveness of addressing these concepts in courses holistically rather than focusing on stand-alone classes.

Modules have also been used to embed security topics rapidly into an existing information assurance curriculum [7].
III. SAMPLE COURSE MODULES IN COMPUTING DISCIPLINES

A. Module: Security Requirements

- This course module will focus on presenting fundamental security issues and their optimization criteria. A brief overview of standard mechanisms for achieving them will also be visited, such as Input Validation, Access Control, Session Management, Cryptography, Auditing and Logging. This module is to be added to introductory courses for computing and engineering students.
B. Module: Security of Web Applications

- This set of course modules will concentrate on the specification, design and implementation of specific security requirements introduced in the Security Requirements course module.
- Possible requirements include Identification and Authentication Requirements; Authorization Requirements; Immunity Requirements; Integrity Requirements; Intrusion Detection Requirements; Non-repudiation Requirements; Privacy Requirements; Security Auditing Requirements; etc.
- Each course module will focus on one specific security requirement, its optimization criteria, and techniques for achieving and analyzing it. The educational outcome of this set of course modules is to provide student with enough depth in software development, especially web applications, with a strong emphasis on performance analysis of implementation.
C. Module: Security and Privacy Issues in Network Systems

- Topics include basic cryptographic protocols, key management protocols, Denial-of-Service attacks and defense mechanisms, wireless network security, key management in multicast and wireless sensor networks, Web service security, and network worms.

- More specifically, the students will learn about the symmetric key cryptographic protocol, public key infrastructure, one-way hash function with data integrity, Diffie-Hellman key exchange, the mechanisms of distributed denial-of-service attacks with possible defense solutions, how to secure wireless LAN, PEAP, and TTLS, how to secure wireless routing, and key management.

- Students will also discuss how multiple nodes in networks can manage their keys in a secure way for group communications. In addition, security risks and their solutions in Web service including TLS/SSL will be studied, and the spreading patterns of network worms with possible solutions will also be investigated. The students will study these principles, then devise attacks in laboratory exercises or projects, and implement defense solutions in actual systems.
D. Module: Mathematics for Cryptography

- This course module will focus on discussing the fundamental mathematics used for private and public key encryptions.
- The purpose of the module is to extract the basics from a comprehensive cryptography class, and introduce security related examples to math, software and hardware design courses.
- The students are not required to have backgrounds in cryptography or networking. As only a basic knowledge of college-level math is needed, this module will reach a broad audience in addition to computing and engineering students.
E. Module: Machine Learning for Network Security

- This course module will discuss the application of machine learning algorithms for intrusion detection, alert correlation, and cyber attack prediction.
- A brief overview of selected machine learning algorithms, such as Bayesian Network, Hidden Markov Models, and Fuzzy Inference will be provided, with case studies of how these algorithms are applied to security problems.
- The educational outcome of this module is to familiarize students with algorithmic analysis and the implication of machine learning to active cyber defense.
F. Module: Security Issues in a Virtual Environment

- This module will introduce the advantages of virtual computing and the security challenges presented by this environment.
- Full virtualization simulates the underlying computer hardware and enables the software to run without changes.
- Virtualization enhances the performance of cloud computing systems by optimizing the computing workload and managing the servers more effectively.
- However, virtualization creates additional layers, hence creating the need for additional security controls and measures. If not managed carefully, the ease of sharing information between systems can be turned into an attack vector very easily.
- It is challenging to create and maintain the necessary security boundaries due to the dynamic nature of virtualization.
- Students will learn secure virtualization techniques in the classroom and the concepts will be reinforced through lab exercises.
G. Module: Secure Coding

- Following the Software Assurance Curriculum Project Volume II: Undergraduate Course Outlines (from cert.org) as a guide, the initial effort will be a gap analysis to identify the topics missing in current programming courses.

- Topics will be mapped to existing courses to identify ways in which to integrate new labs or modules to introduce the material. The “CWE/SANS Top 25” most dangerous and critical software vulnerabilities (cwe.mitre.org/top25/) [8] will also guide this gap analysis.

- Students will study the key vulnerabilities and learn the mitigation techniques for each one. An anonymized archive of previous student project submissions will be used as case studies; students will be challenged to find and correct the vulnerabilities. This is just one hands-on activity designed to help students identify vulnerabilities and weaknesses, and realize the inherent threats in legacy code. Learning modules will address general problems as well as language-specific issues.
To foster a culture of security awareness, students will be required to create a security label (similar to a food nutrition label) for all their course deliverables [9].

These innovative and fun labels serve a dual purpose: to reinforce that programming practices have a continuing impact on the general security of the cyber infrastructure, and that users must become more informed and savvy about what they are purchasing, using and downloading onto their personal devices.
IV. ASSESSMENT AND EVALUATION

- For each of the goals, they have developed a set of objectives, each of which has concrete achievable outcomes that will be assessed and evaluated as the effort proceeds. (Following customary pedagogical practice, used the term assessment to mean the gathering of measured data, and the term evaluation to mean the analysis of the assessed data.)

- Findings will be used to refine the learning experience, research activities, and evaluation methodologies.

- Table I presents the objectives and intended outcomes of this modular approach to cybersecurity education.

- The plan is to use both direct and indirect methods to measure each of the outcomes listed in the rightmost column of Table I. When possible, we will use direct methods, e.g., examination of concrete measurable data, which have greater reliability than indirect methods, e.g., student self-surveys.
<table>
<thead>
<tr>
<th>Goal</th>
<th>Objective</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Embed security and information assurance concepts across the existing university curriculum in computing disciplines.</td>
<td>1.1 Computing majors learn additional Cybersecurity concepts.</td>
<td>1.1.1 Course modules in Cybersecurity are embedded into existing courses taken by computing majors.</td>
</tr>
<tr>
<td></td>
<td>1.2 Computing majors apply Cybersecurity concepts in undergraduate capstone projects.</td>
<td>1.2.1 Capstone projects reveal the application of Cybersecurity knowledge.</td>
</tr>
</tbody>
</table>
V. CURRENT STATUS OF WORK IN PROGRESS

- There are three modules that have been piloted to students in the form of short noncredit seminars.
- The initial modules focus on secure coding, spotlighting the most serious and frequently occurring vulnerabilities.
- At the conclusion of each seminar, students were asked to provide survey feedback on the content, and their assessment of the relevance of the material.
- Preliminary feedback indicates that students are interested in the material; this is supported by the strong attendance patterns at the seminars.
- Student comments indicate that they do recognize the importance of security and safety and are interested in learning more.
- Student feedback along with input from industry partners have been helpful in both fine-tuning the content of these early modules, and helping to identify the most critical modules to target next. An additional four modules are currently under development, and are targeted for rollout in the fall of 2011.
VI. CONCLUSION

- This paper outlines efforts to embed cybersecurity modules throughout the computing disciplines, in response to the societal need for computing professionals educated and experienced in computing security.

- Since security issues affect everyone, not just the core information assurance community, this modular approach ensures that all computing students, regardless of degree program, receive instruction in security concepts.

- A different but concurrent effort focuses on the development of similar security modules to disseminate security knowledge to all students, not only computing majors [10].

- With the rapid growth in mobile computing devices and diverse types of wireless networking (e.g., 802.11, Bluetooth, 3G, and 4G), computing security has become critical for undergraduate disciplines, just as writing skills are considered important (e.g., Writing Across the Curriculum).

- Therefore, a conceptual hands-on education in cybersecurity must be provided for as much of the student body as possible.
Targeted here are General Education courses offered in non-computing disciplines as well as other relevant classes such as first year experience courses, public policy, criminal justice, psychology, economics, and industrial and systems engineering.

This program enhances our existing academic offerings, and we believe it will attract more students to security-focused majors.

The initial positive feedback and increased interest students have shown are encouraging and support this belief.

As we develop and deliver more modules, we will continue to survey our students and gather input from our industry partners on the program’s direction and effectiveness.
COURSE MODULES FOR SOFTWARE SECURITY

Austin Frazier, Xiaohong Yuan, Yaohang Li, Stephan Hudson, North Carolina A&T State University
ABSTRACT

- Each year the reported number of security vulnerabilities increases as does the sophistication of attacks to exploit these vulnerabilities.
- Most security vulnerabilities are the result of insecure coding practices. There is a critical need to increase the security education of computer science students, particularly in software security.
- This paper is designing course modules, to be used at the undergraduate or graduate level, to integrate software system security into our computer science curriculum.
The course modules have developed, and are developing, include: operating system security, software security testing, code review, risk analysis, and database security.

Each course module includes lecture materials, in-class demonstrations, and hands-on assignments. These course modules are designed to be integrated into existing courses.

The software security testing and database security modules were taught at this university in the Fall 2007 semester and received positive feedback from student surveys and questionnaires.

Future work will include the development of more topics in these modules and the creation of new modules in secure software development.
I. INTRODUCTION

- Security has become an increasingly important topic in computer science. Each day we learn of new vulnerabilities or hear of new attacks. Most of these vulnerabilities and attacks are, in fact, due to software defects. According to CERT/CC, between 1995 and 2006, the number of reported software vulnerabilities increased by fifty two percent each year.

- In recent years, a number of colleges and universities have recognized the need for information assurance education and have developed information security courses in the computer science or related disciplines. However, most of these courses focus on network security. Courses and educational resources on software security are very few. Therefore, there is a significant demand for introducing software security into computer science or information assurance curriculum.
This paper discusses the design and development of software system security course modules. These modules will be integrated into existing computer science curriculum. The developed course modules can be integrated into four courses in the department.

Each module consists of several lectures along with in-class demonstrations and laboratory assignments. Instructors can decide how much of a module they would use in their class. The course modules they have developed/are developing include:

1. Operating system security module
2. Software security testing module
3. Code review module
4. Risk analysis module
5. Database security module
In the Fall 2007 semester, the database security module was taught in the senior-level undergraduate “Database Design” course and the software security testing module was taught in the graduate “Software Specification, Analysis and Design” course. This course traditionally covers software testing but does not include secure software testing. Both of these modules received positive feedback in surveys and questionnaires that were given to the students.

The operating system security module has been developed and taught in the senior-level undergraduate “Operating Systems” course in the Spring 2008 semester. They are developing the risk analysis and code review course modules, and they will be used in the graduate “Software System Design, Implementation, Verification and Validation” course.
McGraw [1] introduces a framework for secure software development (Figure 1).

Lightweight software security best practices, called touchpoints, are applied to the normal steps that make up most software development lifecycles. Each touchpoint is ranked by order of effectiveness.

Walden and Frank [2] proposed ten modules that correspond to some of McGraw’s touchpoints. These ten modules were taught in a seminar course.
Figure 1: McGraw’s secure software development life cycle
The two course modules that they designed, code review and risk analysis, correspond to the two most effective touchpoints of McGraw's secure software development cycle.

The software security testing module corresponds to the penetration testing touchpoint. These modules are similar to those by Walden and Frank [2] in that they cover similar topics; however our modules include more tools.

Several different static code analysis tools are used in code review. Also, in code review they added security checklists similar to those created by Taylor and Azadegan [3].

In risk analysis, McGraw's process is presented in addition to Microsoft's STRIDE/DREAD model.
Idaho State’s NIATEC [4] has developed eight teaching modules in Information Assurance (IA) that cover a broad range of IA topics.

These topics include information protection, PC/Workstation security, security fundamentals, information security law and legislation, system and communications security, corporate security management, etc.
II. OPERATING SYSTEM SECURITY MODULE

- Most operating system courses introduce some security topics such as security threats, access-control, malicious software, etc.
- The operating system security module they developed builds on that material and provides students with hands-on experiences.
(1) Buffer overflow and format string attacks [5].

- Since most operating system vulnerabilities are caused by insecure coding practices, it is important for the students to learn about dangerous programming errors.
- The students will learn what buffer overflow and format string attacks are, why they are dangerous, and how to mitigate them.
- Buffer overflows are an important topic in operating system security because they are the most common way for an outside attacker to gain access to a system [6].
- This course module will illustrate how buffer overflows can be used at the kernel level to gain access to the operating system. An example of this would be exploiting a buffer overflow on a network daemon to gain a root shell.
Operating-system level mitigation techniques, such as using a non-executable stack, will also be discussed. A format string vulnerability was found to affect core APIs in most UNIX versions [5] in CVE-2000-0844 [7].

According to CVE: “Some functions that implement the locale subsystem on UNIX do not properly cleanse user-injected format strings, which allows local attackers to execute arbitrary commands via functions such as gettext and catopen.” Samples of these vulnerabilities in C code are introduced along with in-class demonstrations.
(2) The dangers of using weak or default passwords.

- The students will learn how to create secure passwords. The students will use a password strength checking website to test the strength of various passwords.
- In the laboratory assignment, the students will use the open source tool Cain and Abel [8] to do dictionary, brute-force, and cryptanalysis password attacks.
(3) Open network ports, rootkits, and the vulnerabilities of Windows and UNIX services.

- Students will learn why these are dangerous to operating system security and how to test for them [5] [9].

- The students will use the open source tool Nmap [10] to identify unnecessary or rogue services/applications running on the machine. Students will also observe an in-class demonstration that was created for the scanning and detection of rootkits [11].
(4) Operating system vulnerability analysis.

- The students will use Microsoft Baseline Security Analyzer (MBSA) [12] and Nessus [13] to find the vulnerabilities of a Windows 2000 virtual machine.
- MBSA is a free tool to determine the security state of the Windows operating system in accordance to Microsoft and to offer recommendations. MBSA reports the status of firewalls, missing updates/patches, unsecured user accounts, shares, unnecessary services, and other security vulnerabilities.
- Nessus is a vulnerability scanner that uses a vast database to scan and report vulnerabilities for specific operating systems.
- Students will get exposure to an improperly secured operating system and the reports generated by the two vulnerability scanners.
- In the case of MBSA, students will learn how to fix the problems found in the report at the system administration level.
- The development of this module is ongoing and will include operating system security problems caused by race conditions in the future.
III. SOFTWARE SECURITY TESTING

- Traditional software testing is primarily focused on determining whether software meets requirements and specifications. It mainly looks for bugs that don’t follow specifications.
- However, symptoms of security vulnerabilities are very different from those of traditional bugs.
- Security bugs are found by looking at additional software behavior, their side effects, and how the software interacts with its environment [9].
- Software engineering courses usually only discuss traditional software testing. Software security testing is not covered.
- Their course module introduces software security testing in addition to traditional software testing.
The software security testing module first introduces a fault model as a basis for security testing.

This fault model is made up of four “users” that an application will interface with: the kernel, APIs, the file system(s), and the user interface.

Students will learn that based on this fault model, an attack plan can be created which includes attacks that fall into four categories:

- software-dependency attacks,
- user-interface attacks,
- attacks against the application’s design, and
- attacks against the implementation of that design.
This course module introduces attacks in two categories: user-interface attacks and attacks against the application’s design.

These two categories of attacks were chosen initially because software engineering courses typically cover the topics of software design and user interface testing.

These two categories of attacks can be easily integrated with existing software engineering topics to introduce software security.
The following types of attacks are introduced to the students, grouped in their respective attack category by Whittaker and Thompson [9]:

- **User-interface attacks:**
  - Buffer overflows
  - Enabling common switches and options through the user interface
  - Cross site scripting (XSS)
  - Format string attacks

- **Attacks against the application’s design:**
  - Failing to handle errors
  - Forcing the system to reset values
  - Creating loop conditions
  - Data flow attacks
Each attack is explained in the following format: when the attack is applied, why the attack is successful, how to determine whether security has been compromised, and how to conduct the attack [9]. Code examples, both bad and corrected, and in-class demonstrations are introduced for each of these attacks.

The software security testing module includes three hands-on laboratory assignments. Two of these hands-on laboratory assignments are based on the OWASP LiveCD Education Project [14].
In the first laboratory assignment, students will use WebGoat [15] to perform reflected cross site scripting and stored cross site scripting attacks. WebGoat is a deliberately insecure J2EE web application designed for teaching software security.

In the second laboratory assignment, students will learn how to exploit hidden fields and use WebScarab [16] as a proxy tool to intercept and modify parameters. WebScarab is a security testing tool that is able to intercept both HTTP and HTTPS communications.

In the third laboratory assignment, the students will use the Acunetix web vulnerability scanner [17] to scan for XSS vulnerabilities from several Acunetix test web sites.

This course module may be integrated into a junior/senior/graduate level software engineering course that covers traditional software testing.
IV. CODE REVIEW MODULE

- Code review was listed as the most important phase in McGraw's secure software development life cycle (SDLC).
- According to McGraw [1], fifty percent of the security bugs can be eliminated in this phase. It is important to test for security as early as possible in the SDLC. Doing so reduces costs and improves efficiency.
- It is much more time consuming and costly to search through and patch an application after it has been deployed.
In the code review module, the students will conduct code review using security checklists.

This module will be integrated into the graduate level course “Software Design, Implementation, Verification and Validation” in which Personal Software Process (PSP) is taught and practiced.

Code review is an important software development phase that the students learn and practice in that course.

Security checklists for buffer overflows, integer overflows, input validation, and cross-site scripting have been developed and will be used in this course.
Each checklist presents between four to seven conditions that can be checked to indicate the presence of vulnerabilities.

The security checklists were developed based on the security checklists presented by Taylor and Azadegan [3] and from other related literature [18] [19] [20] [21] [22] [23] [24] [25].

Extensive sources were used in an effort to make the checklists as complete as possible. However, passing a checklist does not guarantee that the vulnerability has been completely eliminated.
In addition to introducing all of the security checklists, this course module also: presents the pros/cons of using a security checklist, discusses the rationale behind how each of the checklists were created, and gives a code example demonstrating how each checklist is to be used.

Besides using a checklist to look for traditional software bugs, the students will also use the security checklist to look for security bugs during the code review process.

Next, the students will learn to use static analysis tools and compare the results they get from their manual code reviews with those found by the static analysis tools.
The static analysis tools that will be used include:

- Flawfinder [26] for C/C++ code samples
- RATS [27] for C/C++ and PHP code samples
- OWASP’s LAPSE [28] for Java web applications
- Microsoft’s XSSDetect [29] for XSS vulnerabilities in C# code samples
- Fortify [30] for C/C++, Java, and C# code samples (Note: This version of Fortify is for demonstration only. It only scans for buffer overflow and SQL injection vulnerabilities.)
Static analysis tools use large rule sets to scan for vulnerabilities and are a much faster way to find vulnerabilities.

These tools will be used against code samples, student homework assignments, or a semester long project.

Students will perform several hands-on laboratory exercises where they learn to use each tool and analyze the reports, or results, generated by each tool.

Students will evaluate the results and check for false positives.

They will also be able to compare the results of several tools on the same piece of code.

The commercial static analysis tool, Fortify, represents a huge leap over earlier tools because it incorporates compiler technology.

As a result, Fortify can track control and data flow compared to only keyword searches from earlier tools.
V. RISK ANALYSIS MODULE

- McGraw states that fifty percent of software security problems are caused by design flaws (architectural and design-level problems) [1].
- Therefore, it is important to identify potential threats, determine the level of security risks posed by these threats, and determine which threats require mitigation.
- To achieve this goal, the risk analysis module will consist of Microsoft’s Threat Modeling framework [31] and McGraw’s architectural risk analysis [1].
A threat model is defined as “a security-based analysis that helps people determine the highest level security risks posed to the product and how attacks can manifest themselves” [31].

 Threat models also have the added benefit of helping developers understand what the application, as a whole, is actually doing.

 The students will first learn the threat modeling and risk analysis methodology.

 They will construct data flow diagrams at different levels to decompose their semester project or selected examples. These diagrams list the data flows and assets of the application.
The students will then determine threat targets. The STRIDE (spoofing, tampering, repudiation, information disclosure, and elevation of privilege) method will be used to categorize threats against these targets.

After building one or more threat trees, the DREAD (damage potential, reproducibility, exploitability, affected users, and discoverability) method will be used to calculate the security risk for each threat tree.

The students will learn different ways to respond to and mitigate threats. The students will also use Microsoft’s Threat Analysis and Modeling Tool [32] to perform risk analysis on their semester project or a selected example. Video demonstrations will be used to help the students learn how to use the tool.
McGraw’s architectural risk analysis is made up of three steps:
1. Attack resistance analysis,
2. Ambiguity analysis, and
3. Weakness analysis.

- Attack resistance analysis uses checklists to identify known threats.
- Ambiguity analysis is the process of discovering new risks that would be otherwise missed by just looking for known attacks.
- Weakness analysis involves understanding how flaws in external dependencies affect applications.
- Students will perform a brainstorming activity in their project groups to identify unknown or creative attacks that might occur against their project and identify any external dependencies for security flaws using security websites such as BugTraq [33] or Security Tracker [34].
VI. DATABASE SECURITY MODULE

- There is a strong need to introduce the issue of security into database courses. One particularly popular and harmful type of attack is SQL Injection. Using a SQL Injection attack, a hacker can steal user names and passwords, steal all the tables in a database, or simply delete them.

- The database security course module includes a lecture, a hands-on laboratory assignment, and a homework assignment.
- The lecture discusses how SQL Injection attacks occur and how to prevent SQL Injection attacks.
- In the laboratory assignment, the students use a SQL Injection attack to gain access to a web site.
- The students guess usernames and passwords stored in a database. The homework assignment includes questions based on the information discussed in the lecture and in the laboratory assignment.
The laboratory environment for both the software security testing and operating system security modules is a Windows 2000 virtual machine (VM) that was created using VMWare.

In the future, this VM will be upgraded to Windows XP Professional.

A VM allows for the creation of one operating system image to be loaded on as many computers in the laboratory as there are students in the class.

The VM provides a safe environment for the students to use security tools without harming the environment of the computers in the laboratory. The VM can also be reloaded in the event that it becomes corrupted. VMs save time and provide a uniform environment for all the students during the laboratory exercises. The VM's operating system and software may be updated for use in future semesters, if desired.

