The University Of Toledo

Existing Graduate Course Modification Form

* denotes required fields

Contact Person*: Richard Molyet Phone: 530-8143 (xxx - xxxx) Email: richard.molyet@utoledo.edu

Present

Supply all information asked for in this column.(Supply core, research intensive and transfer module info if applicable)

College*: College of Engineering ▼

Dept/Academic Unit*: Electrical Engineering and Computer Science ▼

Course Alpha/Numeric*: EECS - 5760

Course Title: Computer Security

Credit hours: Fixed: 3 or Variable: to

CrossListings: EECS4760

Insert

To add a course, type in course ID and click the Insert button.

To remove a course, select the course on left and click the Remove button.

Remove

Prerequisite(s)(if longer than 50 characters, please place it in Catalog Description):

EECS 2110 and EECS 3450

Corequisite(s)(if longer than 50 characters, please place it in Catalog Description):

Proposed

Fill in appropriate blanks only where entry differs from first column.

College: --Select a College-- ▼

Dept/Academic Unit: --Select a Department-- ▼

Course Alpha/Numeric: -

Course Title:

Credit Hours: Fixed: or Variable: to

CrossListings:

Insert

To add a course, type in course ID and click the Insert button.

To remove a course, select the course on left and click the Remove button.

Remove

Prerequisite(s)(if longer than 50 characters, please place it in Catalog Description):

EECS 2110 and EECS 3540

Corequisite(s)(if longer than 50 characters, please place it in Catalog Description):

Catalog Description (*only if changed*) 75 words max:	**Catalog Description** (*only if changed*) 75 words max:
Survey of computer security concepts: ethics and responsibility, OS vulnerabilities and intrusion detection, viruses and worms, defensive strategies including secret/public key cryptosystems, firewalls and decoys.	

Has course content changed? ○ Yes ● No

If course content is changed, give a brief topical outline of the revised course below(less than 200 words)

Proposed effective term*: 201740 (e.g. 201140 for 2011 Fall)

File Type	View File
Syllabus	View

List any course or courses to be deleted.

Effective Date:

Effective Date:

Comments/Notes:

To correct a typographical error in the course prerequisite list.

Rationale:

Approval:

Department Curriculum Authority:	Richard G. Molyet	Date	2017/03/23
Department Chairperson:	Mansoor Alam	Date	2017/03/23
College Curriculum Authority or Chair:	Efstratios Nikolaidis	Date	2017/03/31
College Dean:	Mohamed Samir Hefzy	Date	2017/04/17
Graduate Council:	Constance Schall, GC mtg 5/2/17	Date	2017/05/03
Dean of Graduate Studies:	Amanda C. Bryant-Friedrich	Date	2017/05/04
Office of the Provost :	marcia king-blandford	Date	2017/05/10

print

Administrative Use Only

Effective Date: 2016/08/22 (YYYY/MM/DD)

CIP Code:

Subsidy Taxonomy:

Program Code:

Instructional Level:

Registrar's Office Use Only

Processed in Banner on: 2017/05/17

Processed in Banner by:	Tasha Woodson
Banner Subject Code:	EECS
Banner Course Number:	5760
Banner Term Code:	201810
Banner Course Title:	Computer Security

The University of Toledo • 2801 W. Bancroft • Toledo, OH 43606-3390 • 1.800.586.5336

Computer Security

The University of Toledo College of Engineering
Department of Electrical Engineering and Computer Science (EECS)
Computer Science and Engineering (CSE) Program
EECS 4760:001 / 4760:805 / 5760:001 (CRN 43846 / 43848 / 43847)

Instructor:	Dr. Larry Thomas	**Term**:	Fall 2016
Email:	Larry.Thomas@UToledo.edu	**Class Location**:	RC301 (LCCC) / PH 2650 (UT)
Office Hours:	T/R 9:30 – 10:45, 12:30 – 2:30 W: TBD Others by Appointment	**Class Day/Time**:	T/R 8:00 – 9:15 AM
		Lab Location:	N/A
Office Location:	RC 130-I (LCCC)	**Lab Day/Time**:	N/A
Office Phone:	440.366.4341	**Credit Hours**:	3

COURSE/CATALOG DESCRIPTION

Survey of computer security concepts: ethics and responsibility, OS vulnerabilities and intrusion detection, viruses and worms, defensive strategies including secret/public key cryptosystems, firewalls and decoys.

COURSE OVERVIEW

With the possible exception of medicine, perhaps no other field advances at such a rapid pace as does Computer Security. Sensational headlines grace the news media seemingly daily, reporting the latest breach of some computer system, or that some group has obtained millions of credit card numbers from an online retailer. Not all security incidents are so spectacular or newsworthy; nevertheless, all security breaches can be potentially costly to someone. As computers continue to become increasingly ubiquitous, and as the world conducts more and more business via computer (from smartphone to mainframe), the potential implications of computer security multiply exponentially.

This course looks at how we are all affected by computer security, precisely what *constitutes* "security", and how we can maximize our level of security by being aware of the kinds of activities that can threaten that security, and by employing countermeasures to thwart such attempts at compromising our security.

STUDENT LEARNING OUTCOMES

Upon successful completion of this course, the student will…

… be able to identify the three pillars of Computer Security, and examine potential threats in terms of them

… understand how Authentication, Access Control, and Cryptography can be used to thwart potential threats

… understand the underlying principles behind mainstream cryptographic algorithms, specifically including (but not limited to) DES, AES, and RSA, and be able to code at least two of these algorithms

… understand how coding of their own applications can create potential vulnerabilities, and how to avoid them

… understand how security measures can be designed into Operating Systems to facilitate a secure environment

… understand why networked computers have their own particular vulnerabilities, and how to mitigate exposure

… understand how databases present distinct security challenges, and how they can be compromised

… be familiar with how non-technical issues like privacy, ethics, and legal aspects of security can affect computing

TEACHING STRATEGIES

This course consists of two 75-minute lectures per week. The lectures will be based on readings from the textbook (see below). Lecture notes will be provided as PowerPoint slides after the lecture. Students will study several aspects of computer security, including cryptography. Two cryptography-based programming assignments are planned. These are to be completed in C/C++ using Microsoft's Visual Studio (see below).

Additionally, an extra-credit, semester-long challenge will be presented after we have discussed steganography. The *first* student to correctly solve the challenge (which will require writing their own code) will receive a letter-grade bump at the end of the semester.

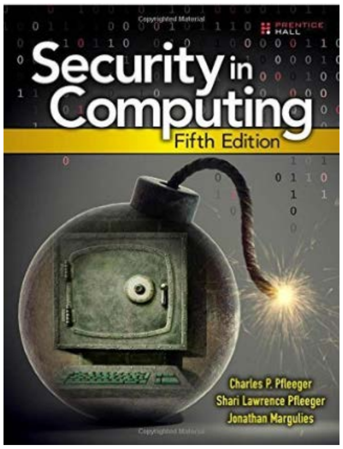
PREREQUISITES AND COREQUISITES

Prerequisites: EECS 2100 (Computer Architecture and Assembly Language, renumbered EECS 2110) and
EECS 2550 (Operating Systems and Systems Programming, renumbered EECS 3540)

REQUIRED TEXTS AND ANCILLARY MATERIALS

Required text:

Security in Computing, 5e., by Pfleeger, Pfleeger, and Margulies
Prentice-Hall, 2015, 944 pp.
ISBN-10/13: 0134085043 / 978-0134085043
According to www.BigWords.com (8/15/2015):
Retail: $120 / 90 (New / Used) – LCCC Bookstore
Rentals: $40 – 70 – LCCC Bookstore
E-Book (subscription): $45.00
Used: $60.75 and up (per BigWords.com)
New: $79.78 and up (per BigWords.com)



I will be teaching from the US edition of this text. "International" editions may be found with a lower price, and are typically substantially the same as the US version; however, be forewarned that the content may not be *entirely* the same. If you opt for an international edition, you are completely on your own for the differences between my text and yours. I don't have the international version of this text; therefore, I have no way of knowing precisely how (and if) they differ.

About used textbooks, rentals, and e-books (in general):

College texts have a (well-deserved) notorious reputation for being ridiculously expensive, driven partly by the fact that they're a small-volume commodity (so printing, warehousing, and distribution costs are spread out over a smaller number of copies than mass-market books), and partly by their relatively short life-span – new editions come out periodically, rendering the old editions largely obsolete, so publishers have to recoup their costs in a shorter period of time, from fewer copies, targeted at fewer readers.

I didn't write any of your textbooks, nor do I own stock in any of the publishing companies, so I don't make a dime from any book sale. I try to save you money where I can.

I'm not a fan of using an e-book as your exclusive resource (i.e., no physical copy). Yes, they're lighter, they certainly take up less space, and are often less expensive, but they're just not the same.

I recommend rentals only when you know you will have no use whatsoever for the book later on. In your major, however, consider most (if not all) of your texts as an investment in your career, and plan on keeping them. Used books are usually fine. Some books now come with scratch-off codes for supplemental resources, and those codes have a limited lifetime. The code you get in a used copy might not let you access the supplemental resources you need – beware!