All in class demos and laboratory assignments for all the software security course modules are included on a single VM. All the software used was either free, open-source, demonstration, or already licensed by the school. An existing computer laboratory, for computer science students only, will be used to host the software.
INFUSING SOFTWARE ASSURANCE IN COMPUTING CURRICULA

Elizabeth K. Hawthorne
According to the Computer Emergency Response Team (CERT) at the Software Engineering Institute (SEI) at Carnegie Mellon University, “nearly every facet of modern society depends heavily on highly complex software systems.

The business, energy, transportation, education, communication, government, and defense communities rely on software to function, and software is an intrinsic part of our personal lives.

Software assurance is an important discipline to ensure that software systems and services function dependably and are secure”.

Toward assured and dependably software in modern society, the ACM Committee for Computing Education in Community Colleges (CCECC) partnered with the SEI to produce *Software Assurance Curriculum Project Volume IV: Community College Education* [1].

This 2011 Technical Report (CMU/SEI-2011- TR-017), sponsored by the U.S. Department of Homeland Security (DHS) National Security Division (NCSD), includes a review of related curricula suitable to community colleges, outcomes and a body of knowledge, expected academic backgrounds of target audiences, and outlines of six undergraduate courses.
According to the American Association for Community Colleges [2], the mission of the community college sector is diverse: preparing students for transfer into four-year institutions, helping working adults prepare for new careers, as well as offering noncredit programs that offer a range of knowledge and skills.

The target audience for the six courses outlined in this curriculum is two-fold:

1) students planning to transfer into some type of baccalaureate degree program in computing, and
2) students with prior undergraduate technical degrees who wish to become more specialized in software assurance.

Assessment rubrics for these three introductory computer science courses were developed by the CCECC [3] and are also included in the report. Volume IV was also informed by three years (2009, 2010 and 2011) of working group reports from the annual ACM conference on Innovation and Technology in Computer Science Education (ITiCSE) [5, 6, and 7].
In developing the first volume [8] in this four-part curricular series, the term software assurance (SwA) was defined as the “Application of technologies and processes to achieve a required level of confidence that software systems and services function in the intended manner, are free from accidental or intentional vulnerabilities, provide security capabilities appropriate to the threat environment, and recover from intrusions and failures”.

The emphasis here is on the concomitant importance of both technologies and process in software assurance. This emphasis was carried through all the way to Volume IV, Community College Education [1].
An appropriate selection of courses for a specialty or concentration in Software Assurance (SwA) is recommended in this report as Computer Science I, II, and III and more specialized courses such as Introduction to Computer Security, Secure Coding, and Introduction to Assured Software Engineering.

These more specialized courses are not intended to be an exhaustive list of possibilities, but rather a set of courses that could reasonably be taken by students wishing to pursue further education in software assurance.
Community colleges could easily offer a certificate program in SwA complimentary to typical associate degrees in computer science, information systems and information technology.

This certificate option is part of the CCECC’s curriculum, assessment and pedagogy online repository – www.capspace.org. Brief descriptions of the six community college courses outlined in Volume IV are as follows.
Computer Science I: This course is the first in a three-course sequence that provides students with a foundation in computer science. Using a high level programming language, students develop fundamental programming skills, including secure coding awareness, human-computer interactions, and social responsibility.

Computer Science II: This course is the second in a three-course sequence that provides students with a foundation in computer science. Using a high level programming language, students develop intermediate programming skills with an emphasis on algorithms, software development, and secure coding techniques, as well as gain an appreciation for ethical conduct.
Computer Science III: This course is the third in a three-course sequence that provides students with a foundation in computer science. Using a high level programming language, students continue to develop programming skills focusing on data structures, algorithmic analysis, software engineering and software assurance principles, as well as giving emphasis to professionalism.
Introduction to Computer Security: This course provides an overview of the fundamentals of computer security. Topics include security standards, policies, and best practices; principles, mechanisms, and implementation of computer security and data protection; security policy, encryption, and authentication; access control and integrity models and mechanisms; network security; secure systems; programming and vulnerabilities analysis; principles of ethical and professional behavior; regulatory compliance and legal issues; information assurance; risk management and threat assessment; business continuity and disaster recovery planning; and security across the life cycle.
Secure Coding: This course covers security vulnerabilities of programming in weakly typed languages like C and in more modern languages like Java. Common weaknesses exploited by attackers are discussed, as well as mitigation strategies to prevent those weaknesses. Students practice programming and analysis of software systems through testing and static analysis. Topics covered include methods for preventing unauthorized access or manipulation of data, input validation and user authentication, memory management issues related to overflow and corruption, misuse of strings and pointers, and inter-process communication vulnerabilities.
Introduction to Assured Software Engineering: This course covers the basic principles and concepts of assured software engineering; system requirements; secure programming in the large; modeling and testing; object-oriented analysis and design using the unified modeling language (UML); design patterns; frameworks and application programming interfaces (APIs); client-server architecture; user interface technology; and the analysis, design and programming of extensible software systems.

Next slide show two possible sequencing options for the six courses given the suggested pre-requisite structure. Other options are also possible to meet local program needs.
### Option 1 for typical course sequencing

<table>
<thead>
<tr>
<th>Term 1</th>
<th>Term 2</th>
<th>Term 3</th>
<th>Term 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS I</td>
<td>CS II</td>
<td>CS III</td>
<td>Secure Coding</td>
</tr>
<tr>
<td>Discrete Structures</td>
<td>Calculus I</td>
<td>Assured Software Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Introduction to Computer Security</td>
<td></td>
</tr>
</tbody>
</table>

### Option 2 for typical course sequencing

<table>
<thead>
<tr>
<th>Term 1</th>
<th>Term 2</th>
<th>Term 3</th>
<th>Term 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS I</td>
<td>CS II</td>
<td>CS III</td>
<td>Secure Coding</td>
</tr>
<tr>
<td>Discrete Structures</td>
<td>Calculus I</td>
<td>Introduction to Computer Security</td>
<td>Assured Software Engineering</td>
</tr>
</tbody>
</table>
It is important to note that the Computer Science I-II-III course sequence, typical at community colleges as well as smaller four-year colleges, is the equivalent of the Computer Science I-II course sequence at other four-year colleges and universities.

The depth of coverage of the topics in each course varies, as do the associated level of Bloom’s Taxonomy.

In many areas, students need to be able to discuss and describe the topics, but in other areas they must be able to apply the techniques learned in the course to actual software projects.
As part of the outreach efforts, a well-attended Birds-of-a-Feather roundtable discussion was held at the 2012 SIGCSE conference.

Volume IV along with previous volumes of this curricular project – Volume I: Master of Software Assurance Reference Curriculum [8], Volume II: Undergraduate Course Outlines [9], and Volume III: Software Assurance Course Syllabi [10] – were disseminated at SIGCSE 2012.
PRACTICAL AND EXPERIMENTAL APPROACHES TO INFORMATION SECURITY EDUCATION

Seventh Workshop on Education in Computer Security (WECS7)
COMPUTER FORENSICS IN THE CLASSROOM

Chris Eagle
Naval Postgraduate School
Digital forensics is one of the fastest growing areas within the computer security field today. Digital devices continue to penetrate every aspect of our lives and digital crimes continue to become more and more sophisticated.

The resulting demand for trained computer forensics analysts presents unique challenges to a computer science degree program. The computer forensics process is traditionally divided into five phases: preparation, incident identification/response, evidence collection, evidence analysis, and presentation of findings. The hard “computer science” within this process lies primarily in the aforementioned evidence collection and analysis phases, yet it would be a disservice to our students to avoid the three remaining phases. In this talk I will discuss the challenges of presenting a one quarter course in computer forensics that is both sufficiently broad to cover all of the requisite phases and sufficiently deep to provide the student with a solid scientific foundation in the field.
A Report on Experience and Learning

Shiva Azadegan, Michael O’Leary, Alexander Wijesinha, and Marius Zimand

Towson University
ABSTRACT

- Describe experiences from the first three years they have offered their track in computer security for their computer science major.
- Present the details of the track, including descriptions of the courses they have offered.
- Discuss the lessons they have learned offering the track, as well as the challenges that remain.
1. INTRODUCTION

- In Fall 2002 at Towson University, Computer Science Department has offered their first course in their new Computer Security Track in the Computer Science major, and last year they graduated their second class of students.

- Their track in computer security is their traditional Computer Science major where their students choose specific courses in computer security for their upper level electives.
The computer security portion of the track is centered on the following seven courses:
- Computer Ethics,
- Introduction to Information Security,
- Introduction to Cryptography,
- Network Security,
- Application Software Security,
- Operating Systems Security, and
By design, their track focuses on the technical, practical, and applied areas of computer security.

Students still take the core computer science courses like Computer Architecture, Operating Systems, and Database Management.

The track courses are actually built upon the core courses. Figure 1 depicts the prerequisite tree for these courses.
Figure 1. Computer Security Track Prerequisite Tree
Enrollment in track has increased each year.

Enrollment in Cryptography course has increased from 15 in the track's first year, to 17 in the second to 21 last year.

Enrollment in Case Studies course was 5 in 2004, and nearly doubled to 9 in 2005;

Graduates from track have been in great demand, and former students have had little difficulty finding industry jobs quite quickly.

Employers have been so pleased with graduates, that they have contacted the faculty here directly, asking to help them recruit more of graduates.
2. FEATURES CONTRIBUTED TO THE SUCCESS OF THE TRACK

- Based on course evaluations and feedback from some of the employers of students, the strong hands-on component of the security track has been the most valuable learning experience for students.

- The courses in the track are built upon the core courses in the computer science program and they all have upper-level computer science courses as their prerequisite. Thus, students do have the theoretical knowledge, the maturity and good programming skills that are necessary to do elaborate and interesting applied projects.

- All of the courses in the computer security track used dedicated computer security laboratory for lab exercises.
3. THE EXPERIENCES & LEARNING

- This section share their experiences and learning and discuss the changes that were consequently made to the track.
The Cryptography course is structured into three modules:
1) symmetric cryptography,
2) public-key cryptography,
3) protocols and other applications.

The required textbook is *Introduction to Cryptography with Coding Theory by Wade Trappe and Lawrence C. Washington* [23]

The course has been offered four times in the Fall semesters from 2002 to 2005 and the enrollment has grown from 12 to 21 students.
Within the first module, some classical cryptosystems are covered such as: shift, affine, substitution, Vigenere, Hill, one-time pad ciphers (the module also discusses the most commonly used modern cryptosystems, DES, with its variants, and AES).

This section of the course is also used to gently introduce some mathematical techniques, such as modular arithmetic and some notions from probability theory.
Within the second module, they consider important that students understand and see the necessity of using rigorous mathematical formalism for the definition of security and secrecy as opposed to an ad-hoc and intuitive approach that does not provide any guarantees and is flatly hazardous in the context of cryptography.

The last module of the course on protocols and applications (such as bit commitment, zero knowledge proofs, secret sharing, digital cash, electronic voting, coin flipping by telephone, etc.).
The assignments fall into three large categories:
(a) math exercises needed to fix the notions and to develop mathematical skills,
(b) concrete attacks on “small” implementations of the crypto systems and crypto protocols covered in class, and
(c) exercises in which students are asked to analyze variations of crypto systems and crypto protocols and to reveal the weaknesses of the proposed variations.

The textbook is a good source for exercises in the categories (a) and (c).
The list of suggested projects takes into account that the course audience includes students that have little or zero programming experience. Thus the list includes:

(a) several programming projects (such as the implementation of differential attack on a small version of DES, RSA, different signature schemes, etc.),

(b) projects that involve reading recent research articles and writing a survey paper, and

(c) projects that ask the student to design protocols for some cryptography functionality using concrete “real-world” objects such as boxes with different kind of locks, pebbles, etc.
This course covers the principles of network security, focusing on specific application areas such as authentication (Kerberos and X.509 certificates), email security (PGP and S/MIME), IP security (IPSec), transport layer security (TLS/SSL), and firewalls.

The course begins with a general overview of common attacks and security mechanisms and services for attack prevention and detection, followed by a two-week introduction to basic cryptography.

The required textbook is Network Security Essentials by William Stallings.
Students working in groups are responsible for a paper/presentation and completion of 3-4 assignments. The group size depends on enrollments; there are typically 3-4 members per group.

In determining the course grade, the four components midterm, final, paper/presentation and assignments have equal weight.

The course has been offered 6 times from Spring 2003 through Fall 2005, once as an independent study, and has seen a maximum enrollment of 10 students.

The material covered in the course is fairly detailed and the objective is to provide students with an understanding of the various methodologies and security protocols employed in network security.

Exam questions have included a combination of short answer, multiple choice and problems.
The instructor assigns a topic for the paper to each group.

Topics have included wireless LAN (802.11) security, cellular network security, denial of service attacks, and IP, ICMP and TCP vulnerabilities.

The paper is written in a formal conference style requiring 4-5 pages in two column format.

While the paper is not expected to be overly technical, it is to be written at about the same level of detail as topics covered in the course text.

The paper also provides an opportunity to assign topics in network security that are of current interest and address new developments in the field.
The assignments constitute the most interesting part of the course.

For their first assignment, students use a socket program to implement the Diffie-Hellman exchange and transfer a message over a network encrypted using AES.

They could write their own code, or use prewritten free software and tools available on the Web.

Other assignments include using GPG and its trust model to transmit secure email, and using Snort to log packets, write rules and trigger alerts based on network protocols and message characteristics.
Overall, the course has been quite successful judging from end-of-semester student evaluations.

Assignments that address IP/TCP vulnerabilities, IPSec, SSL/TLS and wireless security would be beneficial.
3.3 APPLICATION SOFTWARE SECURITY

- The Application Software Security course introduces students to the security concepts in developing software applications.
- This course discusses design principles for secure software development, and some of the security issues in current programming and scripting languages, database systems and web servers.
- This course is the only one that has been taught by more than one instructor. Three instructors with approaches varying from small number of hands-on projects and strong current literature research projects, to equal time allocated to lab and lecture, to completely lab-based, starting with the overview of Assembly language and run-time environment, have taught the course.
Some of the topics and projects covered in this class include:

**Buffer overflow** - Students are presented with an in-depth discussion of stack and heap overflow problems and how to exploits are generated. The reading assignments include papers addressing the overflow problem, integer and format string overflow problems, methods of defense against buffer overflow and secure programming techniques to prevent buffer overflow. In the lab, students work with relatively simple programs with buffer overflow vulnerability and asked to write the exploits.
Threat modeling - There is a good coverage of this topic in “writing secure code” [12]. Microsoft has developed a tool [2] that is freely available and can be used to make the projects on this topic more interesting and challenging. Threat Modeling [20] provides a good reference for this project.

Authentication and authorization - students provided with a broad overview of authentication, authorization and access control techniques. Java security model is presented and students’ projects deal with using JAAS. Also they discussed Kerberos in details and a simple project dealing with configuring a Kerberos server in our security lab and using it to authenticate users for their projects.

Cross-site scripting and SQL injection - Students enjoyed the discussion of both topics. Students did team projects demonstrating these problems.
The operating systems security course introduces students to operating systems security issues and how to secure a system, and it has Operating Systems as a prerequisite.

They allocated equal time to Unix and Windows systems.

However, they spend more time discussing Unix system than Windows.
- First, students are less familiar with the Unix environment than Windows.
- Second, there are much better textbooks, articles and resources available addressing Unix security vulnerabilities than those for Windows.
The coursework for this class consists of literature review, projects and exams.

The students were assigned a paper each week to read and had to write a one-page review. Examples include “Security Report: Windows vs. Linux”.

For the exercises and projects in this class, students had root or administrator access to their virtual machine.

There are 7-8 projects small team projects in this class in addition to the final term project that depends on its scope, and can be either an individual or a team project.

The projects dealing with Unix environment include writing a password cracker program; creating and managing user accounts; installing, configuring and using Tripwire; configuring a simple secure ftp site and creating “jails”; and hardening Linux project using Bastille.
The Case Studies in Computer Security course serves as the capstone experience for the security track. Only offered in the spring semester, it has Operating Systems Security and Network Security as prerequisites. This course has been offered twice, in Spring 2004 and Spring 2005 to five and nine undergraduate students. It is a hands-on course that emphasizes defense, detection, and administration, and is arranged around a sequence of five or six competitive team based laboratory exercises in the security laboratory. In a typical exercise, four different teams of students design and construct their own network of machines.
Students can choose from a range of operating systems, but their resulting network is required to provide a suite of remote services like Web, SSH, or FTP to their competing teams in the laboratory.

Each team is then provided with authentication credentials to one or more of the services offered by some of the other teams, with the conditions that
- No team has root equivalent credentials on any machine from another team,
- No team has all of the non-root credentials for any other team, and
- No team knows which other teams have credentials for the services that they are to provide.
During the live portion of the exercise, each team must verify that the services provided by opposing teams for which they have authentication credentials are correctly functioning.

They then try to illicitly gain access to all of the remaining services and the remote host itself, if possible.

Once the live portion of the exercise has completed, students review their logs to try to determine who accessed their network, and whether they did so legitimately or illegitimately.
Although attacks, and attack tools are described, the majority of the class time is spent learning defensive skills. Topics covered in the class include:

- Account and password management. PAM, password cracking.
- Logging and Auditing. Setting up a log server.
- Simple reconnaissance techniques; ping, nmap.
- Packet sniffers; Ethereal.
- Intrusion detection systems; Snort.
- Configuring common services: IIS, Apache, OpenSSH, WU-FTP.
- Advanced reconnaissance: Null connections and NetBIOS enumeration, SNMP walking.
- Backdoors: netcat, vnc.
- Firewalls. Iptables.
- Security analysis tools: Nessus, Microsoft baseline security analyzer.
- Security configuration tools: Bastille, Microsoft IIS lockdown tool.
Like all new tracks, this security track has had its share of administrative challenges.

Scheduling courses needed for completion of the track has presented some difficulties.

At the paper time, they were offering courses according to the following plan:

- Fall: Cryptography, Network Security, Operating System Security
- Spring: Application Software Security, Case Studies

Summer Workshop on Cyber Security
August 12-16, 2013 - Security Education - TTU
In the past, however, Operating Systems Security, Application Software Security and Network Security have been offered in both Fall and Spring. This has been due to several reasons including newness of the track, prerequisites, student demand, expediting graduation, and faculty availability.

Another factor that has impacted scheduling in the past is the offering of combined undergraduate and graduate classes. During the first few semesters the track was offered, some classes had few students. This was to be expected until such time that new students in the track would become juniors or seniors and start taking the security courses.
At present, there are only five faculty members teaching the security courses.

All are tenured or tenure-track; one has a joint appointment in Mathematics and Computer Science; the others are in Computer Science. With the exception of Application Software Security, the same faculty member has always taught every offering of a given course.

So far, this has worked out well, since the courses have matched the interests and expertise of the faculty.

It also allowed courses to be fine-tuned and improved with each offering, and provided continuity.

However, this approach presents difficulties in that faculty availability and track schedules may not always match.

Although the department currently has about 24 tenured/tenure-track faculty members, most have interests in other areas of Computer Science and Information Systems, which means there is little flexibility in assigning instructors to security courses.
Currently, they are considering a number of modifications and improvements to the track. They include:

- Changing the prerequisite tree for the track. In particular, they are considering removing the prerequisite course in Cryptography from the Network Security course; this will reduce the size of the (lengthy) chain of courses that our students need to graduate. As it stands, Cryptography is a prerequisite for Network Security, which is a prerequisite for Case Studies in Computer Security; this long chain of 300 and 400 level courses has been a difficulty for students.
- they are assessing the contribution of the introductory course, Introduction to Information Systems Security, to the whole track. This course provides a broad overview of technical and human components of information systems security. Therefore, there is a considerable overlap and redundancy between this course and the remaining courses in the track they are considering the possibility of reusing the credit hours to design a new course.

- They may want to designate one of the sections of computer ethics course for the security track students to specifically address some of the security policies and regulations. Right now, the Computer Ethics course is a general course for all of our majors, and we may want to use this course to emphasize the ethical issues that arise in computer security.

- They are trying to forge closer connections between the different courses. Some topics occur naturally in different courses; for example firewalls and intrusion detection systems are described in both the Network Security course and the Case Studies course. We would are trying to ensure that the topic is covered in a complementary fashion in the two courses.
INFORMATION SECURITY ATTACK
TREE MODELING

An Effective Approach for Enhancing Student Learning
Jide B. Odubiyi and Casey W. O’Brien
Department of Computer Science, Bowie State University, Bowie
Network Technology Department, Community College of Baltimore
County, Baltimore
This paper presents a framework for enhancing student learning about the vulnerabilities of information assets of a business enterprise using attack tree modeling. Using this framework, students get an overview of the methodology as well as learn how to implement it with a well-known list of information security vulnerabilities. As a result, students can provide input into threat modeling strategies and operating procedures and thus, increase overall confidentiality, integrity, and availability of computer and network systems. The paper concludes by describing a corresponding system capability metric that a system administrator, student, or red team can use to test the vulnerabilities of the systems within a business enterprise.
1. INTRODUCTION

- For years, engineers have relied on the analysis of failure data to improve their designs by employing the engineering principle of Failure Mode and Effect Analysis (FMEA).
- The main goal of this principle is to identify risks and initiate concerted efforts to control or minimize these risks. By knowing the risks, the project plan can be created more realistically.
- FMEA techniques [1] help to identify failure potential in a design or process before it actually happens.
- This is done by following a procedure which helps to identify design features or process operations that could fail. This procedure requires the identification of all potential failures, including the causes and the effects of those failures.
- Using knowledge of the design, process, and/or system, estimates of the probability of these failures can be identified, together with the severity of the effects of the failure, and the probability of detecting the failure before it becomes critical.
While engineers have historically been good at using failure data to improve their designs and systems, software engineers and information systems administrators have not.

In general, IT professionals dealing with the security of systems do not use similar failure data – attack data – to improve their computer and network systems and the related components that are a part of those systems.

The reasons for this vary from organizations that are weary of divulging such information (for fear of decreased public confidence), to fears that attackers will employ the attacks against their systems, to a lack of detailed and reliable data on attacks [2].
Despite weariness on the part of organizations to disclose attacks on their systems, attack data have become more available in recent years, due in large part to the public and media's increased attention on information security issues.

In addition to the public and media interest in information security-related issues, other resources like the SANS Internet Storm Center [3] and Security Focus [4], as well as many Computer Emergency Readiness Teams (CERTs) and Coordination Centers (CCs) throughout the world bring together timely and comprehensive security-related information.

These resources serve as a forum for in-depth discussions and announcements of computer and network security vulnerabilities. They detail what they are, how to exploit them, and how to fix them.
Countless articles and research publications have been written detailing threat models that can be used to test the security of an enterprise system. The authors of these publications argue that understanding all of the different ways a system can be attacked can help IT professionals responsible for securing systems design and implement countermeasures, to thwart those attacks. In addition, if these same IT professionals can understand who the attackers are and what their abilities, motivations, and goals are, they can implement the proper countermeasures to deal with the real threats [5].
A fundamental understanding of threat modeling can help system developers/integrators build robust and reliable systems. If they know the possible threats, they can use an attack tree to test their systems. For example, after building a Web site, a system administrator can readily apply a threat model to test it. The same is true of students learning about the vulnerabilities of the information assets of a business enterprise. By developing several attack trees (which constitute an attack forest of a business enterprise), students get an appreciation for the hundreds of possible vulnerabilities in a given enterprise system, and the challenges faced by system administrators when securing their computer and network systems. Ultimately, students would then use this model to further test the security of their systems. Several organizations, including The General Accounting Office and the U.S. Department of Homeland Security, employ Red Teams to identify vulnerabilities in their information assets [6].
2. TEACHING THREAT MODELING

- Students cannot build and test the security of computer and network systems unless they understand the threats posed to those systems.
- Therefore, it’s critical that students in Information Assurance (IA), Computer Science, network technology, and other IT related programs be taught not only how to design and build secure systems, but also how to identify the means, motives, and opportunities of their adversaries, identify the threats posed to the systems they are securing, and ways to test it.
- When the above mentioned academic programs are infused with threat modeling concepts and implementation strategies for testing the security of systems, students can do more.
- They can extend the application of this framework to design systems that meet the security objectives of an organization, decide on trade-offs during key design decisions, and help reduce the risks of security-related issues arising during the implementation and operations phases.
Example of how to introduced and enhanced student’s understanding of threat modeling: (what they did)

1. Asked students to compile a list of information assets managed by the university or a typical medium business enterprise. The list of information assets includes several Web servers and Database Management Systems for Payroll, Strategic Plans, etc.

2. Provided the students with 32 information system vulnerabilities including the Twenty Most Critical Internet Security Vulnerabilities [11] and asked them to develop attack trees for them.

By developing several attack trees which constitute an attack forest of a business enterprise, students moved from a conceptual grasp of threat modeling, to an appreciation for the hundreds of possible threats to a system, to the challenges faced by system administrators, to methodologies for testing the security of a system.
3.1 THE ADVERSARY MODEL

- Before students are able to design, build, and implement secure systems, they must understand the means, motives, and opportunities of their adversaries [5, 9].

- Salter, et al [9] developed an Adversary Model to help organizations thoroughly understand their adversaries and characteristics.

- This model characterizes adversaries in terms of their resources, access, risk tolerance, and objectives.

- The learning objective for students is that counter measures are only needed for attacks that meet the adversaries' resources and objectives, but to be useful, the countermeasures must also meet the organization's needs for cost, ease of use, compatibility, performance, and availability.
3.2 IDENTIFYING AND CLASSIFYING ASSETS

- The task of identifying assets that need to be protected within an organization is one of the less glamorous aspects of information assurance.

- Unless an organization knows its assets, where they are located, and what their value is, it's difficult to decide on the amount of time, effort, and money that should be spent on securing them.

- In addition, organizations won't know what the adversaries of their systems want until they've identified the sensitive information and related assets on those systems.

- Students should explore asset classification models, which can be categorized as follows:
  - the identification of an organization’s assets,
  - the accountability of those assets,
  - preparing a framework for classifying those assets, and
  - implementing the classification framework.
3.2.1 THE ENHANCED TELECOM OPERATIONS MAP FOR THE TELECOM SERVICE INDUSTRY

- Figure 1 can serve as an excellent resource to give the learner an appreciation for the processes required for managing and protecting a telecommunication business enterprise.
- The enhanced Telecom Operations Map (eTOM) has been developed by The Telecom Management Forum as a generic framework for organizing a telecommunication business organization into three major areas.

- The three areas consist of
  (a) Planning and Lifecycle Management of the Strategy, Infrastructure, and Product development;
  (b) Operations Management, and
  (c) Enterprise Management—Business Support management.
Each of the three areas consists of computer systems, telecommunication networks and software that must be protected.

The responsibility for protecting these assets falls on the shoulder of the chief security officer or the Information Technology (IT) department.

As a result of mergers and acquisitions, there can be tens of software products just for the Billing operation.

At one time, the former WorldCom had over 30 billing systems. Each billing system consisted of database management systems, Web servers, and accounting software.

Other operations support systems (OSSs) consist of diverse software products.

The eTOM framework provides an environment for students to identify OSSs to analyze for vulnerabilities.
Figure 1. The Enhanced Telecom Operations Map® (eTom) (Source: The Telemanagement Forum) [15]
3.3 Creating an Architectural Overview

- When students are creating an architectural overview of an enterprise system, they need to be explicit about:
  - what the network and related systems are designed to do,
  - how they plan on engineering it to achieve that functionality, and
  - what technologies are required to implement the design.

- This helps them identify the common technology-specific threats and implement solutions to overcome them [7].
- In addition, having students create a diagram of how information flows throughout the organization allows them to discover the critical components and procedures of the enterprise system.
- The student can use the eTOM model in Figure 1 or the instructor may identify a similar map for another industry.
3.4 THE SUM OF ITS PARTS

- Once students have an overview of the architecture of the enterprise, they can use it to break down the system into its constituent components.
- The more an organization knows about its system, the easier it is to discover threats against it.
- The steps involved in breaking down a system into its constituent parts starts with creating a security profile, that is, an analysis of the system’s strengths and weaknesses.
- Students will need to validate all data being sent across their systems. This includes:
  - a comprehensive examination of the trust boundaries of the enterprise network,
  - the flow of data in and out of the enterprise, and
  - any entry points into the system that will ensure that the flow of information is done in a secure manner [8].
3.5 IDENTIFYING THREATS USING ATTACK TREES

- The first step is to identify the threats that might affect an organization’s system and potentially compromise its assets. This includes identifying network, host, and application threats.

- Network threats can be assessed by having students investigate how the data passes through routers, firewalls, switches, and other network devices.

  - Students need to understand the logic and syntax of these device’s configuration files.
  - In addition, students need to be able to determine what it takes to get past or compromise each device. Host investigations should include common configuration categories applicable to all server and operating system resources (patches, files/directories, ACLs).
  - Finally, students should examine the enterprise’s application software [7].
3.6 ATTACK TREES AND THE SANS TOP-20 VULNERABILITIES

- An attack tree provides a method for representing attacks (and similar vulnerabilities) on a system in the structure of a tree.
- The goal of the tree is the root node. The leaf nodes represent different paths to achieve the goal.
- As depicted in Figure 1, a business enterprise typically consists of hundreds of information assets that are vulnerable to attacks.
- Each of these assets can be modeled as an attack tree resulting in an attack forest [5].
- The root of each tree in the attack forest constitutes an event that can potentially damage the enterprise system and its related resource’s confidentiality, integrity, and availability.
- The SANS twenty most critical Internet security vulnerabilities provide a rich environment for the students to learn about well documented attacks [11].
- The SANS top-20 2004 vulnerabilities consist of two top-10 vulnerable services in Windows and UNIX operating systems. The SANS top-20 list is updated annually and provides useful information about each vulnerability, systems affected, and how to protect the services. Students can use the protection measures to develop attack scenarios.
3.6.1 THE MAIL TRANSPORT SERVICE AND ENTERPRISE SERVICE (NIS/NFS) VULNERABILITIES

- Two of the twenty critical vulnerabilities identified by SANS are the Mail Transport Service (MTS) and the misconfigured enterprise services: Network File System (NFS) and Network Information Service (NIS).

- The Simple Mail Transfer Protocol (SMTP) is one of the oldest Internet application protocols that employ Mail Transport Agents (MTAs) as servers to send emails from senders to recipients.
Examples of these MTAs are sendmail, QMail, Courier-MTA, Postfix, and Exim.

A. They are vulnerable to buffer/heap overflow attacks when they are not patched or when their patches are out of date.

B. An attacker can compromise the Mail Transport Service by exploiting open relays (i.e., a situation that permits the sender and receiver that are not part of the domain to send and receive mail).

C. A third major vulnerability of the MTS is exploitation of non-relay problems such as misconfiguration of the user account database, thereby exposing the service to spam attacks.
The NFS and NIS services are used in UNIX servers to hold directories of file systems and to provide locations of distributed databases respectively.

They are vulnerable to buffer overflow attacks due to unpatched services, Denial of Service (DoS) attacks and weak authentication.
3.7 Modeling An Attack Tree

- The Web server is one of the SANS top-20 most critical vulnerabilities due to add-on software modules such as CGI scripts, PHP bugs, servelets, etc.
- The process for modeling an attack tree for a Web server’s vulnerability is described in this paragraph.

Since each tree has a root node that represents the attacker’s goal, and the leaf nodes represent different paths to the root, each child node represents the steps an attacker can take. Modeling the attack tree involves associating a logical AND and a logical OR with each node. In essence, a node of an attack tree can be decomposed into an AND or an OR node. An AND node or an OR node decomposition can be represented in graphical or textual formats.
- Figure 2 is a textual representation of one of the top vulnerabilities to Windows-based systems [11]:

GOAL: (G0) Gain Privileged Access to a Web Server Using a Known Vulnerability

AND  G1. Identify organization’s domain name.

G2. Identify organization’s firewall IP address

   OR  1. Interrogate domain name server

   2. Scan for firewall identification

   3. Trace route through firewall to Web server

G3. Determine organization’s firewall access control

   OR  1. Search for specific default listening ports

   2. Scan ports broadly for any listening port

G4. Identify organization’s Web server operating system and type

   OR  1. Scan OS services’ banners for OS identification

   2. Probe TCP/IP stack for OS characteristic information

G5. Exploit organization’s Web server vulnerabilities

   OR  1. Access sensitive shared intranet resources directly

   2. Access sensitive data from privileged account on Web server

Figure 2. Web Server Attack Description
As presented in Figure 2 above and stated earlier, a node of an attack tree can be decomposed into an AND or an OR node.

Both the AND and the OR decompositions can be represented in graphical or textual format as shown in Figures 3 and 4 below. All the attack sub-goals (such as $G_1, G_2, G_3, \ldots, G_5$) must be achieved for the exploit to succeed.
Figure 3. A Graphical Representation of an Attack Tree described in Figure 2

\(G_0 \equiv G_1 \cap G_2 \cap G_3 \cap G_4 \cap G_5\); \(G_2 \equiv G_{21} \parallel G_{22} \parallel G_{23}\)
The attack tree in Figure 3 can generate 24 attack or intrusion scenarios [10] in Figure 4. The 24 possible attack scenarios are:

- The realization that a simple Web server attack tree can generate 24 attack scenarios helps the student to understand the importance of developing secure software as a countermeasure against Web server exploits.
Figure 4 Twenty-four possible attack scenarios
3.8 BENEFITS OF DOCUMENTING THE THREATS: FROM AN ATTACK TREE TO AN ATTACK FOREST

- Figure 3 is one of hundreds of vulnerabilities that make up an organization’s attack forest.
- By creating an attack forest:
  - an organization has a roadmap for testing for both known and unknown vulnerabilities,
  - an organization has also has a means to document the threats to their systems and assets.
- the attack forest document for an organization will include:
  - each threat;
  - a description of the threat;
  - the target of the attack;
  - the risk of the attack;
  - the techniques likely to be used in carrying out the attack; and a risk management strategy.
For example, when dealing with a SQL injection attack, the target is the database and the technique is that the attacker types a command into a textbox that is automatically added into a T-SQL command without client-side validation. To counter this threat, one can use regular expressions to validate the user name, and use a parameterized query to access the database [12].
4. CHALLENGES

Teaching threat modeling can be difficult for a number of reasons, each of which is described below.

4.1 From Past to Present

Organizations trying to protect their assets from attackers and current intrusion detection tools face a common problem: Being able to generalize from previously observed behavior to recognize future behavior, either malicious or normal.

Signature-based misuse detection techniques are acutely prone to this problem, as are anomaly-based detection tools that must determine normal behavior that is not identical to past observed behaviors, in order to reduce false alarms [13].

By understanding this problem, students can use both advances in intrusion detection techniques and adversary models as ways to better understand who their attackers are, and ultimately, implement the proper countermeasures to deal with the threats.
4.2 Input Data: Developing Attack Trees

As stated earlier, there are multiple ways students can identify the threats to a given enterprise system.

Encourage using a combination of known threats [11] as the basis for developing an enterprise attack forest.

When combining known lists of vulnerabilities with attack trees and attack trees’ patterns, new vulnerabilities can be identified.
ADDING WHEN, WHERE, AND WHY TO HOW

Scott J. Roberts and Andrew L. Reifers
School of Information Sciences and Technology, Penn State University - University Park,
NSA Center for Information Assurance Excellence
Educators recognize that laboratory based computer security courses do far more for students' understanding than purely theoretical courses.

Laboratories with common hacker tools are invaluable for students in an Information Assurance curriculum. These tools help students better understand threats and the defense mechanisms needed to protect individual systems and entire networks from these attacks.

Students greatly benefit from understanding the threats they are called to defend against, making them more effective protectors of enterprise or government networks.

Conversely these labs offer students hands on experience with tools that could easily be turned and used against others.

Through the use of situations that show ethical uses of these techniques students can have these beneficial experiences without becoming the very danger they are being trained to protect against.
1. INTRODUCTION

- With the growth of networked businesses and organizations many positive elements of society have branched out and begun to flourish into cyberspace. Commerce has fully embraced the Internet, and e-commerce is becoming more mainstream everyday.

- With nearly all information and an ever-increasing amount of financial transactions taking place online societies negative elements are sure to follow.

- Crime goes where the money is and thus the Internet has quickly given birth to a new breed of criminals.

- The Internet has, to a limited extent, become the new Wild West: lucrative, expanding, and largely lawless.
Teaching a student tools that enumerates vulnerabilities to be patched shows them how to enumerate vulnerable machines for exploitation.

Although dangerous when misused, these tools are still of great use to an information security professional, when used in the right context.

Information Assurance professionals should know when, and where to use of techniques and tools they possess.
2. CURRENT CURRICULUM

- At Penn State University much work has already been done to provide the best possible learning experience for future Information Assurance Analysts, Engineers, and Managers.
- Inside the School of Information Science and Technology, the Information Assurance program works to educate these future leaders in many aspects of computer security.
  - All Information Assurance students are required to take the standard set of introductory IST courses, including an Introduction to Networking, an Introduction to Databases, and two basic programming courses.
  - Next, All Information Assurance Track students are required to take the introductory technical course, Network Security, as well as the introductory policy and ethics course, the Legal and Regulatory Environment of Privacy and Security.
  - In addition each student is required to take one additional Information Assurance related course of their choosing, such as Wireless Network Design and Security, Web Application and Database Security, or in some cases an independent study with an Information Assurance track professor.
Laboratory requirements vary from class to class, though typically, as would be expected, the technical courses have a lab component. These laboratory times are conducted outside of class, and are generally dependent on the current lecture track, though this is not always the case.

Students in teams of 5 to 8 people work together through a technical lab proctored by a Teacher’s Assistant (TA). These labs cover both malicious and defensive tools, giving students a basic overview of their setup and operation. Afterwards students are given a homework sheet with a few basic questions about the technology used and its operation.

The current Information Assurance lab curriculum includes two weeks studying the attack tools Metasploit Framework and L0pht Crack Password Cracker. The other eight weeks of lab detail use of a piece of standard Cisco network defense appliances and technology. Weeks three and four teach the use and maintenance of a Cisco router, weeks five and six are spent learning the Cisco Pix firewall, seven and eight are a Cisco Intrusion Detection System, and the last two weeks round out the series studying a Cisco Virtual Personal Network appliance. All these labs were Cisco approved lab sessions also taught by a graduate student TA.
One of the biggest problems in any undergraduate Information Security/Assurance curriculum is balancing the many elements that need to be taught [5].

Computer security is a constantly evolving and changing world, defenders are being pressed to understand and deal with new threats, old threats revisited, and old threats subtly changed.

At this point it becomes a struggle to keep up to date and decide what new issues need to be brought to the students attention and what can simply be left for them to discover once they reach their places of employment.

On top of this is the difficulty of pacing a course correctly.
Computer Security and Information Assurance cover a huge realm, from programming to policy, network administration to system configuration, and now spyware to spam [5].

It’s difficult enough for most classes to get through the defensive basics and how to develop security policy.

It is often impossible to even begin the most basic overview of attacks such as buffer overflows, configuration exploitation, spyware, spam, and the many other challenges facing individuals, corporations, and governments.
It’s quite understandable that defensive technology and the policy of running security in a corporate security is the priority of academic network security courses.

The vast majority of Information Assurance students will be working in defensive roles, protecting network infrastructures, not attacking other networks.

With the exception of government run and funded Information Warfare, corporate penetration testing, and underground hacking competitions such as Root-Fu or Capture the Flag,
Very few students will ever put into practice the attacks they’re being taught to defend against [6].

This emphasis on defense while largely ignoring the intricacies of attack may be most practical in the short-term view it hurts students in the long run.

When Penn State Network Security students were asked the question:
“Do you feel an understanding of techniques a malicious hacker would use will benefit you if you go to work in the Information Assurance field?”

They unanimously agreed that understanding hacker techniques would further their ability to properly secure information.
The same survey also brought to light that a majority of the students actually took personal time after the conclusion of the attack lab sessions to do their own investigation into the tools taught.

With this in mind, combined with how little is actually taught about the intricacies of the attacks they eventually will be forced to defend against, students realize that their education omits an important issue of security.

In short, defenders can better defend when they have a firm, technical foundation into how the attackers will attack.
4. JUSTIFICATION AND DUE DILIGENCE

- With students themselves clamoring for experience in this area the question truly becomes is it necessary?
- courses should stimulate students' interest
- courses should also be applicable to the students after graduation.
It takes a unique blend of experience, knowledge, contacts, and luck to be able to identify, analyze, and advise on today’s threats.

This is combination that is difficult, if not impossible, to teach. [2] Students can however be given a framework from which to continue their learning, and for that it is necessary to teach lab exercises detailing the use of malicious tools.

Understanding the configuration of a device is simple, understanding the potential impact of an exploit or tool to an enterprise network is complicated.
There is a fine line between teaching students to understand a possible attack and giving them the knowledge to carry out such an attack [4].

With every hacker tool that is taught educators risk creating the very menace they are trying to teach students to defend against.

This paradox begs the question, how is it that these tools can be taught without simply giving the defenders more to defend against.

In order to properly educate students in both the concepts behind a tool and correct ethical use of the tool it takes more than simply demonstrating the tool, but also showing students how it can be used appropriately. [6]
5. PROPOSED SITUATIONAL LAB CURRICULUM

- As a proposed solution to this dilemma development They started on a new form of lab exercise to find a way to not simply teach the “how” of using a tool, but also the “when, where, and why”.

- These labs use situations, mostly through involving valid auditing practices where similar tools would be used by corporate or government security teams to test their various defense in an effort to straighten their current position, not to compromise a foreign network [2].
This provides a way to help students understand that a given tool, while inappropriate, and probably illegal, to use without permission, can be a valuable asset when trying to keep malicious attackers at bay.

This gives students a perspective to see that a tool is just a tool, and it is the user’s implementation of that tool which decides if it is used to improve and protect a given network, or attack and wreak havoc against an unsuspecting target.

These situational labs also provide a more real life experience than simply setting up a router and answering a few facts in a corporate style threat analysis report as was the case in former labs.
The beginning of the lab is a fictional situation, or story, into which the rest of the lab takes place [5].

This story details a time when a tool might be used in a valid security improvement context. It may be a password strength audit for a tool like L0pht Crack, or testing the practicality of a newly installed firewall for a tool like NMap, or possibly doing a network vulnerability audit for a tool like Nessus.
Students are asked to place themselves in the shoes of an Information Assurance Engineer at a fictional company and use these tools to find out how they can further protect the company.

After a brief set of open-ended instructions that encourage exploration and experimentation students are given a set of goals that force the students to properly utilize the tool in the laboratory network environment. These goals fit the context, and help guide the students in solving the problem posed in the context.
Students might have to use L0pht Crack to conduct their password strength audit to find what employees are failing to comply with company password policy.

Each context and problem is tailored to the tool being taught, giving students a perspective on “when, where, and why” to use each tool.
The L0pht Crack lab would have homework that helps students craft an audit report detailing the amounts of passwords that complied and did not comply with standard company password policy.

Furthermore students would create a list of users failing to comply and provide recommendations for remediation of these problems, as well as possible improvements to the policy as it stands.
After students progress through five such labs they are faced with a final capstone project before lab curriculum progresses to set of defensive lab with the same emphasis on real world context.

Continuing with the idea of context being essential to properly help students understand the use of these tools students are presented with an interesting set of problems in what is by far the most open ended lab.

Student teams are put in a situation where they are using the tools they have learned to conduct an open penetration test, given a target of significance on the network and must use the skills they developed using these various tools to take advantage of the vulnerable system.
The use of context and treating the lab like a real exercise are what gives students the “when, where, and why” that is lacking in traditional, procedure based labs.

While a step-by-step walk through of a tool might give a student the barest amount of understanding about how to use a tool it is only the application of that tool that makes it valid.

Further, in the corporate world, the use of a tool is only as valid as the results it generates, thus it is doubly important that reporting is an integral part of these lab exercises, reinforcing both the labs content, and giving students practice at producing these real world style reports.
6. CHALLENGES AND OPPORTUNITIES

- In creating a set of labs that mimics the context of real life security audits, realism is perhaps the most difficult, yet most necessary, characteristic to establish.