Books purchased online can often be sold back (at a reduced rate) at the end of the semester. I don't recommend this for books in your major – now is the time to start building your CSE library. If you *are* interested in selling back a book, check the bookstore or on-line sites like http://www.BigWords.com for buy-back values.

TECHNOLOGY REQUIREMENTS

You will have at least two programming assignments that will require you to use Visual Studio. I suggest you consider putting your VS project folder on a flash drive to maximize portability between home and school. Visual Studio 2015 will be used. If you are using an older version, note that you can obtain the latest version free from Microsoft. The "Express" version is available to everyone free; the "Pro" version is available free to students who have registered with Microsoft's DreamSpark program, and the "Enterprise" version is available free to students registered at LCCC for Toledo classes through our DreamSpark Premium institutional account.

When your projects are complete, you will submit a 7-zip archive of them to Blackboard. If you do not have 7-zip, you can get it free from www.7-zip.org.

If you are registered for the course, and are unable to sign on to Blackboard, go to MyUT.UToledo.edu, click on "Account Maintenance", and follow the prompts to reset your password. This is the only mechanism that is sure to inform all of the other systems (including Blackboard) of the new password. If you have done so and are still unable to get into Blackboard, see me ASAP. I will be posting the lecture slides to Blackboard, so you will definitely need access.

Lecture slides will be in PowerPoint. If you do not have PowerPoint, you can get the free PowerPoint viewer from Microsoft: https://www.microsoft.com/en-us/download/details.aspx?id=13

About Flash Drives

Flash drives are an easy way for you to keep your development environment with you, and to have the portability of being able to work on your code both at home and on campus. The portability of flash drives also makes them easy to leave behind or misplace. In order to keep from losing your drive, you might want to consider putting it on your key ring (it's hard to leave your flash drive in the building when that also means leaving your car keys in the building). It is also recommended that you have a file in the root of the flash drive with your contact information, perhaps named something like "If found, please contact.txt". This may help get your drive directly back to you much faster. Drives found without clear contact information go in to the lost-and-found bin, where they sometimes sit for a LONG time.

Eventually, you *will* lose a flash drive. If it is your only copy of your work, then you have a huge problem. An easy way to not lose too much work from your flash drives is to get the program USB Flash Copy (available in a free version from http://www.USBFlashCopy.com). This program will recognize your flash drive when you insert it, and will automatically back up its contents to a directory on your hard drive. There are a number of settings you can customize (the hard drive location, files to exclude, how often to backup when the flash drive stays connected, etc.). The only difference between the free and $40 versions is that the free version pops up a status box in the corner of the screen while it's working; the paid version gives you "stealth mode", in which the status box is invisible. This is cheap insurance!

As the capacity of flash drives has risen, the price has fallen. If you're going to dedicate a flash drive to your Visual Studio projects, it doesn't need to be a large drive. If you have a huge drive, you'll be tempted to put other things on it, and that will just slow down USBFlashCopy. I recommend a drive of no more than 4 GB for your projects (2 GB is plenty, so you can probably repurpose an old one).

If you want the ultimate in backup support, you may want to also consider an on-line storage service. If you use DropBox, you can have USBFlashCopy back up your flash drive to the DropBox folder associated with your account, and then you can still access your files from anywhere, even if you happen to forget your flash drive (DropBox continuously keeps your cloud folder in sync with the DropBox folder on your hard drive). DropBox is ***free*** for up to 2 GB of storage (paid storage starts at $10/month ($100/year) for 100 GB of storage space).

I will have DropBox e-mail everyone an invitation to open an account. If you already have one, great. If you don't want one, that's fine too. If you DO open an account as a result of an invitation I have DropBox send to you, you'll get 2.5 GB (an extra 512 MB) free. DropBox is also supported on Android and iOS. Your use of DropBox is completely optional. If you have some other cloud storage mechanism already in place, feel free to use that.

UNIVERSITY POLICIES

Policy Statement on Non-Discrimination on the basis of Disability (ADA).

The University is an equal opportunity educational institution. Please read The University's Policy Statement on Nondiscrimination on the Basis of Disability Americans with Disability Act Compliance (http://www.utoledo.edu/policies/administration/diversity/pdfs/3364_50_03_Nondiscrimination_o.pdf)

Academic Accommodations

The University of Toledo and Lorain County Community College (LCCC) are committed to providing equal access to education for all students. If you have a documented disability (or you believe you have a disability) and would like information regarding academic accommodations/adjustments in this course, please contact the LCCC Disability Services office (http://www.lorainccc.edu/disability+services) and/or the University of Toledo's Student Disability Services Office (http://www.utoledo.edu/offices/student-disability-services/index.html).