- Students must believe in the possibility of truly being faced with this type of challenge when they enter the working environment for them to fully apply themselves.

- As labs are continuously created they must constantly evaluate the realism and validity of the situation in which they are used, since simply teaching the tool without a realistic use situation defeats the entire idea of helping students understand the ethical use of these tools.

- Reporting templates must also follow the labs and uphold the continuity by making the reporting requirements accurate.
Realism in the lab environment, especially for the penetration testing exercise, is essential, making the environment setup incredibly complicated and requiring careful planning.

Care must be given to make sure that the objective can be accomplished using whatever group of tools is taught in a given semester.

Furthermore, multiple attack methodologies should be valid for accomplishing the given goal.

A true to life diversity of clients, servers, appliances, services, and operating systems should be represented.

This is complicated by the need to make sure vulnerable versions of operating systems and services are in place in the environment, and that valid attacks and exploits are made available to the auditors, either by pre-configuring their testing host box, or by making the appropriate files available on a resource server or disk.
The detailed reporting provides a challenge over past lab formats as TA’s are forced to not simply check objective, short answer questions, but instead to read and evaluate longer, detailed audit reports, where answers may vary greatly, but still be correct. This limits the ability for a professor to simply provide a key to the TA, and instead requires the TA to be extremely familiar and use their own judgment to evaluate the validity of various answers. Lab creators must provide clear guidelines that attempt to remove as much ambiguity as possible from their guidelines of answers and minimize the amount of subjective judgment that those grading the exercises must use.
7. CONCLUSION

Information Assurance educators are faced with a difficult task in the years ahead. With a constantly growing need for Information Assurance professionals and a continuously changing realm of security it is a struggle to maintain both the correct body of knowledge and ethical attitude. Teaching malicious tools provides an essential part of developing a holistic approach and understanding to security. By giving students an understanding of attack methodologies and by allowing them to utilize the same tools malicious attackers use, students develop the holistic understanding of security that they desperately need. By teaching these tools in a context where students can use them ethically they understand not only “how” an attack succeeds, but “when, where, and why” that same tool can be used to keep the bad guys out. Essentially participating in a class that teaches the “when, where, and why” is the sheriff’s badge of the new Wild West. This work was supported by NSF DUE-0416827, and by DoD IA Institutional Capacity Building at Penn State Grant.
THE CYBERCIEGE INFORMATION ASSURANCE VIRTUAL LABORATORY

Mike Thompson

Naval Postgraduate School
ABSTRACT:

- **CyberCIEGE** enhances information assurance education and training through the use of computer gaming techniques.

- In the CyberCIEGE virtual world, users spend virtual money to operate and defend their networks, and can watch the consequences of their choices, while under attack.

- These tutorial sessions will introduce CyberCIEGE to educators and information assurance training professionals. The IA training tool is provide to develop new scenarios.
The CyberCIEGE project creates an Information Assurance (IA) teaching/learning laboratory. In addition to rigorous scientific foundations, it involves the application of abstract principles to the real world. A hands-on virtual laboratory provides a dynamic and often surprising context where abstract principles can be applied and discovered.

CyberCIEGE is an innovative computer-based tool to teach network security concepts. The tool enhances information assurance education and training through the use of computer gaming techniques such as those employed in SimCity™ and RollerCoaster Tycoon®. In the CyberCIEGE virtual world, users spend virtual money to operate and defend their networks, and can watch the consequences of their choices, while under attack.

In its interactive environment, CyberCIEGE covers the significant aspects of network management and defense. Users purchase and configure workstations, servers, operating systems, applications, and network devices. They make tradeoffs and prioritization decisions as they struggle to maintain the ideal balance between budget, productivity, and security. In its longer scenarios, users advance through a series of stages and must protect increasingly valuable corporate assets against escalating attacks.
The CyberCIEGE encyclopedia of security concepts contains a wealth of information assurance knowledge. Users can read the encyclopedia, or watch its instructional movies!

CyberCIEGE supports many educational venues, from basic workforce awareness training to university classes. It can help organizations meet DoD Directive 8570 obligations for IA training, annual IA awareness refreshers, and appropriate IA education. CyberCIEGE contains support for the creation of tools to record and assess student progress. Best of all, CyberCIEGE is extensible.

CyberCIEGE includes a language for describing its security scenarios. Using this language, educators may construct or modify scenarios that can then be played by students. CyberCIEGE was created by the Center for Information Systems Security Studies and Research (CISR) at NPS, and Rivermind, Inc., of San Mateo, CA.
TEACHING ASSURANCE USING CHECKLISTS

Matt Bishop
University of California, Davis
Industry, the government, and many other entities have expressed concern about the way schools are teaching programming. The central theme is that students do not learn how to program "securely." Several proposals to improve the quality of programming involve checklists, or items that students must demonstrate mastery of in order to achieve the appropriate goal (such as passing the class or graduating).

Underlying these proposals is the goal of teaching assurance—building programs and systems that meet specific requirements—and making it a part of everyday work, as opposed to a specialized discipline applied only when there is time, or when there are specific assurance-related requirements. This raises several issues, especially when one is considering "security assurance."
What is “security?”

The standard answer, “security is whatever the security policy defines it to be,” does not provide the guidance needed to develop a new student’s intuition about how to develop programs.

Instead, one needs to focus on a specific set of security requirements. No single set covers all situations, but certain elements arise in many requirements.

For example, the requirement that a program handle error conditions appropriately means that a student must write programs in which error conditions are detected. Buffer overflows may not cause security breaches (especially if availability is not at issue), but they do cause errors to arise.

Thus, a better pedagogic question than “how can we teach students to write more secure programs” is “what principles should we be teaching students to enable them to write more secure programs, and how can we teach them how to apply these principles to specific situations?”
This formulation recognizes that the term “secure programming” is imprecise because it means different things to different people, and in different environments. It also adds to our burden because we must teach students to evaluate situations and determine what “security” means in the specific case. This requires teaching principles.

History plays a part in this. Early papers, such as the Anderson Report, described threats to systems and created a framework for addressing them. These are important not only because current technology implements many of the ideas, but also because the framework itself teaches one how to think about assurance. Consider the reference monitor, a “guardian at the gate” that controls access to a resource. This means that all accesses of the resource must pass through the reference monitor. Spaghetti code, which allows multiple paths of access without a rigorous examination of why each path of access must be present, leads to poor coding and (potentially) security problems. The reference monitor must be checked, to ensure it works correctly; this leads to the requirement of smallness, so it can be validated. Complexity may be required, but unnecessary complexity is a weakness. Striving for simplicity, of style if not of content, is the mark of a good programmer. Finally, if an attacker can alter the reference monitor, or the data upon which it relies, that subverted control will allow unauthorized access to the resource. Hence the programmer must guard against such tampering, by (for example) checking error conditions to prevent attackers from exploiting them. From one framework comes several ideas central to protection.
Over time, technologies change, and methodologies adapt to new circumstances. But principles do not change. They provide guidance for developing new methodologies and technologies. Because of the rapid pace at which our field, computer science, is evolving, focusing on principles and teaching students how to analyze situations, and apply these principles, is critical.

Checklists have a part to play in this process. There are many different types of checklists. Such lists can provide guidance or specific items to consider, and may be used by a student or a grader (auditor).

- A checklist can simply list items to prompt students’ memory, for example “check for possible errors and handle them.” The usefulness of this type of checklist depends entirely on the student’s ability to translate the items on the checklist into the particular domain in which the student is working. This list provides guidance, and the student is expected to look beyond the list as appropriate.
- A checklist can list specific items that students are required to satisfy. These checklists are usually specific, for example “check the return values of all functions to ensure the function worked properly.” These checklists are useful for reminding the student about details, but risk the student performing the items on the checklist without considering their appropriateness, and not looking beyond the checklist for items that need to be considered.

- A checklist can be used to aid in the evaluation of a student’s work, or of the result of that work. Again, this checklist is primarily a guide to the grader or auditor, who is expected to translate the generality of the items into specific criteria appropriate for the work.

- A checklist can list specific items that the student’s work is to satisfy. Here, the checklist requires the assessor to determine if the item is met. If so, it is checked off; if not, the item is marked unsatisfied. The set of satisfied (or unsatisfied) items determines the score.
The best checklists are derived from principles, and their items develop logically through the derivation of principles, methodology, and application to a particular domain, guided by experience of practitioners.

This allows one to justify the checklist rigorously and to see how the principles strengthen the practice; assurance at its best.

The type of education being sought, and the type of environment in which the checklist is used, determines the appropriateness of any particular checklist.

Used properly, a checklist can enhance a student’s education; used improperly, that same checklist can hinder the student’s progress, as well as fail to achieve the goals of that student’s education.
EDUCATING SYSTEM TESTERS IN VULNERABILITY ANALYSIS

LABORATORY DEVELOPMENT AND DEPLOYMENT

Leonardo A. Martucci, Hans Hedbom, Stefan Lindskog and Simone Fischer-Hübner

Department of Computer Science, Karlstad University - Sweden
This paper presents a vulnerability analysis course developed for system testers and the experiences gained from it.

The aim of this course is to educate testers in the process of finding security weaknesses in products.

It covers the four steps of a vulnerability analysis: reconnaissance, research and planning, mounting attacks, and assessment.

The paper describes in detail ten different laboratory assignments conducted within the course. For each experiment, an overview and a description on how to run the assignment together with the expected knowledge obtained are presented. In addition, a course evaluation and lessons learned are also provided.
1. INTRODUCTION

- A constantly growing number of security vulnerabilities, threats and incidents has increased the efforts of IT and Telecom industry to invest in the development of more secure systems.
- A lack of security functionality and security assurance bears high risks for IT and Telecom vendors as it can, for instance, result in high costs for patching running platforms on the clients' sites, dissatisfied clients due to security breaches and system downtime, as well as reputation damage for the vendors' clients.
- Vulnerability Analysis (VA) is an important means for improving security assurance of IT systems during test and integration phases.
- The approach that VA takes is to find weaknesses of the security of a system or parts of the system.
- These potential vulnerabilities are assessed through penetration testing to determine whether they could, in practice, be exploitable to compromise the security of the system. The Common Criteria have requirements on VA to be performed for the evaluation of systems with an Evaluation Assurance Level 2 (EAL2) or higher [0].
For improving security of their platforms before deployment, a major telecom company decided to increase their efforts in VA performed in their test labs, which requires well-educated software testing personnel.

This company decided to outsource the education and training of its employees in the area of VA to an academic institution with experience in both VA and education.

For that purpose, a compact VA course was developed at Computer Science department to be held during three working days for an international and heterogeneous, in terms of knowledge in the security area, group of software testers from industry.

The emphasis of this course was put on practical, hands-on experiments in VA. The course was first held in spring 2005, with 16 participants, and two times more in fall 2005 with 15 and 14 participants, respectively.
The VA course described in this paper was developed primarily on request from the industry.

The requested course was aimed for software testers with no or little knowledge about security in general and vulnerability analysis in particular, but with an extensive knowledge in software testing.

Since the target group was practitioners, a practical course was requested.

The list of topics included in the VA course was, naturally, agreed with the contractor.
The contractor had presented us a preliminary list of topics that should be covered by the course. This list was later transformed into a content list and a laboratory set that was discussed with a contractor’s group composed of software testers, software designers, managers, industrial researchers and security experts.

A three-day course (24 hours course) was selected as the best choice for the course length, since the testers should not be absent from their tasks for a long period.

The course also includes a follow-up one-day recycling class after one year to update the participants about newly found vulnerabilities and VA tools.

Approximately 30 to 40% of the course covers theoretical aspects and 60 to 70% is used for practical assignments.

The latter was intended to give the attendees hands-on experience on how to conduct a VA of either a software component or a complete system. The course is structured in four blocks covering the theoretical part with hands-on assignments providing support to the theory.
The course is divided into the four following blocks:

a) **Introduction to Computer and Network Security.** This block includes motivation, terminology, evaluation criteria, an overview of security standards, risk analysis, ethics and course rules.

b) **Computer and Network Security Protocols and Tools.** It covers cryptography, pseudo-random number generation and testing, public key infrastructure (PKI), firewalls, intrusion detection systems (IDS), virtual private networks (VPN), IPSec, SSH, and SSL/TLS. This block also includes the assignments presented in sections 3.3 to 3.5.

c) **Vulnerability Analysis.** An in-depth description of VA four distinct steps, i.e.: reconnaissance, research and planning, attack mounting and assessment.

d) **Known Vulnerabilities, Reconnaissance Tools and Information Gathering.** This block includes common host attacks, a short section on viruses, worms and anti-viruses, system hardening strategies and several practical assignments presented in sections 3.6 to 3.10.
The last assignment is a full-day final project final practical assignment that concludes the course, a “putting it all together” experiment that summarizes the full vulnerability analysis process.
3. HANDS-ON ASSIGNMENTS

- This section justify the selection of this list of experiments, present the learning goals and also describe how the laboratories were deployed, i.e. network topologies used, applications needed and operational systems employed.

- In addition, this section describe how the knowledge obtained can be applied in the students’ working environment in order to improve the security of a delivered product.

- However, this section first introduce the ethical rules, the laboratory environment and the preparations that had to be done before running the course.

- The laboratories are described in this paper in the same order as they were presented during the course, in order to follow the theory presented in parallel.
3.1 ETHICAL RULES

- VA course teachers clearly instructed the participants that VA strategies and penetration tools must be used for testing purposes and controlled environments only, in order to avoid intentional or unintentional harm to others.

- Hence, this VA course set up the following ethical rules that participants were requested to adhere to:
  - Do not experiment with VA-tools without explicit permission of an authorized party.
  - Do not pass on/publish material, tools, and vulnerabilities to unauthorized parties.
  - Do not use your technical skills in criminal or ethically questionable activities.
  - Always report flaws to vendors/developers first.
  - Software tools provided in this course must only be used in a laboratory environment and on laboratory computers.
3.2 THE LABORATORY ENVIRONMENT

- The laboratory is prepared for up to 20 students working in pairs.
- The workstations are dual boot - Windows XP/Linux and Fedora Core 3 Linux distribution - Intel Pentium 3 900MHz with 256MB of RAM, with one auto-sense Ethernet/Fast Ethernet network interface card (NIC).
- Two additional workstations were configured as servers - one running Windows Server 2000 and one running Fedora Core 3 Linux - running various services.
- Servers are used in some, but not all, laboratories as target systems. Moreover, the server machines are also used to store image files for the other workstations, in order to fast redeploy images if necessary.
3.3 Password cracking

- Passwords are a very basic and common authentication method. Unfortunately, passwords are not the most secure way to implement authentication, but, on the other hand, passwords are the most practical and also less expensive way to implement it.
- During the implementation phase, it is common that developers use some default passwords for debugging and testing their applications before being delivered to integration and test teams in order to speed up the development phase.
- However, it is not uncommon that those debugging and test passwords are left in the application after the development phase is over.
- Those development and debugging backdoors are a serious threat to deployed platforms, as they can even grant full privileges. Detecting those built-in passwords is a must activity for testers.
- This experiment goes one step beyond detection, as students are encouraged to crack password files, in order to evaluate how easy it can be achieved with open-source tools.
3.3.1 Running the Assignment

- In this experiment, students run a local password cracker application. There are several open source tools available for this purpose.
- The choice was John the Ripper v.1.6, a password cracker tool which main purpose is to detect weak UNIX passwords [0], since it is an easy to use and easy to deploy.
- The lab also provide synthetic (artificially populated) **passwd** and a shadow files from a Linux box.
- Before starting to use the tool, students provided with some well-known rules that are used to choose good passwords, such as substituting letters for numbers, how to identify if passwords are encrypted with **crypt( )** or hashed with **MD5** and also the structure of password files.
- Finally, the students just have to **unshadow the passwd file and run the John the Ripper tool**. Since running the password cracking tool can take long time to run, it is advisable to have easy passwords added in the password file. Therefore, students can extract some passwords from the password file quickly.
Even though the experiment is more focused on cracking passwords, developers should first be able to detect if there are backdoors in the system, and, if possible, correct the problem and document it.
3.4 TESTING FOR RANDOMNESS

- Pseudo-random number generators (PRNG) are used in several applications with a very broad scope, from simulation and numerical analysis to gaming applications.
- PRNG are also a crucial cryptographic primitive and a flawed PRNG implementation can compromise the security of a whole system.
- A classic example was a PRNG flaw in earlier versions of the Netscape browser that could compromise the security of SSL connections [0].
- Well-known PRNG have, in most of the cases, public algorithms and are submitted to several batteries of tests. In addition, the U.S. National Institute of Standards and Technology (NIST) also has a list of approved random number generators applicable to FIPS PUB 140-2 standard [0].
- However, it is possible that a programmer might use a flawed implementation of a given PRNG. Therefore, it is crucial for testers to identify weak binary sequences produced by a PRNG. This is the main goal of this assignment.
3.4.1 RUNNING THE ASSIGNMENT

- In this experiment, the NIST Statistical Test Suite [0] is used to evaluate outputs from different PRNG.
- It runs in Windows 32bit systems. It implements 15 statistical tests and also has embedded implementations of well-known bad PRNG, such as the XOR RNG, and also NIST approved RNG, such as the ANSI X9.31.
- Sample data files are also provided as companion part of this test suite.
- The NIST test suite is the latest and most complete test for randomness.
- Moreover, installing and running the NIST Statistical Test Suite installation is straightforward and it is also freely available.
Since generating sufficiently large files of random numbers is a time-consuming activity, the sample files available with the NIST Test Suite are used, since outputs from good and also bad PRNG are provided.

However, the recommendation is to produce files generated using Java classes `java.util.Random` and `java.security.SecureRandom` beforehand and also with the `rand` function from C. Therefore, students can evaluate the output quality of those PRNG, which are popular among developers.
Since analyzing the outputs of the NIST Test Suite demands basic knowledge of statistics, it is crucial to provide to the students a short background on hypothesis testing before running the test suite. It is also important to briefly explain the tests and the test requirements as well, since tests have different requirements regarding the minimum number of samples and the minimum length (in bits) of each sample.
In order to evaluate if a given PRNG provide weak binary sequences, developers should inform and provide methods to call random number generator objects or functions in order to test them (if any are used) or, at least, document that a given PRNG is being used for security reasons.

In addition, it is possible to verify if a given implementation of a given PRNG class produces weak binary sequences or not, and, eventually, validate a PRNG class. Furthermore, testing for randomness can be automated.
3.5 FIREWALL

- Firewalls can be briefly described as filters that follow some screening policies defined by the network administrator.
- There are several firewall applications and boxes available in the market and even open source firewalls, such as `iptables`.
- It is of ultimate importance for system integration teams to know how firewalls work, where they should be placed in a network topology and how their rules are defined and verified. The goal of assignment is to provide hands-on experience on such aspects.
3.5.1 Running the Assignment

- Students are divided into groups of two.
- Each group is given a description of a setup as the one described in Figure 1 and are asked to write firewall rules in Linux using `iptables` implementing a given policy defined in the problem statement.
- Rules are written as if all the necessary NIC were present. Note, however, that the students are only working with virtual interfaces and not with two separate NIC.
**Figure 1.** Network topology for the firewall assignment.
When deploying systems that have a firewall, testers should be able not only to test the system from a black box point of view, but also read, understand, verify and evaluate firewall rules, in order to look for inconsistencies in the access control list. Therefore, hands-on experience is fundamental for testers in order to perform such tasks.
Communication protocols are basically defined in open and proprietary standards. Those standards are usually well-known and were evaluated by a committee and also by the users’ community. Therefore, finding out shortcomings in standards is not an everyday event.

However, finding out vulnerabilities in implementations of communication standard is not uncommon since standards provide guidelines for implementing, but not for the coding. Thus, careless programming and also coding errors can lead to a series of vulnerabilities that can compromise a running system.

Testers should therefore be able to detect such flaws in the communication protocols during test and integration phases.
There are basically two ways to test the implementation of a communication protocol.

- The first is to extensively review the source code in order to find out implementation flaws, but this method is cost and time inefficient.
- The other alternative is the black box testing approach, i.e. a functional test method that evaluates a system from the input and output point of view in order to identify system vulnerabilities.

It is important to testers not only to be familiar with those tools but also how to interpret results and draw conclusions from them.

In this experiment, the students are encouraged to work in larger groups and execute a black box testing against a running system.
3.6.1 RUNNING THE ASSIGNMENT

- The goal of this experiment is to check if the SNMP implementation of this router is vulnerable to a malformed packet generated using the PROTOS test suite and also identify the vulnerability type (i.e. test material, test group and test case).

- The PROTOS tool is used as a black box testing tool against the SNMP protocol implementation of Cisco 1005 router running IOS 11.1(3). The PROTOS Test Suite c06-snmpv1 is used here to perform a Denial of Service (DoS) attack against the Cisco router.

- PROTOS c06-snmpv1 test suite is divided into four separate test material packages: two test packages regarding packets with encoding errors and other two test packages regarding packets with application exceptions.

- Since PROTOS is a Java-based application, any OS can be selected, as long as a Java Virtual Machine is installed.
Students were divided in two groups - each group with four workstations. The network topology used in this assignment (for each group) is shown below in Figure 2.

if it exists. PROTOS c06-snmpv1 test suite documentation is used to identify test groups. It is useful to provide a short introduction of the SNMP protocol PDU in order to provide some background information on this protocol.
Figure 2. Network topology for the vulnerability test using the black box test method assignment.
In this assignment students are encouraged to cooperate in order to find out if the router is vulnerable or not to SNMP malformed packets. All workstations continuously ping the router in order to check if it alive or not. No console access to the router is provided until the test case is identified. After the test case is identified - in the particular case of this router running the IOS 11.1(3), it is a basic encoding rule (BER) error in the length field of a Get-Request SNMP PDU - students are allowed to connect to the Router using the console port and witness the router crashing after receiving just one malformed packet.

Basically, PROTOS just send one test case after another, but the running process doesn't stop if the target system misbehaves. Therefore, students have to keep track of the echo reply messages, and follow PROTOS execution to finish this assignment.
Black box test strategy is very successful method to identify vulnerabilities in implementations of communication protocols.

Automated testing tools are especially useful in the security evaluation of a given system because an entire test battery can be executed with a single command.

The PROTOS test suite is a freely available tool, but few communication protocols are available – WAP, HTTP Reply, LDAPv3, SNMP, SIP and H.225.0.

Additional test suites are available within Codenomicom, the commercial version of PROTOS.

Using such tools, testers can quickly evaluate the robustness of protocol implementations and report vulnerabilities found back to the development team in order to fix them.
3.7 NETWORK ANALYZERS AND ARP SPOOFING

Network analyzers are tools used to capture and analyze network traffic being transmitted in a given collision domain.

These tools simply set the network interface card to promiscuous mode in order to bypass the MAC filtering rules. They are also very common tools for testers to evaluate the network traffic received and transmitted by a given system.

However, testers also need to understand that some weaknesses are inherent to the protocol design. It is well-known that some protocols were designed with very little concern about security, such as FTP, TELNET and etc.

Although the decision of using such protocols is not up to test and integration teams, they should be able to identify and report the existence of such protocols.
The experiment goals:

- Prove to the students that some general affirmations about security are not true, and that skepticism and hands-on knowledge (in this case networking) are very important points in the security area.
- Use the network analyzer to capture the access password and the privileged user password.
- Prove that knowledge in network environment (i.e. topology and protocols) is definitively important to assess system vulnerabilities.
- A second part of this experiment is prepared in order to prove to students that knowledge of the networking device internals and network topology can impact security analysis. The same assignment also present an ARP spoofing experiment.
3.7.1 Running the Assignment

- The assignment is divided in three parts, each part with a different network topology.
  - In the first part, students verify that a very popular network protocol, the TELNET, is very weak regarding security using a network analyzer tool. The main goal of this part is to introduce a network analyzer tool to the students.
  - In the second part, a rather insignificant change in the network topology modify test results - and it is up to the students to figure out why changes occurred.
  - Finally, in the last part of this assignment students test an ARP spoofing tool on a switched network, verify the achieved results and explain those results according.
The first part of the assignment has two goals:

- Introduce the network analyzer (Ethereal) through a very simple scenario
- Demonstrate that the design of some protocols have not taken security as a requirement

TELNET, a popular terminal emulation application is used to prove the assertion.

All workstations and a Cisco 1005 router are connected to a 10Mbps hub. The instructor task is to access the router via TELNET. The network topology is depicted in Figure 3.
Figure 3. Network topology for the network analysis assignment - Part I and Part II.
The goal of the second part of this assignment is to demonstrate that knowledge of network device and its internals can make difference on a VA test.

The scenario is basically the same of the first part of this assignment, presented in Figure 3: a system administrator is configuring a router using TELNET.

However, the network topology has slightly changed: the 10Mbps hub was upgraded by the system administrator to a 10/100Mbps auto-sense hub.
Before starting to access the router via TELNET, the instructor must force his network interface card to 10Mbps (without students noticing it).

Therefore, TELNET traffic from the instructor workstation to the router will be send in the 10Mbps backplane. Since the NIC of the student workstations will be automatically set to 100Mbps

No TELNET traffic will be transported in the Fast Ethernet backplane of the hub, and, therefore, they will not be able to capture the traffic exchanged between the instructor’s workstation and the router, but they will be able to verify that the router is up and running using ping or TELNET.
Students are then invited to think and explain why such subtle change in the network topology had dramatically changed the achieved results. The answer lays in the internals of the auto-sense 10/100Mbps hub.

Since the auto-sense 10/100Mbps hub has two independent backplanes, one running at 10MHz and the other at 100MHz, it also has two independent collision domains connected by a layer 2 bridge. Thus, the traffic between the router and the administrator’s workstation is transmitted only on the 10MHz backplane, but this traffic is not forward to the 100MHz backplane, where all student workstations are connected.
The goal of the third part of this assignment is to prove that some common assertions on security can be proven wrong with a clear understanding of the subject ARP (Address Resolution Protocol in the case of this assignment).

Since one MAC address shall be spoofed by only one machine per time period in order to capture the all traffic flow, and in order to allow all students to work in parallel, student workstations will spoof other student machines, in a logical ring topology.
In order to create a constant traffic flow, students were instructed to continuously ping the workstation to their right or to their front.

The target assigned for the ARP spoofing experiment is the ICMP echo request traffic flowing from the next workstation to the right to the one next to it.

Therefore, students send to the next workstation to the right ARP reply PDU in order to update its ARP table with the attacker’s MAC address instead of the MAC address of the recipient.

Figure 4 presents the ICMP echo-request PDU traffic flow chain, the ICMP echo-request target flow, the attacker and target workstation.
attacker poisons ARP table of this workstation

Target flow

1

ICMP message flow

2

Figure 4. The ICMP echo-request traffic flow.
The logical chain topology depicted in *Figure 4 allows all students to work in parallel in order* to optimize the time to run the assignment.

With such kind of topology, *students play the role of attackers and victims at the same time*, since while they are poisoning the ARP table of their neighbor to the right, the other neighbor is poisoning their ARP table.

Cain & Abel was used as ARP poisoning tool in parallel with Ethereal, which was used to capture traffic in the network segment.

However, the main goal of the experiment is not just the understanding of how such attack is performed, since it is basically straightforward as soon as the students were told that ARP is a stateless protocol.
3.7.2 KNOWLEDGE OBTAINED

- Network analyzers are powerful tools that testers should be able to master, since they can provide a good insight of the data being transferred from and to a running system.
- However, only mastering such tools is not enough without knowledge about the test environment, as proved in the second part of this assignment, and about the protocols involved, as demonstrated in the ARP spoofing experiment. In addition, ARP spoofing is an easy to detect attack.
- The goal of this assignment is to demonstrate that a lot of myths in the security area are not true, and those myths can only be proven wrong with a good understanding of the target system and its running protocols.
- In conclusion, it is fundamental for testers to have a deep understanding and knowledge of the test environment and of the running system especially when testing for security.
3.8 PORT SCANNING

- Port scanners are used in the network reconnaissance phase of VA.
- Such tools basically scan one or more devices in order to find open ports that could be exploited for an attack.
- It is a fundamental task for testers to evaluate if unexpected open ports can be found in a system, especially during the integration phase of a system, just before deployment.
3.8.1 Running the Assignment

- The goal of this experiment is gathering information about two running systems. Two workstations configured as servers (one Linux and one Windows server) are used as target systems.

- Students run Network Mapper (NMAP), an open source port scanning tool, under Linux (it can be also compiled for Windows 32bit, since the source code is available).

- Servers run several network services, such as FTP, HTTP, NetBIOS (Windows server), etc.

- Students have to find and identify the servers in a given IP network range, since their IP addresses are not provided, find out the operational system running and identify the open ports in each server.

- The network topology employed is a simple star topology with all workstations and servers belonging to the same broadcast domain and in the same IP subnet.
Port scanners are reconnaissance tools usually employed by attackers in order to gather information about a system and identify open ports that can lead to successful break-ins.

Testers should use such tools in order to identify unexpected open ports (i.e. ports that are open, but should not be), report them, and, eventually, terminate a process in order to close those ports during system integration phase.
3.9 NODE HARDENING

- Part of the testing process is to test systems that are to be delivered and to do system tests of product. In that phase it is desirable that the testers make sure that the products are as secure as possible.

- One step in this securing process is to be sure that no unnecessary services are running and that the services needed run with minimal rights only. It is also vital to be sure that all the software used is patched to the latest patch level.

- The whole process is known as host hardening. Host hardening can be conducted in several ways, such as using checklists, experts in the area or software tools.
3.9.1 Running the Assignment

- An open source tool for the hardening process was selected for this assignment.

- The chosen tool was Bastille [0].
  - It is a node hardening software that lets the user answer a number of questions on how they want to be configured and then configures the system according to the answers.
  - It works as a high level automated checklist.
  - It can also be used to analyze the current configuration and give suggestions on detected suspicious or dangerous configurations.
Students were asked to make their own computers as secure as possible given that it still should be functional as a networked computer.

They were also told that, later on, they will use a security scanner to test how secure their systems were (section 3.10).

The goal of this exercise is to provide to the students an understanding of the host hardening process and the phases that are part of this process.
Even though there are tools that can be used in the hardening process they still rely on in depth knowledge of the system that is to be hardened.

It is also essential that these tools are up to date with the latest configuration files.

Even though they can be a great help in the hardening process and also in the later verification of the hardening there are still a large manual portion of both these steps that require practical system knowledge.

A system tester needs to know that in order not to be overly confident in the reports that these tools generate.
A security scanner is a tool that attacks, or simulates attacks, on a system.

- The attacks it will perform are kept in a configuration file. The more up-to-date this configuration file is, the more accurate the scan. This means that an up-to-date security scanner will scan the system for all known vulnerabilities from the outside.
- Some scanners also have components that make it possible to perform attacks both from the inside and from the outside.
- A number of flavors exits but the most known ones are IS [0], Retina [0] and Nessus [0].
- These tools are often a good starting point when doing a VA of a system since they automate the testing of the known weaknesses.
The problem with these tools are generally not running them but rather how to configure the attacks and how to interpret the results, since false negatives and false positives in the attack process is highly influenced on the setup of the tool.
3.10.1 Running the Assignment

- In this exercise, two target servers were set up. One Windows 2000 (set up as a domain server) and one running Fedora Core 3 Linux.
- Both not patched, running all standard services and acting as target systems for the security scanner.
- The Windows server ran IS and the Linux server ran Nessus. Neither the configuration nor the IP addresses of the servers were given to the students.
The students were divided into groups of two.

All the groups were asked to find the servers' IP addresses and to find out which operating systems they were running.

Students were also told that the servers were the only two network devices running an HTTP server in the network.

After finding the servers, the students had to scan them and report all the vulnerabilities found.

Finally, they were also requested to scan their own workstations in order to verify the hardening process that was performed earlier (section 3.8).
3.10.2 KNOWLEDGE OBTAINED

- These tools are a good help to the testers in the verification process. However, they cannot be fully trusted and will inevitably generate both false positives and false negatives.
- A good knowledge of the system is needed in order to be able to correctly interpret the tool outputs.
- A tester also needs a good understanding of the tool in order to configure it correctly and customize its behavior if it is to be used on a non-standard system or with non-standard services.
Finally, we let the students put all their knowledge together in a final “grain to bread” exercise.

The goal was to make this assignment as close to their reality as we could, in a controlled environment, with limited resources and time.
3.11.1 RUNNING THE ASSIGNMENT

- In this exercise, the students were divided into groups of four.
- Each group was given a Fedora Core 3 Linux server with all services running.
- Every group also got a requirement document describing the role of the server and the security requirements on it.
- They were asked to find out what needed to be done in order to fulfill the requirements and also to perform the changes and verify the results.
They also had access to the Internet and were told that they could use it freely. Moreover, they could also use any other freely available tool found on Internet. The students had to report all the miss-configured parameters, vulnerabilities and also had to suggest changes in the system in order to adhere to the specification. The results were discussed in a summing-up session just after the exercise.
3.11.2 KNOWLEDGE OBTAINED

- This assignment provided to the students a wider insight of the highly creative art of VA and it further reinforced their conclusions from previous assignments that the combination of good system knowledge, good understanding of the tools internals and their shortcomings are essential in order to correctly interpret and test a system.
This section presents the course evaluation according to feedback from the students. Also, present the lessons learned from the teachers and university staff’s point of view.

Just after the course conclusion, the students received a questionnaire used to evaluate the course and the usefulness of the laboratories.

The students were invited to individually answer the questions following a scale from 1 (lowest grade) to 7 (highest grade).

They were also invited to comment their choice. No identification field was included in the questionnaire in order to provide anonymity to the students.

A summary of the achieved results is provided in Table 1, regarding 36 returned questionnaires.
<table>
<thead>
<tr>
<th>Questions</th>
<th>Average Grade</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation and usefulness of the assignment:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security scanning</td>
<td>5.8</td>
<td>0.9</td>
</tr>
<tr>
<td>Port scanning</td>
<td>5.8</td>
<td>0.9</td>
</tr>
<tr>
<td>Node hardening</td>
<td>5.6</td>
<td>1.1</td>
</tr>
<tr>
<td>Black box test method</td>
<td>5.4</td>
<td>1.1</td>
</tr>
<tr>
<td>Password cracking</td>
<td>5.4</td>
<td>0.7</td>
</tr>
<tr>
<td>Network analysis (Parts I and II)</td>
<td>5.4</td>
<td>0.9</td>
</tr>
<tr>
<td>ARP spoofing</td>
<td>5.2</td>
<td>1.2</td>
</tr>
<tr>
<td>Putting it all together</td>
<td>4.8</td>
<td>1.2</td>
</tr>
<tr>
<td>Firewall</td>
<td>4.4</td>
<td>1.2</td>
</tr>
<tr>
<td>Testing for randomness</td>
<td>4.0</td>
<td>1.7</td>
</tr>
<tr>
<td>Did the course fulfill your expectation?</td>
<td>5.1</td>
<td>1.1</td>
</tr>
</tbody>
</table>
According to Table 1, it is possible to verify that seven out of the ten assignments of the course were highly classified by the students, being graded in the interval 5.2 and 5.8.

They were: security scanning, port scanning, node hardening, vulnerability analysis using the black-box method, password cracking, network analysis and ARP spoofing.

The reasons reported by the students included the usefulness of these assignments in real test environments and the easy and fast deployment in the tests.

However, some participants were already familiar with some security tools and others reported that some of these assignments were not particularly useful in their specific tasks.
Firewall, testing for randomness and the final assignment were considered the least interesting assignments, being graded between 4.0 and 4.8. Some considered that the time reserved for the final experiment was short, while others thought that it was no more than a sum-up of the previous exercises. However, part of the students considered it a good concluding exercise.

The firewall assignment was considered not that useful because testers hardly evaluate or write access-list rules in general, so this experiment was down rated by the participants compared to other assignments.

Finally, testing for randomness was considered hard to be implemented and most of the students were not comfortable with their background in statistics.

Indeed, testing for randomness is a test were a security primitive is being tested, which is usually underrated by testers, that are more concerned in securing systems in an upper abstraction layer. To the question “did the course fulfill your expectations?” the average obtained grade was 5.1 (with a standard deviation of 1.1), what was considered very good result from the teachers and the contractor’s point of view, regarding the heterogeneity of the audience.
COMPUTER SECURITY EDUCATION

Past, Present and Future

Carol Taylor, Rose Shumba, and James Walden
University of Idaho, Indiana University of Pennsylvania, and Northern Kentucky University
ABSTRACT:

- This paper presents an overview of computer security education in academia.
- It examines security education from its recent past in the late '90s, evaluate its present state and discuss its future potential.
- A brief history of government programs that affect security education is presented along with their role as funding sources.
- The software industry perspective on security education and their relationship with academic security programs is also discussed.
- It defines goals and objectives for the future of academic computer security programs and address barriers to successful achievement of those goals.
Computer security became a tangible Computer Science subdiscipline in the 1970’s as the need to protect information became important with growing computer use in government and industry. At that time computer security research was funded by the military and primarily aimed at the protection of sensitive information. Computer security researchers and practitioners were few in number, worked primarily in the defense industry, and were mostly self taught. Today, computer security is well established as an area of research and study within Computer Science. There are defined career paths for computer security professionals and an array of professional training and academic degree programs. If we compare the activity and interest in the field of computer security with its inception, one can say that a great deal of progress has been made. Yet, there is ample evidence that much more remains to be done.
Popular press reports describe daily the number of vulnerabilities found and the latest abuse of our systems by individuals in search of easy profit. -Tumbleweed Communications, an e-mail security provider estimates that two-thirds of all e-mail is illegitimate traffic [1].

Several well-respected Computer Security researchers and educators also question the state of our knowledge and practice as a discipline.

Roger Schell describes how the lack of science in computer security has actually led to a decline in the number of secure systems from a peak in the 1990's [4].

Eugene Spafford, also questioned the quality of security practice in his paper, *A Failure to Learn From the Past* [5]. Spafford recounts the 1988 Internet worm incident and points out that the same conditions that allowed the worm to wreck havoc on systems still exist in 2003 nearly 15 years later.
As security researchers, it is discouraging to see the low level of practice in the real world with the constant stream of new system vulnerabilities and the increasing number of malicious programs in search of one of the many unpatched systems.

But, as educators, we are hopeful that in time, through education, we can improve the current state of computer security by producing students trained in secure coding, with knowledge of secure system design and operation.

While there are multiple studies in the security education literature that document experiences of individual departments in developing academic security programs, there is at present no general study of security education.
The paper goals:

- This paper examines the state of computer security education from the past, present and future.
- It include views from three separate groups that have a strong interest in security education: academia, government and industry.
- It reviews the state of academic security education since 1997, the year of the first CISSE conference [6], in order to assess progress over the past eight years.
- Evaluate the current state of computer security education plus provide personal insights from respective programs in computer security.
- Last part of the paper discuss the future of security education in terms of goals and objectives and note possible barriers to progress.
2. COMPUTER SECURITY EDUCATION IN THE PAST

- This section trace the evolution of government initiatives for academic computer security education over the past eight years.
- It provides statistics on the programs and the events in government that have influenced the overall progress of the early academic security programs.
- It also briefly mention the state of early academic programs.
Early academic programs in INFOSEC education were primarily for graduate students. Consequently, undergraduates wanting to learn about computer security had to take graduate courses or do so through independent study.

Graduate level security courses typically concentrated on Multi-level Security (MLS) concepts or simply covered cryptography without a lot of practical system analysis [23].
The first NCISSE conference was held in 1997 and was established as a forum for dialog between government, industry and academia to define requirements for information security education and encourage development and expansion of information security curriculum at the graduate and undergraduate levels [6].

This conference was the earliest official forum to recognize computer security and bring together academics teaching security with key people in industry and government (Figure 1).

Around the same time 1996-1999, President Clinton established the President’s Council on Critical Infrastructure Protection (PCCIP) and subsequent President Decision Directive 63 (PDD63) [7].

In establishing the PCCIP, the president recognized the vulnerability of the US infrastructure and acknowledged its importance to national security.

PDD63 simply expanded the definition of critical infrastructure to include cyber security [7].
Soon after PDD63 appeared, in 1999, NSA established the Centers for Academic Excellence in Information Assurance (CAEIA) [8].

These centers were academic institutions with expertise in cyber security as demonstrated by a number of security oriented faculty and curriculum that met federal security training standards [8].

The purpose of this program was to increase the number of “security professionals of different disciplines”.

During the first year, seven schools were established and recognized at the Third Annual NCISSE conference [7].
In February of 2000, President Clinton released the National Plan for Information System Protection [9] which established the Scholarship for Service (SFS) program managed by the National Science Foundation (NSF) [10].

This program provided scholarships for undergraduate and graduate students for up to two years in exchange for an equal amount of Federal Job service upon graduation.

In order for a school to obtain a scholarship program, they must first qualify as a Center for Academic Excellence [8].

During the first two years of the program, 150 students were enrolled [11].

In 2002, President Bush created the Department of Homeland Security (DHS) which united 22 agencies into one common group for the purpose of improving homeland security. One of their responsibilities was and continues to be funding R & D for new scientific understanding and technologies in support of Homeland Security [12].
In 2003, the President’s National Strategy to Secure Cyberspace was passed by President Bush. It identified four major actions and initiatives for awareness, education and training which include [13]:

1. Promote awareness nationally to empower all Americans to secure their own systems.
2. Foster training and education programs to support national and cyber security needs.
3. Promote private sector to support widely recognized cyber security certification and training programs.
4. Increase efficiency of existing federal cyber security training programs.

From 1997 to 2003, the US government created many initiatives for cyber security which appeared to recognize its importance for national security and the continued well-being of our nation. However, the only money allocated to academic security education was from the NSF SFS program.
Figure 1. Timeline of Events Affecting Security Education
This section looks at the present state of computer security education. It also review the growing body of literature on established academic programs and discuss the typical approaches for establishing programs. Security education standards are discussed plus government and industry influences.
3.1 ACADEMIC PROGRAMS

- Recently, a number of colleges have reported on their experiences adding computer security to their programs.
- Most schools appear to take one of two approaches:
  - integrate security within individual CS courses [14, 15, 16] or
  - create a special computer security degree or track [17,18,19].
- A few schools have opted for a combined approach where they have both specialized and integrated courses [20, 21]. There are reasons as to why a university chooses one approach over the other in their development of a computer security emphasis which often includes factors beyond the control of the institution. There are also tradeoffs with regards to these alternative approaches to computer security education.
Schools that choose to create a computer security track or special degree appear to have faculty that have experience in computer security or are strongly interested in pursuing security training [20, 19].

There also appears to be department or institutional support for a Security track and at least enough funding for course development.

Integrating security into existing CS courses without offering specialized Security courses is one way that schools with limited resources in terms of faculty or funding can still offer security to students within their programs [14].

Faculty in these programs do not need not be retrained or develop completely new courses.
The effectiveness of each approach relates back to the goals for the CS graduates of a particular program in terms of computer security expertise.

Programs that want to produce graduates with strong computer security skills capable of obtaining a computer security position have created specialized programs in computer security.

CS programs that want their students to have exposure to computer security but not necessarily produce computer security professionals can achieve this through an integration of security principles into existing CS classes.

There is no clear evidence that specialized courses in computer security are superior to standard CS courses with integrated security components.
However, schools that have chosen to integrate security within their existing curriculum point out several advantages over the specialized course path [14, 21]:

- Provide a security foundation to all their CS students as opposed to only those with a security interest
- Security concepts are learned within the broader CS topics such as system design, network administration, and programming
- The approach is available to all schools even those with limited resources and only requires faculty creativity and motivation
3.2 GOVERNMENT SUPPORT OF ACADEMIC PROGRAMS

The government has several programs that currently support academic computer security education. Many of these programs were begun as the result of government initiatives related to national security. Here, we view the current status of these programs.