ACADEMIC POLICIES

Students are required to comply with the University of Toledo's Academic policies (available on-line at https://www.utoledo.edu/policies/academic/undergraduate/index.html), and LCCC's Academic Policies (available at http://www.lorainccc.edu/NR/rdonlyres/10E67848-7A20-4457-8593-6004EE2137E6/0/pol2412.pdf)

COURSE EXPECTATIONS

You are expected to be present and on-time for all classes and tests. You alone are responsible for all material covered and announcements made in class. Makeup exams will only be given to those who have a *documented University excuse* and who have notified me of their absence *in advance* so that alternate arrangements may be made. There will be no make-ups for quizzes.

I will take attendance every lecture and every lab. Poor attendance will negatively impact your grade. If you are not present when the roll is called, but you arrive late, it is your responsibility to make sure that the instructor has marked you as “Late”, rather than “Absent”. Please do so after class.

My own interpretation of the Academic Integrity Policies:

Academic integrity is required of all students. Unless specifically directed, all work submitted must be your own individual work. Students may bounce ideas off one another, but you must write all of your own code, and do your own homework. You must be prepared to prove that your work is your own, by explaining it to me or with written evidence. Copying from a book, journal, or online source, even if you credit the source, is not considered original work and will be treated as cheating. Cases of academic dishonesty will be dealt with most severely, up to and including receiving an F for the semester, and a permanent entry on your transcript. Repeated instances can result in being expelled from the University. Seek help from your instructor before or after class, during office hours, or via e-mail most any time. They are here to help you with your problems in this course; don’t confuse your instructor’s role in your education with your peers’ role.

In programming-based courses, it is natural for students to find themselves working on projects in the lab while their classmates are working on the same project at the same time. Naturally, conversations regarding those assignments will take place. To the extent that those conversations consist of concepts, topics, and ideas, they are not only acceptable, they are encouraged – much of the learning in your courses will occur from one another outside of class. When these conversations drift into the realm of specific ways to solve a problem, the resulting work ceases to be individual effort, and becomes a group project, which is expressly forbidden.

As a trivial example, pretend that you have been assigned this algebraic problem: Given: $3(\sqrt{}x) + 4 = 19$, solve for x.

An acceptable conversation for this problem might include something along the lines of “you need to find some set of operations you can apply to both sides of the equation, such that you ultimately isolate x on one side of the equation; that will leave the result on the other side.” At this level, you are discussing approaches to solving this kind of problem in general (topics, concepts).

If, on the other hand, the conversation includes any or all of, “subtract 4 from both sides, divide both sides by 3, and then square both sides. That gives you x = 25”, then it has crossed over from “general concepts” into “specific solution”, and would be considered cheating.

If you leave something like this on the whiteboard in the lab…

$$\begin{aligned} 3(\sqrt{x}) + 4 &= 19 \\ 3(\sqrt{x}) &= 15 \\ \sqrt{x} &= 5 \\ x &= 25 \end{aligned}$$

…then it's the same as giving someone a copy of your homework, and that's cheating, too. Feel free to use the whiteboard to work through problems, but when you're finished, erase it. Don't copy what someone else left on the whiteboard. That's just like digging through the trash for a classmate's paper.

In a correct computer program, certain components of the program will be common to every student's submission. However, despite the use of coding conventions, programming is still a highly individualized practice, and there is not one single, correct way to write a program. Thus, individual programmers' style will come into play (much like a fingerprint), and two equally correct programs will typically look markedly different in any of several ways.

If you copy code from someone else, the amount of work you'll have to do in order to change it enough to make your copied work look like an individual effort (without breaking the code) is typically more than the amount of work it would take to just do it yourself from scratch.

Your goal in a programming assignment isn't just to turn in working code; it's to learn how to write and debug the code that implements a solution to the problem at hand. If you jump straight to "copy it from someone else just to turn it in", you've short-changed yourself, and have not prepared yourself for the next step in the curriculum. The problems this causes will snowball as you go forward.

When you get an assignment, start it early, and involve your instructor when you run into problems; don't wait to get started, and then count on your classmates to save you from your poor time management!

Unless you have explicit permission in advance, all work is due at the scheduled deadline. Submissions 0-24 hours late will be graded at 50% maximum credit; submissions 24-48 hours will be graded at 25% maximum credit; submissions more than 48 hours late will receive no credit.