- **NSA-DHS Centers of Academic Excellence in Information Assurance [8]**
  - Current: Has 67 Centers in 27 states
  - Started: 7 schools
  - Funding: Provides no monetary support for the Centers

- **NSF Scholarship for Service Program [10]**
  - Current and Future: 350 graduates by 2005
  - Started: 150 enrolled
  - Funding: An annual budget of $30.5 million

The programs directly support Computer Security education within academic institutions. These programs appear to be thriving with an increasing number of Centers and students enrolled in scholarship programs.
In order to provide incentives for faculty and attract students into a field, the field of computer security needs a certain level of support in research dollars.

Research fuels education by providing opportunities for faculty to publish, students to work on projects, and money to purchase equipment [23, 20].

Several long-time researchers and educators have noted that Computer Security needs a continued long-term commitment of basic research funding if it is to sustain itself as a viable area of study [22, 23].

At the first NCISSE conference in 1997, Bishop [23], a computer security researcher and educator, discussed the need for long-term funding as providing a stable base of resources and people which could be drawn from without having to continually start from scratch. In a later talk at CISSE in 2000, Bishop commented that the government appears to be offering no support for basic security research which he states could ultimately prove disastrous [24].
Spafford briefed congressional staff in July, 2005 on the serious lack of funding in cybersecurity research [22].

Spafford’s group, the Presidential Information Technical Advisory Committee (PITAC), issued a report in spring of 2005 that condemned the meager government investment in computer security research.

Spafford noted that the NSF has become the primary agency for funding cyber security research with an annual budget of $30 million a year. This translates to only 8% of proposals being funded which as pointed out by the Computer Research Association (CRA) is discouraging student entry into the field [22].
The lack of long-term research funding was also noted by the Cyber Security Industry Alliance (CSIA), a group of security vendors who re-iterated PITAC findings in their own report [25].

The CSIA report stated that the Department of Homeland Security’s budget in FY05 for science and technology is over $1 billion but that the budget for cyber security is just $18 million or about .02% [25].

Andy Birney, the editor of *Infosecurity* magazine, holds the government partly responsible for the nations’ current cyber-security problems.

Birney claims that a lack of government investment in security research discourages PhD students from entering the field [26]. This in turn creates shortages of faculty trained in security at academic institutions that produce the students entering the work force as programmers.
Another recent funding trend that affects computer security programs is the significant cuts from DARPA spending for university research [27].

DARPA has been a long-term source of basic Computer Science research funding for many years. This past year DARPA has cut the portion that goes to universities from $214 million to $123 million.

They have shifted away from general research projects to more concrete deliverables produced in shorter time frames.

This shift has resulted in a huge increase of proposals being directed towards NSF as one of the last Computer Science funding sources [27].
3.3 INDUSTRY VIEW OF ACADEMIC PROGRAMS

- The computer industry comprises an important part of the United States economy, and almost all modern products and services use computer software.

- In the Report of the 2nd National Software Summit, leaders from academia, industry, and government argued that software should be elevated to an issue of national importance with a goal of “Achieving the ability to routinely develop and deploy trustworthy software products and systems, while ensuring the continued competitiveness of the U.S. software industry” [28].
The Build Security In (BSI) Software Assurance Initiative from the Department of Homeland Security seeks to achieve that goal in collaboration with academia and industry [29].

However, few companies accept responsibility for the poor quality of software that exists in most commercial products. Instead, some within the industry blame universities for producing programmers that don’t know how to produce secure code.

Davidson, CSO of Oracle, appears to be a leading critic of academia [30, 31]. She believes academia should help shape the CS field and foster a culture of security. Davidson believes academia should produce CS majors that place more value on properties of safety, security and reliability above coolness and elegance. Davidson does not feel that industry should have to train programmers in security coding practices since they should have already acquired these skills prior to graduation [30].

Another group of software companies including Oracle and Microsoft, among others, discussed the failure of academic programs to produce security conscious programmers at a San Francisco Secure Software Forum in February, 2005 [31].
However, others point out that academia can’t be entirely responsible for the problem of secure code.

One panel member from the Secure Software Forum blames software companies that are still putting features above security [29]. The view that it’s partly industry’s fault that we have so much bad software is supported by Birney.

Birney refutes the popular belief that the coding from three perspectives: academia, industry and government. root cause of vulnerabilities is insecure coding [26]. He discusses secure

Birney believes that a lack of government funding for academic computer security programs leads to a shortage of faculty with backgrounds in security as was discussed in the previous section.

Furthermore Birney shifts some of the responsibility for vulnerable software to industry that still places development speed and profit over security.
While many in industry seem eager to blame academia for bad software without doing anything to help the situation, Microsoft is an exception in that they are working to fix the perceived problem.

In 2002, Microsoft shut down for several weeks in order to train its workforce in secure software development [32]. Furthermore, they are one of the few companies investing in academic education through their 2-year old Trustworthy Computing curriculum program.

They are offering $750,000 in grant money to 15 universities to produce security related curriculum. The curriculum materials are then made publicly available on their web site.
4. INSTITUTIONAL EXPERIENCES IN SECURITY PROGRAM DEVELOPMENT

- This section offers the authors' individual experiences in Computer Security program development.
- Each program is different and is representative of various types of schools that develop security expertise in CS.

- The University of Idaho represents a mature, long-term security program since they were one of the original designated NSA CAE's.
- Indiana University of Pennsylvania is a more recently designated CAE (2002) and represents a less established security program.
- The computer security program at Northern Kentucky University is the smallest of the three and represents a non-CAE program that is mostly based on the efforts of a single faculty.
In establishing security programs at all three schools there were several commonalities noted.

All three schools noted some difficulty with a lack of computer security curriculum standards.

All programs began as the effort of one (or a few) faculty who instigated the security effort.

All three schools are not major universities with large amounts of funding, so these programs were established in spite of limited funding.

Students at the schools appear to be very interested in the topic and enrollment in the programs continues to be strong.
Information assurance curriculum development at the University of Idaho began in 1991 with the arrival of Dr. Jim Alves-Foss. Dr. Alves-Foss graduated from UC Davis with a specialty in computer security and became the first IA faculty at the University of Idaho.

The first security course developed consisted of a combined upper division undergraduate and graduate course, in computer security that emphasized both theory and practical knowledge.

The addition of a second IA faculty, Dr. Debra Frincke, in 1993 resulted in the creation of several more security courses, Network Security and a senior/graduate level seminar in Intrusion Detection.

These early courses were followed by a senior/graduate course in Survivable System Analysis, a seminar in Security Policies and a course in Exploit Techniques and Defense. Other CS faculty became interested in IA and assisted with the development and teaching of these courses. In 2004, several additional courses were added including Forensics analysis and a lower level general Security Course [33].

These courses evolved as the perceived need arose and as an outgrowth of faculty research interests.
During the period of curriculum development effort, they became an NSA CAE/IAE [8] and also participated in the NSF Scholarship for Service (SFS) program [10].

The NSA program has certain curriculum requirements which must be met in order to qualify for program continuance.

The NSA CAE/IAE designation is closely tied to the National Security Telecommunications and Information Systems Security Committee’s, NSTISSI training standards especially 4011.

In becoming an Academic Center of Excellence, the institution must demonstrate that their curriculum complies with the 4011 standard plus at least one other standard selected from the 4012 – 4015 documents [34]. Certification verifies that the college teaches skills that cover each of the seven topic areas of 4011.

In 2005, we have also begun integration of security concepts within several of our standard CS courses. We are planning on introducing secure coding into our beginner coding classes plus a computer security integrated software engineering class.
Indiana University of Pennsylvania (IUP) was designated a Centre of Academic Excellence in 2002. Since then, there has been noticeable improvement in curriculum development.

In 2003, a Bachelor of Information Assurance degree, jointly offered by the Computer Science department and the Criminology department track was introduced.

In 2005, a Master of Science in Information Assurance was recently developed.

This is an interdisciplinary program designed to meet the industry and government needs for computer/network/information security professionals. The first offering was in the fall of 2006.

- Through the SIGSCE Special Projects fund and local IUP Senate funding, hands-on exercises for Information Assurance courses have been developed. These are being pilot-tested in the department.

- To gain an industrial perspective of information assurance, industry partners provide guest lecturers on legal issues in Information Assurance classes and at the Information Assurance club meeting plus state police consultants provide guest lectures on legal issues.
4.3 NORTHERN KENTUCKY UNIVERSITY

- Computer security curriculum development began at Northern Kentucky University (NKU) in 2002 with the introduction of a graduate computer security course by Dr. Charles Frank.
- Undergraduates enrolled in the class as a senior-level special topics elective course.
- The course focused on security fundamentals and network security and included a variety of lab exercises.
- The math department also offered a cryptology class, in which many computer science students enrolled.
- In 2004, NKU added a new Computer Information Technology (CIT) degree with a track in Network, System Administration and Security.
- An undergraduate class in computer security was added as a requirement for the new track and as an elective for both CIT and CS majors.
This section discuss the future, where we are going with particular attention to objectives and potential barriers to success.
In trying to visualize the future of computer security education, it is useful to set goals and define specific objectives for reaching those goals.

No one in the security field would argue with the general belief that providing a security background is beneficial to all students graduating from CS departments.

One overall goal would be to increase the number of CS graduates with an understanding of computer security principles. Consequently, one objective that would help in reaching this goal is to increase the number of CS programs that teach computer security.

Analyzing the specific steps needed to realize the objective of expanding the security education programs leads to a discussion of barriers to success for computer security education, the topic of the next section.
5.2 Barriers to Success for Computer Security

- Achieving the objectives of promoting or increasing security concepts in CS programs requires some investment on the part of both institutions and the faculty member(s). These respective responsibilities for faculty and institutions are outlines in Table 1.

- In addition to the specific activities of faculty and institutions that wish to add computer security to the curriculum, there are other possible barriers to establishing a computer security emphasis. These are outside the control of faculty and their colleges and include:

  - No standard for CS curriculum development
  - Lack of government funding in basic research
  - Limited industry involvement
<table>
<thead>
<tr>
<th>Responsibility</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Institution</td>
<td>Faculty</td>
</tr>
<tr>
<td>Reduced Teaching Load</td>
<td>Learn Computer Security</td>
</tr>
<tr>
<td>Travel and/or Training Support Grants</td>
<td>Collaborate with Computer Sec. Research Inst.</td>
</tr>
<tr>
<td>Tenure Support</td>
<td>Travel to Conferences</td>
</tr>
</tbody>
</table>
Each of the barriers is explained in terms of its relations to computer security education.

- There is currently no accepted standard for college level computer security curriculum development. This presents a barrier to the development of computer security programs. Without an accepted standard, departments must work harder to define course content [14, 19]. The lack of an academic computer security curriculum standard was recently studied by one of the authors [35]. That study noted the inadequacy of the 40XX Training Standards for academic programs and described the problems faced by academia in trying to map their curriculum to these standards.

- The lack of government funding was addressed in a previous section and noted as a disincentive for promotion of security education. If there is little or no research funding available in a given field of study, then there is no way to support graduate students who are to become future faculty and eventually promote their own research programs. Consequently, disciplines that lack research support struggle to recruit students and faculty since there is a perception of a lack of resources in the field. The current dismal situation where only a small percentage of cyber security proposals are funded by the NSF is not conducive to promoting computer security programs within academia.
The software industry is concerned with the perceived lack of security awareness in students graduating from CS programs. Yet, they are not as a group volunteering to assist with this problem either by funding or other direct involvement.

The objectives of the CISSE conference were to establish a working partnership between government, industry and academia [6].

Industry and government should provide better support for higher education. Yet, outside of the institutions with large, well-established programs, partnerships between industry and academia are not common.

In addition to directly funding academic research projects, industry could provide a number of other opportunities. There could be an exchange of faculty and industry in internship settings in order to share expertise, students could benefit by working on real problems [23], industry could serve on Department advisory boards.
6. CONCLUSION AND FUTURE WORK

- This paper have provided an overview of computer security education.
- It presented government initiatives and other events from the past eight years, examined the current state of academic progress and discussed future objectives for promoting security within CS along with the perceived barriers to success.
- There are a number of areas that need to be addressed in order for security education to progress.
- A lack of research funding in academic programs appears to be a major roadblock to the creation of a viable national security program.
- Faculty training along with institutional support appears to be a problem for programs that lack any faculty with a background in security.
6.1 FUTURE WORK

There is a strong need for a survey of CS and IT departments to determine current status, future plans and needs for security education. The University of Idaho is planning to survey schools that have mapped their curriculum to the CNSS 8 40XX training standards to get feedback on their experience mapping their curriculum to the 40XX criteria. However this is intended to be a targeted survey and not a comprehensive survey of all CS departments. Other projects that would benefit security education include:

- An academic curriculum standard for both undergraduate and graduate programs
- Integration of computer security into accreditation programs (e.g. ABET)
- Support for schools beginning security programs
- Curriculum help and mentorship from established programs
Those who work in both security and education see promotion of security education for all graduating CS majors as one of the few concrete steps we can take that could favorably improve the state of cyber security.

Ultimately, security education should increase the level of competence in our developers to produce better quality, more robust systems capable of surviving most disruptions, intentional or otherwise.
AN ANIMATED SIMULATOR FOR PACKET SNIFFER

Xiaohong Yuan, Percy Vega, Jinsheng Xu, Huiming Yu, and Stephen Providence
Department of Computer Science, North Carolina A&T State University, 1601 East Market St., Greensboro, NC 27411, Email: {xhyuan, jx, ucshmyu, svp} @ncat.edu; Phone: (336)3347245. *Everett Consulting Corp., 320 McKinley St. #10, Hollywood, FL 33019, Email: percyvega@gmail.com Phone: (954)7325664
1. INTRODUCTION

- Visualization has been used in computer science education in the fields of algorithm, computer networks, computer architecture, and operating systems [1-4].
- The surveys of computer science educators conducted by the Working Group on “Improving the Educational Impact of Algorithm Visualization” suggest a widespread belief that visualization technology positively impacts learning [5].
- It seemed that computing educators are convinced that visualization can make a difference in helping learners better learn the concepts.
- It has become increasingly important to provide today’s computing students with training and education in security issues.
- Universities have tried to incorporate security issues into computer science curriculum. In some computer science programs, some security topics are incorporated into existing courses. Others developed new, required courses to the computer science curricula [6, 7, 8].
- Resources for teaching computer and information systems security at the undergraduate level will be useful for computer science educators [9].
This paper presents a software tool that demonstrates the security concept of packet sniffer and related computer networks concepts through animation.

It is intended to be used in an undergraduate level computer security course or a computer network course.

To make this tool more beneficial, exercise problems are designed to improve learner’s involvement in learning with this tool [10].

There is a believe this visualization tool will help students better understand the packet sniffer and network concepts, improve student participation, and make teaching and learning more enjoyable.
Packet sniffer is a program that captures all of the data packets that pass through a given network interface, and recognizes and decodes certain packets of interest.

A packet sniffer is sometimes referred to as a network monitor, or network analyzer. It is normally used by network or system administrator to monitor and troubleshoot network traffic. However, it is sometimes also used by malicious intruders for illicit purpose such as stealing a user’s password or creditcard number [11].

Typically, a computer in a network would only capture data packets that are intended for it and ignore packets that are not intended for it. However, if its network interface is configured into promiscuous mode, the computer is capable of capturing all packets traversing the network regardless of the destinations of the packets. A packet sniffer can only capture packets within a given subnet.
There exist various commercial and free packet sniffer tools.

- **Ethereal** is an open source tool for troubleshooting, analysis, software and protocol development and education. It can capture data from a live network connection, or read from a capture file. Captured network data can be browsed via a GUI, and more than seven hundred protocols can currently be dissected.

- **AnalogX PacketMon** is a fast and simple to use network monitor that captures IP packets. It uses advanced rules for filter [13].

- **Network Probe** [14] is a tool for traffic-level network monitoring, analysis and visualization. It provides full graphical representation and detailed statistics of the traffic and the type of data traveling across the network. It allows the user to isolate traffic problems and congestions. It also allows the user to search, sort, and filter network traffic information by protocols, hosts, network interfaces, etc.
3. THE PACKET SNIFFER SIMULATOR

- The packet sniffer simulator demonstrates visually how a packet sniffer works in a local area network environment, and how data packets are encapsulated and interpreted while going through the protocol stack.
- The packet sniffer simulator consists of a suite of five demos. Demo I to IV progressively demonstrate how a packet sniffer works.
- The simulation is based on a network with two subnets. The two subnets are connected with a router, and each subnet has a hub. The first subnet has a star topology and the second subnet has a bus topology.
- Demo V shows a protocol stack and how a data packet is encapsulated while going down the protocol stack at the source computer, and interpreted while going up the protocol stack at the destination computer.
The learning objectives of this packet sniffer simulator are:

a) Explain the differences between a hub, a bridge/switch, and a router

b) Explain bus and star topology

c) Explain how a data packet is transmitted in a local area network

d) Explain the purpose of “promiscuous mode” of a network interface

e) Explain what a packet sniffer does, and how it works.

f) Explain the encapsulation and de-encapsulation process of a data packet while going through the protocol stack
Macromedia Flash MX Professional Edition was selected to implement the demos of the packet sniffer simulator.

The main environment in which Flash animations can run is web page, as a Flash Applet. These animations can also run as a standalone application.

The only requirement for both environments is to download and install Macromedia Flash Player which is freely available.

The following sections explain the five demos, and the network and security concepts they demonstrate.
3.1 DEMO I: DIRECT PATH

- Figure 1 shows the user interface of Demo I. Demo II, III, and IV have the same user interface as Figure 1.
- **The user interface can be grouped into four components.**
  - The top-left component “Demo Sequence”, is a textbox that lists the names of the five demos. It allows the user to select the demo he wants to view.
  - The top-right component “Description Messages”, is a text-area that briefly describes the animation.
  - The bottom-right component shows the network architecture. It shows two subnets connected by a router. The subnet to the left has a star topology; the subnet to the right has a bus topology. Each subnet has a hub connecting the computers. The computers are numbered from 0 to 8.
  - The bottom-left component displays the data for simulation. The data used in the simulation is the source address and the destination address of a data packet. Three buttons are provided, which allow user to select from loading default data from an input file, or generating input data randomly, or entering data manually.
The input data are displayed in the table that has two columns.

1. Under the first column “From” are source addresses of the data packets.
2. Under the second column “To” are destination addresses of the data packets.
3. Under the two-column data table, are a “play” button (an arrow within a square), and a checkbox “Play continuously”.
4. If the checkbox “Play continuously” is checked and the “play” button is clicked, the animation will run from Demo I through Demo IV sequentially and go through all the data pairs in the data table sequentially in each demo.
5. If the “Play continuously” button is not checked and the “play” button is clicked, it will only show one step of the animation, i.e., the animation for one data set. The user can step through the simulation by repeatedly clicking the “play” button.
6. At each step of the simulation, the “description messages” text-area is updated to reflect what is going on.
Figure 1. Demo I: Direct Path

Dem Seqeance
- Direct Path
- Real Path
- Promiscous Mode
- Packet Sniffer
- Telnet over TCP/IP

Description Messages
A client on PC1 sends a packet to the destination PC8 that is named as Packet 1-8. This packet may travel through many different nodes of the network to arrive at PC8.

Demo 1 shows the final path that Packet 1-8 traveled from PC1 to PC8.
Demo I displays the path a data packet from a source goes through to reach the destination.

A data packet is represented as an oval shape, labeled by the source and destination numbers.

For example, Figure 1 shows a data packet, Packet 1-8 moving from computer 1 to the hub. The packet will go through the hub to the left, the router in the middle, the hub to the right, and finally arrive at computer 8.

Keep in mind that, since hub is used in the two subnets, the real path that Packet 1-8 traversed is not the same as the direct path.

Demo II demonstrates the real path of a data packet in the simulated network.
3.2 DEMO II: THE REAL PATH

- Since the computers in the subnet to the left are connected by a hub, all traffic can be seen by all computers on the subnet.
- The same is true with the subnet to the right which has the bus topology.
- A network segment that is not switched or bridged (i.e., connected through a hub) is called a common collision domain.
- Physically, any signal sent across a common collision domain reaches all attached computers.
The network interface hardware detects the electrical signal and extracts a copy of the frame.

It checks the address of each incoming frame to determine whether it should accept the frame.

The network interface hardware compares the destination address in the frame to the computer’s physical address (also called hardware address, or media access address).

- If the destination address in the frame matches the computer’s physical address, the interface hardware accepts the frame and passes it to the operating system.
- If the destination address in the frame does not match the computer’s physical address, the interface hardware discards the frame and waits for the next frame to appear [15].
Figure 2 shows that, when computer 1 generates a data packet, the data packet is captured by computer 0, 2 and 3 in the same subnet. Meanwhile, the data packet is forwarded to the router in the middle of the network.

The data packet is forwarded to the subnet to the right by the router, and all the computers attached to the bus receive Packet 1-8. Only computer 8’s network interface accepts Packet 1-8, and the other computers discard Packet 1-8.

This is depicted in Figure 3, in which the comment “mine” pointing to computer 8 indicates that the data packet is accepted by computer 8’s network interface, and the comment “not mine” indicates the data packet is discarded by the computer’s network interface.
Figure 2. Demo II (a): The Real Path When a Packet is Sent by the Source Computer
If the hub in the subnet to the left is replaced by a switch or bridge, and the computers in the subnet to the right are connected to a switch or bridge too, then the path Packet 1-8 traverses will be the same as Demo I.

By comparing Demo I and Demo II, the concepts of repeater, hub, bridge, switch and router can be reviewed.
Figure 3. Demo II (b): The Real Path When a Packet is Received by a Destination
A repeater is a hardware device used to extend a network cable. It has two ports.

A repeater receives signal from one port, regenerates the signal and sends out to the other port.

A hub is a multi-port repeater. A hub receives signal on one port, and regenerates the signal and sends the signal out to all the other ports.

Repeaters and hubs work at the Physical layer of the OSI model.
A bridge works at the data-link layer of the OSI model. It examines each incoming packet.

First the media access address (MAC address) of the sender and the port number through which the packet enters are added into the routing table of the bridge.

Then the MAC address of the recipient is looked up from the routing table to determine which port should the packet be forwarded to.

- If the recipient’s MAC address is in the routing table, then the port number is looked up and the packet is forwarded to that port.
- If the recipient’s MAC address is not in the routing table, the bridge sends the signal to all ports except for the one where it was received.
A switch is a multi-port bridge. It has the functions of a bridge, but uses a dedicated processor to implement the function, so it is faster than traditional software based bridges.

A router works at the network layer of the OSI model. It uses network addresses (for example, IP address) to determine how to forward a packet [16].
Normally, a computer’s network interface hardware checks the destination address of each incoming frame to determine whether it should accept the frame. It discards a frame whose destination address does not match its physical address.

However, a computer’s network interface hardware can be configured by software into promiscuous mode, which overrides the conventional address recognition.

Once in promiscuous mode, the network interface does not check the destination address, but accepts all frames. The network interface simply places a copy of each frame in the computer’s memory and informs the CPU about the arrival of the frame [15].
Figure 4. Demo III: Promiscuous Mode

PC5’s network adapter has been set to Promiscuous (P) mode, which means that any packet that arrives at PC5 will be made available to its applications by copying the packet to PC5. Because the Packet Sniffer is not enabled or is not installed in PC5, the packet is discarded by PC5.
Figure 4 shows the result of an animation of data packet transmission with Computer 5 configured into promiscuous mode.

When Packet 1-8 was transmitted on the common bus in the subnet to the right, computer 5 accepted the packet even though the packet was not addressed to it.

However, since there was no packet sniffer installed on computer 5 to process the packet, the packet was simply discarded by the operating system of computer 5.

Notice the difference between the “DISCARD” comment and the “not mine” comment. In the case of “not mine”, the computer’s network interface hardware discards the data packet because the destination address is not the same as the physical address of the computer; whereas in the case of “DISCARD”, the data packet is accepted by the network interface hardware but is discarded by the operating systems of the computer.
3.4 DEMO IV: PACKET SNIFFER

- Figure 5 shows the result of an animation when computer 3 has a packet sniffer installed.
- The network interface on computer 3 is also configured into promiscuous mode.
- When Packet 1-8 was sent out to the network, computer 3 captured the package and accepted it, even though the packet was not addressed to computer 3. This is indicated by the "mine" comment in Figure 5.
- The packet sniffer then examines the content of the data packet according to its configuration. The packet sniffer can be configured by the user to determine what fields to examine and what information it keeps. For example, a packet sniffer can be configured to examine all frames originated from a particular computer, or all TCP/IP traffic, or gather general traffic statistics.
Figure 5. Demo IV: Packet Sniffer

PC3's network adapter has been set to Promiscuous (P) mode, and the Packet Sniffer installed in PC3 is enabled. Any packet that arrives at PC3 will be processed by the Packet Sniffer application. The Packet Sniffer analyzes information in the packet.

**Demo Sequence**
- Direct Path
- Real Path
- Promiscuous Mode
- Packet Sniffer
- Telnet over TCP/IP

**Description Messages**

**Load Default Data**
**Generate Randomly**
**Enter Manually**

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>
Consider a packet sent from computer 4 to computer 7. When Packet 4-7 reaches the router, it will not be forwarded to the subnet to the left, so computer 3, which has the packet sniffer installed on it, will not be able to capture Packet 4-7.

Similarly, if the packet sniffer is installed in one of the computers in the subnet to the right, it would not be able to capture data traffic going through the subnet to the left.

A packet sniffer only works in a common collision domain.
3.5A DEMO V: TELNET OVER TCP/IP

- Demo V demonstrates how a data packet is encapsulated and de-encapsulated while going through the protocol stack. It assumes a Telnet application sending data packets over a network with TCP/IP protocol.
- Figure 6 represents three computers (PC0, PC1 and PC2) connected to a hub. Each computer is represented by a rectangle.
- In each rectangle a protocol stack of five layers are displayed.
- The five layers are: application, transport, network, data link and physical layer.
- The animation shows a data packet generated at the application layer at PC0 being encapsulated while moving down through the protocol stack, and being de-encapsulated while moving up through the protocol stack at PC1 (Figure 7 and Figure 8).
- The user can step through the animation by clicking the “play” button repeatedly, or run the simulation continuously by checking the checkbox “Play Continuously”.
Figure 6. Demo V (a): A Data Packet Generated by PC 0 at Application Layer
Figure 7. Demo V (b): The Encapsulation Process
Figure 8. Demo V (c): The De-Encapsulation Process
Encapsulation means that, when the data packet moves down through the protocol layers, a header (and a trailer sometimes) is added at each protocol layer.

- The transport layer adds a header that contains the source and destination port numbers. The network layer adds a header that contains the source and destination IP addresses and the transport protocol type.
- The data link layer adds a header and a trailer with source and destination physical addresses and the network type. And the physical layer converts the frame generated by the data link layer into bits.
- On the destination computer, the de-encapsulation process occurs. The headers (or trailers) are removed in reverse order.
In Figure 8, the data frame is transmitted to PC1 and PC2.

At the data link layer, the network interface hardware of PC2 recognizes the destination address of the incoming frame, and finds out it is not addressed to PC2, so PC2 discards the data packet at the data link layer. Whereas PC1’s network interface hardware recognizes that the data packet is addressed to PC1. So the frame is forwarded to the operating systems and is further de-encapsulated until it reaches the application layer.
THE SCIENCE OF INFORMATION PROTECTION

Daniel F. Warren
Naval Postgraduate School
The presentation of Information Protection material can be improved in two important ways.

First, if the material is arranged in a systematic/scientific fashion it can show how all the various pieces fit together and it can also demonstrate completeness by showing that all threats are addressed.

Second, if each protection technique is preceded by a clear description of the threat that it addresses learning is significantly enhanced because the protection technique is motivated.

This paper presents an information threat model that 1) arranges the material in a scientific/systematic fashion and 2) facilitates a threat-first presentation of Information Protection techniques.
1. INTRODUCTION

The threat model presented here is based on the following seven threat classes. These threat classes are mutually disjoint and together constitute all threats.

1. The Non-Human threat
2. The Human Error threat
3. The Authorized Person threat
4. The Unauthorized Insider threat
5. The Outsider Attacking Enterprise Data on the LAN threat
6. The Outsider Attacking Enterprise Data Outside the LAN threat
7. The Malicious Software threat
1.1 THE FOLLOWING SECTIONS DESCRIBE:

- why most Information Protection treatments fail to be scientific/systematic,
- why most Information Protection treatments fail to motivate the material,
- the nature of the threat model,
- how the model can be used to arrange Information Protection material in a scientific/systematic fashion,
- how the model facilitates a threat-first presentation of Information Protection techniques and
- why the classes of the threat model constitute all threats.
2. Why Current Information Protection Treatments Fail to be Scientific/Systematic

- A recent ad for a new information security book announces that the book covers the following major themes: Cryptography, Access Control, Protocols and Software.

- A scientific minded reader is bound to ponder:
  - Do these topics cover all facets of Information Protection?
  - What are the relationships between these different themes?
  - Is there a significance to the given order?
In general, Information Security or Information Protection books present topics in a somewhat ad hoc manner that does not lend itself to answering these questions. This, in turn, lessens their effectiveness as teaching tools.

At a minimum every reader would like to know if an Information Protection book covers all aspects of information protection.

The ability to answer this question and the others posed above requires a systematic approach, which is a direct consequence of following the scientific method.

The Merriam-Webster on-line dictionary defines “scientific method” as: “principles and procedures for the systematic pursuit of knowledge involving the recognition and formulation of a problem, ...”
3. Why Current Treatments Fail to Motivate the Material

- Information Protection and First Aid are similar in the sense that they are both problem/solution disciplines. If a First Aid course is organized around solutions the approach is less than optimal and fails to motivate the material. Such a presentation might start with aspirin and describe what aspirins do and what aspirins don’t do. It might then present bandages and describe what they do and what they don’t do. And so forth.

- This material is better presented if it is organized around the problems. For acid indigestion do this and this and this. For bleeding do this and this and this. And so forth. When presented this way the student’s interest is first peaked by the problem. “Hmm, how is acid indigestion addressed?” Then when the solution is presented it has context and meaning. The student is waiting to hear the solution and is likely to better comprehend it because of the context.
Currently most Information Protection books are arranged around solutions, like Cryptography, Access Control, Protocols, etc. And often the context is ignored.

When execution domains are presented how often is the threat that they address also presented? If the student does not know what the problem is they are not going to be tuned into the solution.

From a pedagogical standpoint a better approach is to arrange Information Protection topics around the problems and then present the protection techniques that address them.

Since the threat model introduced here arranges Information Protection material around the problems it naturally motivates an engaging presentation of protection techniques.
4. THE THREAT MODEL

- The threat model presented here is based on a collection of seven threat classes that are mutually disjoint and together constitute all threats. These classes are based upon a typical enterprise computing situation, like the one shown in Figure 1.

- This situation includes two Local Area Networks (LANs) that are physically far apart so they communicate via a Wide Area Network (WAN), like the Internet.

- The goal of the enterprise Information Protection policy is to protect enterprise information from unauthorized observation, modification and denial of availability and to protect the authenticity of source information for enterprise messages.
Figure 1. Typical Enterprise Computing Environment
Persons that are authorized to access enterprise information (such as employees) are called “insiders.”

Persons that are not authorized to access enterprise information are called “outsiders.”

The LAN systems and network infrastructure are assumed to be protected by typical LAN physical and personnel security measures.

The model refers to information residing on the LAN systems or on the LAN wired networks as “information on the LAN” and information that is residing on systems and networks outside the LAN or in a wireless portion of the LAN as “information that is outside the LAN.”
The seven threat classes of the model are:
(1) Non-Human threat
(2) Human Error threat
(3) Authorized Person threat
(4) Unauthorized Insider threat
(5) Outsider Attacking Enterprise Data on the LAN threat
(6) Outsider Attacking Enterprise Data Outside the LAN threat
(7) Malicious Software threat (orchestrated by participants of classes (4) or (5))
Figure 2. Graphical representation of the seven threat classes
Examples threats in each threat class are given below.

<table>
<thead>
<tr>
<th>Threat class</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Non-Human</td>
<td>Fire</td>
</tr>
<tr>
<td>(2) Human Error</td>
<td>An employee accidentally deletes a file</td>
</tr>
<tr>
<td>(3) Authorized Person</td>
<td>A researcher sells some research data to a competitor</td>
</tr>
<tr>
<td>(4) Unauthorized Insider</td>
<td>A mail clerk circumvents the file Access Control mechanisms and obtains some research data</td>
</tr>
<tr>
<td>(5) Outsider Attacking Enterprise Data on the LAN</td>
<td>Someone in Iceland breaks into the research computer and obtains research data</td>
</tr>
<tr>
<td>(6) Outsider Attacking Enterprise Data Outside the LAN</td>
<td>Someone in Florida intercepts microwave traffic and captures enterprise data that is flowing across the Internet</td>
</tr>
<tr>
<td>(7) Malicious Software</td>
<td>Someone in Iowa plants a Trojan horse in a music sharing program</td>
</tr>
</tbody>
</table>
5. How the Model Supports a Scientific/Systematic Treatment of Information Protection Material

- Table 2 lists the threat classes and general categories of protection techniques that address each threat class.
- Viewed this way it is possible to see how the categories of common Information Protection techniques (Passwords, ACLs, Firewalls, Cryptography, etc.) fit into the big picture of Information Protection. And since these threat classes contain all threats complete protection against all threats is achieved if each threat class is adequately addressed.
<table>
<thead>
<tr>
<th>Threat Class</th>
<th>General Protection Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Human (1)</td>
<td>Physical Security</td>
</tr>
<tr>
<td>Human Error (2)</td>
<td>Error Security</td>
</tr>
<tr>
<td></td>
<td>Training</td>
</tr>
<tr>
<td></td>
<td>Secure Defaults</td>
</tr>
<tr>
<td></td>
<td>Simple interfaces</td>
</tr>
<tr>
<td>Authorized Person (3)</td>
<td>Personnel Security</td>
</tr>
<tr>
<td></td>
<td>Personnel Security (background checks)</td>
</tr>
<tr>
<td></td>
<td>Separation of Duties</td>
</tr>
<tr>
<td></td>
<td>Audit</td>
</tr>
<tr>
<td>Unauthorized Insider (4)</td>
<td>Operating System Security</td>
</tr>
<tr>
<td></td>
<td>Identification and Authentication</td>
</tr>
<tr>
<td></td>
<td>File Access Control</td>
</tr>
<tr>
<td></td>
<td>Memory Protection (Process Separation, Rings)</td>
</tr>
<tr>
<td>Outsider Attacking Enterprise Data</td>
<td>Perimeter Security</td>
</tr>
<tr>
<td>on the LAN (5)</td>
<td>Firewalls (Network Traffic Access Control)</td>
</tr>
<tr>
<td>Outsider Attacking Enterprise Data</td>
<td>Communications Security</td>
</tr>
<tr>
<td>Outside the LAN (6)</td>
<td>Cryptographic Access Control</td>
</tr>
<tr>
<td>Malicious Software (7)</td>
<td>Malicious Software Security</td>
</tr>
<tr>
<td></td>
<td>Trojan horses: MAC policies</td>
</tr>
<tr>
<td></td>
<td>Worms: Assurance techniques</td>
</tr>
<tr>
<td></td>
<td>Viruses: Ad Hoc Antivirus techniques</td>
</tr>
</tbody>
</table>
The table below identifies three techniques that are frequently used to address failures in Access Control mechanisms.

<table>
<thead>
<tr>
<th>Access Control Failures</th>
<th>Design and Development Principles that Promote a Sufficient Degree of Assurance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Defense in Depth</td>
</tr>
<tr>
<td></td>
<td>• Intrusion Detection</td>
</tr>
</tbody>
</table>
6. HOW THE MODEL SUPPORTS A THREAT-FIRST PRESENTATION OF INFORMATION PROTECTION TECHNIQUES

- A straightforward application of the threat model to the presentation of Information Protection material could simply allocate a section or chapter to each threat class, such as is done in Table 4.

- In order to instill interest in the student the presentation of each protection technique should begin with the threat that is addressed by the protection technique.
<table>
<thead>
<tr>
<th>Chapters</th>
<th>General Information Protection Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Non-Human Threat</td>
<td>Physical Security</td>
</tr>
<tr>
<td>2. Human Error Threat</td>
<td>Error Security</td>
</tr>
<tr>
<td>3. Authorized Person Threat</td>
<td>Personnel Security</td>
</tr>
<tr>
<td>4. Unauthorized Insider Threat</td>
<td>Operating System Security</td>
</tr>
<tr>
<td>5. Outsider Attacking Enterprise Data on the LAN Threat</td>
<td>Perimeter Security</td>
</tr>
<tr>
<td>6. Outsider Attacking Enterprise Data Outside the LAN Threat</td>
<td>Communications Security</td>
</tr>
<tr>
<td>7. Malicious Software Threat</td>
<td>Malicious Software</td>
</tr>
<tr>
<td>8. Access Control Failures</td>
<td>Access Control Failure Techniques</td>
</tr>
</tbody>
</table>
Hypothesize a system with no user Identification capabilities and show how an Insider user can access information that they are not authorized to access.

Hypothesize a system with a user Identification capability and a file access control mechanism (e.g., ACLs and ACL program) but no user Authentication capability and show how an Insider can access information that they are not authorized to access.
Hypothesize a system with user Identification and Authentication and a file access control mechanism (e.g., ACLs and ACL program) and
- consider storing the password information in the clear,
- consider storing the password information encrypted with conventional cryptography,
- consider storing the password information in a hashed format.
- Introduce the Dictionary Attack.
- Introduce the Brute Force Attack.
- Go over password selection guidelines
Hypothesize a system with user Identification and Authentication (I&A) and a file access control mechanism (e.g., ACLs and ACL program) but no process address space control and show how an Insider can directly read and write bytes of memory and access information that they are not authorized to access.

Hypothesize a system with user Identification and Authentication, a file access control mechanism (e.g., ACLs and ACL program) and process address space control but no ring mechanism and show how an Insider can bypass the file access controls and ultimately access information that they are not authorized to access.
7. WHY THE SEVEN CLASSES INCLUDE ALL THREATS

- The follow figures prove that the seven classes of the model do, in fact, include all threats. The proof involves repeatedly partitioning the set of all threats until the 7 classes of the model are derived.

- First, the set of All Threats is partitioned into Human-Based threats and Non-Human-Based threats.
Figure 3. Partitioning all threats into Human and Non-Human-Based threats
Then the set of Human-Based threats is partitioned into the set of Intentional threats and Accidental threats. Intentional threats are, by definition, performed by Untrustworthy Persons.

With the exception of Non-Human threats (fires, lightening, etc.) and Accidental Errors all threats are a result of Untrustworthy Persons.
Figure 4. Partitioning Human-Based Threats into Intentional and Accidental Threats
Untrustworthy Persons come in two varieties, those that are authorized to access the data and those that are not authorized to access the data. Thus, the set of Untrustworthy Persons is partitioned into Unauthorized Persons and Authorized Persons.
Figure 5. Partitioning Untrustworthy persons into Unauthorized and Authorized sets
Unauthorized Persons come in two varieties, those that are Insiders and those that are Outsiders. Thus, Unauthorized Persons are partitioned accordingly.
Figure 6. Partitioning Unauthorized Persons into Outsiders and Insiders
Outsiders can threaten enterprise data in two different places, in the environment outside the LAN or on the LAN. Thus, Outsiders are partitioned accordingly.
A seventh threat class (Malicious Software) is introduced because Unauthorized Insiders and Outsiders Attacking Enterprise Data on the LAN can attack LAN information in two ways, directly and indirectly.

Direct attacks are prevented by appropriate Access Control techniques. This is due to the nature of Access Control, it grants access to authorized persons and denies access to unauthorized persons and both the Unauthorized Insiders and the Outsiders Attacking Data on the LAN are not authorized for the targeted data.

Indirect attacks are generally not addressed by typical Access Control techniques. Indirect attacks use Malicious Software to attack “from the inside.” This type of attack is so fundamentally different it is broken out into its own threat class. Access Control techniques keep unauthorized users from gaining direct access, they do not prevent a victim from unknowingly sending their files to a website in Eastern Europe when they execute a Trojan horse program. If the victim is authorized to send data to a website Access Control cannot address this problem.
Figure 8. Introducing the Malicious Software Threat Class
Since the seven classes above are derived by partitioning the set of all threats they must contain every information threat.

As such, adequately addressing each threat class provides complete coverage against all threats.
The given threat model provides a systematic way for presenting Information Protection topics.

Systematic presentations, like the one proposed here, enhance learning by showing how all the pieces fit together and by establishing a sense of complete coverage against all threats.

The use of the threat model presented here also promotes learning because it organizes the material around the threats that provides context and necessity for each protection technique.
NETWORK SECURITY AND INFORMATION ASSURANCE AT JACKSON STATE UNIVERSITY

Houssain Kettani
Department of Computer Science, Jackson State University, Jackson, Mississippi, 39217
This paper share the story of Network Security and Information Assurance education at Jackson State University. It narrate its start, collaboration with another university, and steps towards establishing a Center of Academic Excellence in this domain.
1. INTRODUCTION

- The example of the Internet and communication networks in the progress of our lives nowadays is water. We cannot imagine our economy, education, communication with others, defense, pleasure, etc, without the Internet.
- E-mail, HTTP, WWW, all are thought nowadays to be necessities, although we know for sure that Adam and Eve did not have them!
- Consequently, the protection of information becomes more and more challenging. Accordingly, the interest in Network Security and Information Assurance (NSIA) is growing day by day and the latter is posed to be in the heart of Computer Science education.
Acknowledging this fact, the Department of Computer Science at Jackson State University (JSU) moved into providing NSIA as part of its curriculum.

Initially, a network security course for undergraduates and graduate students was offered once a year in collaboration with Mississippi State University (MSU), Mississippi State, Mississippi.

Thus, in spring 2002, this course was offered as a distance learning class.

Eventually, the course was offered in spring 2004 by Dr. H. Kettani, a faculty member of the Department of Computer Science at JSU and is being offered in every spring semester.
Meanwhile, the department has received two generous grants from the National Science Foundation to enhance the department’s NSIA program and establish a Center of Academic Excellence in NSIA. As a result, two undergraduate and two graduate students have been supported and are performing very well in their NSIA career.

This paper elaborates on the journey of the Department of Computer Science at JSU to establish NSIA awareness in its program, our success, and future plans.
2. ABOUT JACKSON STATE UNIVERSITY

- Jackson State University (JSU) is one of the nation's largest and prominent Historically Black College/University (HBCU) located at Jackson, the capital of the state of Mississippi. As of fall 2003, Jackson State University, enrolled 7,815 students. Of this total enrollment, 63% are female, 81% are undergraduates, and 94% are African Americans [1].

- The College of Science, Engineering, and Technology at Jackson State University provides an excellent opportunity for large numbers of the African Americans, both men and women, to obtain engineering education. As of fall 2003, this college enrolled 1,655 students, which is 23% of the total enrollment of the university. Of this college's total enrollment, 52% are female, 87% are undergraduates, and 94% are African Americans [1].
The Department of Computer Science offers both Bachelor of Science and Master of Science degrees. The Bachelor of Science degree program is accredited by the Accreditation Board for Engineering and Technology (ABET).

The Department of Computer Science is also a leading producer of African Americans with the Bachelor of Science degree in Computer Science.

As of fall 2003, this department enrolled 330 students, which is 20% of the total enrollment of the College of Science, Engineering, and Technology. Of this department's total enrollment, 38% are female, 76% are undergraduates, and 84% are African Americans [1].
3. COLLABORATION WITH MISSISSIPPI STATE UNIVERSITY

Mississippi State University (MSU) has developed a strong NSIA program and is two hours drive north east of JSU. Thus, when we were thinking of developing our NSIA program, we decided that it would be a good idea to learn from MSU’s experience and collaborate with its Computer Science Department.