GRADING

Grading will consist of exams (at least a mid-term and a final; there may or may not be more tests) programming assignments, and attendance / participation:

Grading formula:	Programming Assignments:	40%
(tentative)	Attendance (lecture, labs, and may include other events):	10%
	Tests, Quizzes, and Exams	50%

Grading of programming projects:

First and foremost, programs should work. Correctly. If the programming assignment details a feature set, rest assured that I will be testing your code against the requirements (as it works in the real world). Minor features (such as command-line argument validation) will be worth fewer points than core functionality (such as encryption code that can actually encrypt and decrypt a file). If your code doesn't work and/or doesn't compile, I can't award it many points.

Timing:

It is my intention to schedule exams for Thursdays, so that I can get them graded over the weekend, and we can review the exam and the correct answers during the next class period. You are strongly encouraged to follow along, and mark up your copy with the correct answers.

Labs and homework assignments are to be graded within two weeks of their submission. Grades will be posted to Blackboard, and I will send out a mass e-mail to the class when your grades are available.

Extra Credit:

Opportunities for extra credit at the end of the semester to salvage a poor grade will not be provided. You have fifteen weeks to earn credit. Waiting until you have under-performed to decide, after-the-fact, that you should have worked harder is a recipe for disaster, and I won't facilitate your participation in this approach. The lone exception to my extra credit policy will be the "Steganography Challenge", which is open to all.

Midterm Grading:

Midterm grades are reported to the University of Toledo Registrar. They are calculated based on the work submitted so far, attendance, and other information available to the instructor. These grades are a "best guess" at the grade you would likely receive at the end of the semester with similar performance. Midterm grades will be entered the week of October 4, and are used to assist students with determining where they stand academically in the course. Attendance is also recorded during Midterm grading to meet state and federal laws regarding financial aid disbursement. Please note, if you are not attending class it could impact your financial aid (scholarships, grants, loans or Federal Work Study).

If you decide you are not going to attend this class (or any other class you have registered for), you must formally withdraw (drop) from the course. You can do this by logging onto the myUT portal, clicking on the "Student" tab, and then under My Toolkit clicking on Register/Drop/Withdraw. The deadline for withdrawing is October 28, 2016. If you plan to withdraw (from *any* class; not just this one), you should do your instructors the courtesy of letting them know, rather than simply disappearing and letting them wonder what happened to you. For more information about add/drop dates please visit the Registrar's Office online at: http://www.utoledo.edu/offices/registrar/registration_dates.html.

Final Grading Scale:

Based upon your final average, the following grades will be assigned:

-	A (92 – 100%)	A- (90 – 92%)
B+ (87 – 89%)	B (82 – 86%)	B- (80 – 82%)
C+ (77 – 79%)	C (72 – 76%)	C- (70 – 72%)
D+ (67 – 69%)	D (62 – 66%)	-
-	F (below 60%)	-

For an interpretation of these grades, see https://www.utoledo.edu/offices/registrar/student_records/grades.html

Graduate Students:

Those taking this course for graduate credit (EECS 5760) will be held to a higher standard of work; specifically, you programming assignments will require that you implement features and algorithms above and beyond the requirements for the undergraduates (those taking the class as EECS 4760)

COMMUNICATION GUIDELINES

It is my intention to respond to e-mails within one business day. I will frequently be available via e-mail until late evening, though there are no guarantees as to how late I will check e-mail.

STUDENT SUPPORT SERVICES

The University of Toledo, and for those taking this course on the LCCC Campus, the Lorain County Community College, have numerous resources that can help you in both academic and non-academic areas. See your instructor and/or Adrienne Aguilar if you need assistance.

COURSE SCHEDULE

The textbook consists of thirteen chapters, whose lengths vary greatly, as shown below:

Chapter #	Chapter Title	First Page	# Pages
1	Introduction	1	36
2	Toolbox: Authentication, Access Control, and Cryptography	36	96
3	Programs and Programming	131	102
4	The Web – User Side	232	49
5	Operating Systems	280	62
6	Networks	341	161
7	Databases	501	51
8	Cloud Computing	551	36
9	Privacy	586	62
10	Management and Incidents	647	56
11	Legal Issues and Ethics	702	67
12	Details of Cryptography	768	46
13	Emerging Topics	813	39

We will average about a chapter per week, except for Chapters 2, 3, and 6, which will likely require a week-and-a-half (or two) each. Shorter chapters (such as 1, 8, 13) may be covered in a single class period.

I will pull information from another text to augment the material in Chapter 12, and we will spend some extra time on the details of cryptography.

Please note that there is a significant amount of material for you to read (and digest). Allow yourself adequate, uninterrupted time to do both. It is far, far easier to *keep* up than it is to *catch* up. If you find yourself getting behind in your reading, reprioritize and get caught up. I will not necessarily cover everything in lecture than you will find in the text. Make your own notes from the text (prior to class), and then augment them with my lecture notes.