Accordingly, a network security course for undergraduates and graduate students was offered once a year in collaboration with MSU as distance learning class. Thus, in spring 2002, the course was offered at MSU by Dr. R. Vaughn, a faculty member of the Department of Computer Science at MSU and was broadcast to JSU. In spring 2004, the course was offered as semi distance learning class, where Dr. Vaughn alternated between MSU and JSU. In addition, this course had a laboratory that was offered at JSU and supervised by two JSU graduate students as teaching assistants. The adopted lab experiments followed the lab manual that was developed at MSU.
In response to a National Science Foundation (NSF) request for proposals for a Scholarship for Service (SFS) grant, and in collaboration with MSU, JSU was awarded in August 2002 two SFS grants of $350,000 for a period of 4 years.

The purpose of these grants is to increase NSIA awareness on campus. With these grants we were able to support two undergraduate students and two graduate students for duration of two years.

These students were required to have an internship in NSIA domain at a government institution of their choice, and work for two consecutive years at such institution after the completion of their degree.
Note that the students that supported did not have trouble in finding such internships.

In the past two summers, we had one graduate and one undergraduate student who worked at the Federal Bureau of Investigation (FBI) at Washington, DC, a graduate student who worked at the Federal Administration Agency (FAA) at Washington, DC, and an undergraduate student at NASA Stennis Space Center, Mississippi.

In May 2005, first supported student has earned her Masters of Science and started working at another Government agency.
The grant also helped in sending students and faculty members to attend educational and training conferences and workshops in NSIA domain.

For example, in 2004, Dr. Kettani was sent to two conferences organized by SysAdmin, Audit, Network, and Security (SANS) Institute.

Dr. Kettani also attended the Sixth Workshop on Education in Computer Security (WECS6). Thus, in spring 2005, he instructed the course independently from MSU. The course still has a lab, and the enrollment is a dozen to a score of students.
5. TOWARDS A CENTER OF ACADEMIC EXCELLENCE

- The first step in establishing a Center of Academic Excellence in NSAI is to map the courses that we offer in NSIA domain to two standards of the Information Assurance Courseware Evaluation (IACE).

- Accordingly, in February 2004, the Review Committee of IACE has validated the mapping at 100% for the Committee on National Security Systems (CNSS) National Standards 4011 and 4014 Entry Level.

- Consequently, JSU has received recognition and a certificate during the CNSS Annual Conference, Norfolk, Virginia, April, 2004. Consequently, we have applied to the 8th annual NSA and DHS for the Centers of Academic Excellence in Information Assurance Education Program Offering in December 2005.
ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation (NSF) and the Naval Postgraduate School, Monterey, California. The authors are thankful to Dr. R. Vaughn from Mississippi State University (MSU), Mississippi State, Mississippi, for his collaboration and dedication to initiate our NSIA program.
TWO SUCCESSFUL MINIPROJECTS IN AN OVERVIEW INFORMATION ASSURANCE COURSE

Judith L. Gersting
University of Hawaii at Hilo
1. INTRODUCTION

In the Computer Science Department at the University of Hawaii at Hilo, they have recently introduced an undergraduate Information Assurance course. This is a senior-level elective course for computer science majors, and is intended to provide a general overview of the field.

Information assurance can be somewhat overwhelming to teach because its scope is so broad. It touches on many areas of computer science (programming, operating systems, database, networks), as well as mathematics (cryptology), and management, legal, and ethical issues.

But it is a very important topic, one in which there is a great deal of student interest and for which there is no end of related items to be found in the news.
The author criteria for success, which were:

a. Students learned something that was new to them.

b. A "real world" security problem or issue was explored.

c. Students found the activities enjoyable.
2. STEGANOGRAPHY

- Steganography is the art of hiding secret information in plain sight, and in various forms, it dates back to ancient times.
- Today it generally involves highjacking the least significant bits of an image file to store secret information such as a text file, and with the ready transmission of images over the Web, it is a potential tactic for instantaneously broadcasting secret information in plain sight around the world.
The assignment package the author put together on this topic involved the following student activities:

A. Students reported on historical uses of steganography.
B. Students used a free steganography software tool called S-Tools [6], which allows for least significant bit encryption of data within an image file. In a closed lab exercise, students explored his tool, answered several questions about how the tool works, and investigated four copies of a 900 KB image file it was prepared. Each looked the same; one was nothing but the image and the other three held increasingly larger files. Students were impressed with the fact that a 650 KB text file - an entire book chapter - could be embedded in this image and appear quite undetectable.
The author provided information on the BMP file format as background for a programming assignment that consisted of three parts. Students worked in small teams to write (in C++) three programs that constitute a simple version of S-Tools to meet the following specifications:

i. As a first pass, read in and write out a BMP image file. The output image should, of course, look like the input image.

ii. Read in an image file and a text file and encode the text file sequentially in the least significant bits of the image file.

iii. Read in a steganographic image file containing text encoded as in part ii and recreate the original text file.
D) In a closed lab session, each team used a BMP image file and a text file of their choosing, encoded the text file in the image file, and passed the steganographic image file to another team, who attempted to decode it and recover the original text file. This proved a bit more difficult than anticipated because, while each team had tested their programs on their own files, some of the programs (either the encoding or the decoding) didn't work correctly when paired with another team's program. To pin down the errors, the teams began trading text and image files and by the end of the hour one team still had an encoding error (later fixed) but all other code was working properly.
E ) One student took this project a little farther. Using two image files, one quite complex and one rather simple, he inserted random bits into each image, first randomizing just the low order bit, then the two low order bits, 3 bits, 4 bits, up to 7 of the 8 bits, and wrote out the image file for each case. Two interesting observations could be made. One was that even with 7 out of every 8 bits mangled, the original images were still discernable. The other was that, contrary to what one might expect, the more complex image absorbed the mangled bits less successfully than the simple image, that is, slight variations or degradations appeared visible in the complex image with fewer mangled bits than in the simple image. This little experiment will be included as part of the overall steganography assignment package when the course is next taught.
3. SECURITY PLANNING

- The second assignment package, which came near the end of the semester, involved management issues, specifically security planning. A security plan for an organization should address the following:
  - Risk analysis
  - Security policy
  - Who is accountable for what
  - Timeline for change
  - Structure for periodic revision
Risk analysis involves making an inventory of the organization's assets, determining how each might be vulnerable, how each might be protected, and the cost/benefit ratio of implementing such protection.

The security policy, based on the risk analysis, spells out what the organization's security decisions are, what assets are accessible to whom and for what purpose, and what protections are in place.

The security plan will also include assignment of responsibilities for security measures, a timeline for implementing new security controls, and a mechanism to ensure periodic review and revision of this whole process.
3.1 THE OCTAVE-S GUIDELINES

The OCTAVE method [1] was developed at the Software Engineering Institute at Carnegie Mellon University. It involves specific steps for building a security plan for large organizations, i.e., those with over 300 employees. (More information on the OCTAVE method can be found in [2].) OCTAVE-S was developed later as a more streamlined version suitable for small organizations.

The OCTAVE-S Implementation Guide is a 10-volume series that includes documentation and guidance for use of the OCTAVE-S system. Obviously a complete application of the OCTAVE-S system would be a major effort, and training is recommended. Volume 10 of the series, however, is an example scenario of the OCTAVE-S methodology applied to a fictitious medical facility called MedSite. This is the document made available to students for this assignment.
Most valuable in this example scenario are the worksheets illustrating asset identification, impact evaluation, identification and evaluation of current security practices, identification of critical assets and their risk profiles, protection strategies, recommendations, and action plans.
Part of the assignment statement given to students follows:

"Your job is to write a security plan for a fictional enterprise. Obviously, the Octave-S example in Volume 10 is more than is required for this assignment, but you should read it to gain some ideas for making lists and/or tables about your fictional enterprise. Your first task, of course, is to fill in some details of your fictional enterprise so that you can then identify its most important assets, determine in what ways and how likely they are to be vulnerable, estimate the costs of that vulnerability, know who is responsible for these assets, and so forth. Your report should be on the order of 12-15 pages, and should be an example of professional writing."
This was also a team project. Each team had a different fictional enterprise, namely:

- A credit union
- A regional airline
- An electric utility company
- A political campaign headquarters

Students had less than two weeks to complete the project.
3.3 THE RESULTS

- This was a very successful assignment in terms of student enjoyment, although they all wished they had had more time (and in this case this familiar student lament may have been justified).
- The fictional nature of their enterprises gave the students the freedom to "invent" assets, while the worksheet formats from the OCTAVE-S Volume 10 document helped to provide structure for their reports.
- All team papers were well-written, and used selected worksheet formats from the OCTAVE-S Volume 10 document.
- The political campaign team had the greatest difficulty identifying assets. The credit union team thought that their enterprise was rather dull.
The most interesting report came from the electric utility company, which early on decided to be a hydroelectric facility, and then found an actual risk assessment methodology document applicable to their enterprise [4].

By the time this assignment was completed, a graduating student in this course had his job offer in hand and had learned that one of his responsibilities was going to be to write a security plan for the organization hiring him. This certainly added interest in this assignment for him, and for his fellow students.
4. CONCLUSION

- The author objective after attending WECS6 was to again offer the overview security course, but to incorporate more hands-on activities or assignments than previously.
- Using material inspired from WECS6, the unit on steganography took about 3 hours of class and closed lab time.
- The unit on management issues took about 4 hours of class time, including time for presentation of team reports. Other material from WECS6 was filtered throughout some of the other course topics.
- Nothing specific was done for or with underrepresented groups in computer science, although the University of Hawaii at Hilo is considered a minority institution. No publications resulted from this work, but new materials for assignments and activities were developed.
- The IA course will be a part of our regular course rotation and will be offered every two years. We hope to continue to develop activities and assignments in various areas to further enrich student learning opportunities.
WECS6 FOLLOW-UP REPORT

Charles Anderson
Western Oregon University
This report is a summary of the author educational efforts in the field of computer and network security following his attendance at the WEC6 workshop and tutorials.
1. OBJECTIVES

- His initial objectives immediately following WECS6 were very modest:
  - to enhance the coverage of security topics for those classes in which typically discuss security - e.g., networking.
- He decided to offer an introductory class in security as an unpaid overload, and he began to discuss the possibility of expanding Computer Science (CS) major to include an emphasis in security.
- In preparation for that, he created a new block of course numbers (460 - 469) for security classes, and wrote course descriptions for two initial offerings.
2. COURSES TAUGHT

- In the 2004-05 academic year, he incorporated security or expanded the coverage of it in a number of classes that he usually teach.

- He also taught a new introduction to security course. Furthermore, as a direct result of the WECS6 conference, he added security to a new class, which initially seemed to have little to do with security.
2.1 EXISTING COURSES

This section discuss courses that he typically teach and how he has changed his presentation of security topics.
This is a first course in networking aimed primarily at his Information Systems (IS) majors (non-programmers). It is focused primarily on LAN networking.

From the WECS tutorial, he got some new ideas for presenting the networking topics (independent of security) from J. D. Fulp’s introductory talk – e.g., using a spreadsheet to construct a template to overlay on binary data to show the idea of fields within a packet.

This class includes a section on host-based and LAN oriented security which was reorganized and greatly clarified based on information from the entire workshop.

While teaching this class he had the epiphany that preparing students to understand network attacks should be a primary learning outcome of the course.
2.1.2 IS 452 – INTERNET – 3 UNITS

- This is the second class in networking, which is focused primarily beyond the LAN. It is a follow up to CS 350.
- In addition to general reorganization and clarification of security topics, he applied information and materials that he developed for Introduction to Information Assurance course (see below) to discuss topics such as VPNs and firewalls.
2.2 NEW COURSES

2.2.1 CS 459 - Introduction to Information Assurance - 3 units

This was a brand-new course that he taught as a direct result of the WECS workshop. Because it was new, it was an elective for the students.

Despite the fact that the students were not required to take the course, it was one of the largest/most popular courses. It was available to more senior CS and IS students - i.e., programmers and non-programmers.

On the first day after dealing with the typical administrative chores, he showed the *Strategic Cyber Defense* DVD that they got at the tutorial; the students got a kick out of the video.
He used *Corporate Computer and Network Security* by Panko as the textbook.

Topics included access control, physical security, attack methods, firewalls, host security, and some basic cryptography.
This class covers programming in C and assembly language. It is a rarely taught elective. The majority of the assignments were security-related.

Based on a talk by Everett Bull at the WECS conference, he got a copy of the textbook *Computer Systems: A Programmers Perspective* by Bryant and O'Halloran and gave the students two programming assignments from the book.

- The first was the “bomb defusing” exercise that teaches the students how to use a debugger and work with disassembled binaries - i.e., reverse engineering.
- The second assignment involved creating stack overflows in a program.
- In addition, as an exercise in algorithm design and optimization, he had the students write a brute-force password cracking program, and we compared the speeds of their implementations. The idea to do this was a result of a talk at the tutorial by one of the CISR discussing password strength and cracking times.
3. CURRICULUM DEVELOPMENT

- At WOU, Computer Science students select an emphasis or elective sequence as part of their major. An emphasis is composed of three senior-level courses in a given subject area.
Although these courses are nominally intended to be part of our Computer Science major, it is my intention to make as many of them available to our Information Systems majors as well.

Both majors have separate networking and operating systems courses, but the new courses accept prerequisites from either major. At this time, the only course/topic I see as exclusively for CS majors will secure programming because students majoring in IS often choose the major to minimize the programming that they must do.
3.1 NEW COURSES

- **Introduction to Information Assurance - CS 460:** this is a basic survey class. It will be an “official” version of the CS 459 class I taught last year and will cover a similar selection of topics. (The CS 459 course number I used before was just a temporary kludge to be able to teach the course without waiting for formal approval of an official course number.) I hope to make this a required course for all CS and IS majors, not just the CS majors who select the security emphasis.

- **Special Topics - CS 469:** this course was added immediately to allow flexibility to offer classes in more specialized topics before official course numbers are available. Once such course numbers are available, this course may be used for unique topics or seminars.
The previous two courses have already been approved. Department hasn’t finalized the other courses in the security emphasis, but I am considering the following possibilities.

- Network Attack and Defense: a discussion of tools and techniques for attacking systems and protecting them from such attacks.
- Computer Forensics: tools and techniques for forensic analysis of computer systems.
- Secure Programming: an introduction into secure programming practices - e.g., avoiding buffer overflows, SQL injection attacks, etc.
This section discuss various little topics related to WECS and his teaching endeavors.

At the conclusion of Introduction to Information Assurance class, he discussed the Scholarship For Service (SFS) program, and shared information about the graduate program at the Naval Postgraduate School.

One of his students, Justin Hoeckle, was very interested in SFS. He was accepted to a number of schools, and is now studying at Johns Hopkins. (He didn’t apply to NPS because of desire/need to be on the east coast.)
With regards to under-represented student groups, his university attracts a large number of students who are the first in their family to attend college.

Justin Hoeckle was one of these, and his success is due, in small part, to the security class he took from me. Although the state of Oregon doesn’t recognize Asians as minorities, they are quite rare on our campus, and one of them, Minh Nguyen, took all of my networking classes and the security class.
He was to Cornell for graduate school but was unable to attend due to a lack of financial aid. (As a citizen of Vietnam, SFS was not an option for him.)

He has not developed any significant new materials related to security, other than the usual assignments and PowerPoint presentations.
5. SUMMARY

- To summarize the results of his participation in WECS:
  - He is working to create a new security emphasis for CS majors.
  - He has taught the first course dedicated to security on their campus.
  - He has significantly refined his presentation of security in his existing courses and had the opportunity to incorporate security into courses that he never considered before.
  - He has discovered his own personal niche for teaching and learning: security.
REPORT ON THE DEVELOPMENT OF CSN 290

The Creation of an Introductory Network Security Course
Marina A. Cappellino
Genesee Community College
ABSTRACT:

Approximately two years ago, he spoke with administration at Genesee Community College about attending Network Security training. The Director of Computing was considering expanding our technology programs to include at least one course— if not an entire degree— on the topic of Information Assurance (this includes Network and Information Security).

Administration was supportive of this new initiative. At that point, he took the lead in developing an entirely new course entitled an Overview of Computer and Network Security.
The process of developing a course from the ground up was brand new to the author. Some of the questions he had to address were:

- What would be considered important material?
- How could I boil down the masses of information received into a one semester introductory security course?
- How would I ascertain the level of proficiency of students?
- What would be the outcomes of the course?
- Could the material covered in this course be immediately implemented by students at their place of employment?
- Would abuse of college property in the forming of hacking be a concern?
- Would administration continue to support this effort through sustained funding for training and additional equipment needs?
1. INFORMATION ASSURANCE OBJECTIVES FOR THE 2005-2006 ACADEMIC YEAR

- Upon the advice of the dean and looking at projected enrollments, they decided to offer the network security course in the Fall 2005 semester.
- This new topics course enrolled 10 students. Course objects, as required by a SUNY wide student learning outcomes initiative, were clearly stated in the syllabus as follows:
  1. Explain at least two reasons why network security is important.
  2. Demonstrate knowledge of basic network security principles.
  3. Identify and describe at least two risk mitigation strategies.
  4. Describe the daily tasks involved with managing and troubleshooting computer security technologies.
5. Document knowledge of how to create a secure computer networking environment

6. Explain the concepts involving authentication, along with at least two of the types of attacks and malicious code that may be used against your network, and at least two countermeasures that could be taken against such attack.

7. Define terms and concepts such as intrusion detection systems, firewalls, and cryptography

8. Discuss at least two types of scanning and analysis tools

9. Discuss at least two legal and ethical issues within the field
2. What was your process in incorporating WECS6 material and how much of the material did you include in the course?

- The author attended several training sessions, seminars and conferences on the topic of Network and Information Security, one of which was WECS6 training at the Naval Postgraduate School in Monterey.
- WECS helped me define the topics he would cover in the course and also gave him the opportunity to gain a better understanding of Network Security.
Upon learning that the WECS training included colleagues from four year colleges and universities, I was unsure of the amount of relevant information I would receive, which would be useful in an introductory community college course.

I felt that perhaps the information would be targeted at a higher level. As I progressed through the WECS training and conference, my concerns were alleviated. There were a number of sessions that directly assisted me preparing for my network security course. Those sessions which presented material beyond the scope of my course, were invaluable as well.

They afforded me the opportunity to gain a deeper understanding of network security. As a result, I went into my course with more confidence due to the newly gained knowledge.
While the book I selected, *Fundamentals of Network Security* by Eric Maiwald, covered an overview of many topics, there were certain topics I chose to cover in greater detail.

For those topics, I would refer to many of my new resources for the clearest and most concise way of explaining the information.

Professor Fulp's WECS notes were very helpful in the creation of supplemental slides on Firewalls.

In addition, his session on Information Assurance Education Pedagogy was very interesting and extremely applicable—not only to my network security course, but to other technology courses I teach.

I learned some helpful techniques for effectively teaching technology and the importance of a solid foundation in the basic concepts. His notes underscored the importance of clarifying for students, the terminology that can get so confusing so quickly.

Professor Fulp spoke on the topics of “principle of least privilege”, “hard on the outside vs. hard on the inside” and “defense-in-depth” all of which I covered in my course.
Professor Fulp’s seminars were not the only ones from which I obtained a wealth of information and ideas.

I also learned and incorporated concepts from Professor Dinolt’s presentation on Cryptography. Seeing the code for the “I Love You” virus was very interesting and made my lecture on IDS signatures easier to understand. The labs designed for my course were a more basic version of some labs I worked with at WECS.

I found the demonstration on Steganography fascinating and incorporated this topic into my class as well- which the students very much enjoyed.
To reiterate, the benefits of the WECS training and conference were multifaceted. First, between 50%– 60% of my course is comprised of concepts I learned at WECS.

I incorporated a sizable amount of material into my lectures, demonstrations and labs. I also obtained additional knowledge in Information Assurance from WECS, beyond which I taught in my own course.

Another invaluable part of my experience at WECS was the camaraderie I gained from colleagues of universities as well as colleges in Hawaii and Mexico. It is important to know the expectations of four year colleges with respect to knowledge of incoming students. I wanted to make sure that, in general, the material I hoped to cover in my course would be considered adequate introductory knowledge for a student eager to pursue Network Security at the university level. This topic was touched upon in my informal discussions with WECS participants from four year colleges and universities. The information gained and contacts obtained from four year institutions will be of even greater importance if Genesee Community College decides to embark on a degree granting program in Network Security/ Information Assurance.
As mentioned above, what helped me in developing my course was the opportunity to travel to a variety of different seminars and training sessions. Though I ended up with a large amount of information at the end of each training seminar, there were certain topics that came up over and over again. These topics, I deemed as especially relevant to network security and made sure to cover them in my own course. Starting out, my quandary was “How do I figure out what is considered most important and how do I gear the information received to the intellectual level of my students?” Going to different seminars assisted me in developing answers to that question. Through the many seminars, workshops and conferences attended, the administration at Genesee Community College has been supportive every step of the way. In fact I doubt this endeavor would have been possible without the sustained support of the college.
I realize that assessing the different areas of security covered is an ever changing process that may take on a somewhat different form each time I teach the course. What was important and relevant in the area of network security even five years ago is completely different than what it is today. I anticipate that some topics I consider important today might not be so in years to come. Continued training is even more essential in Network Security than in other areas of technology.

One concern of teaching a network security course is the potential problem of hacking outside of a lab environment. This semester, our department instituted a policy in which every student enrolled in a Computer Systems and Networking course be required to sign a code of conduct form. We believed it was not wise to signal out one course (i.e. the network security course) for mandating the signing of a code of conduct form. Having all students in CSN sign this form, helped to emphasize responsible computing for every student, at every level and for every CSN course. We had no issues this semester with abuse. I believe this was partly due to the fact that students were well aware of the ramifications of such actions.
Having completed the course less than two weeks ago, I have no official written feedback from students. Students did however, express enjoyment and informed me that they had gained a great amount of knowledge from having taken the course. In the coming weeks, I will start to assess the course and lessons learned and begin to incorporate changes and enhancements. I anticipate the course will be offered again in the Fall 2006.