General Information

Instructor

Matt Bishop
Office hours: Tu 9:00AM–10:00AM and Th 4:00PM–5:00PM Pacific Coast time, or by appointment
Office: 3059 Engineering Unit II *Email*: bishop@cs.ucdavis.edu
Phone: (530) 752-8060 *WWW*: http://seclab.cs.ucdavis.edu/~bishop
Note: Please put *ECS 253 – URGENT* in the subject of all email to help me see it quickly!

Lectures

TuTh 12:10PM–1:30PM in Room 1062, Banier Hall

Course Outline

Elements of cryptography and data security; system security, and network security. Both theory and applications will be covered, but theory will be emphasized.

Course Goals

Some goals we hope you achieve:
1. learn the importance of computer security;
2. understand how to use cryptography in support of security services
3. learn the basic theory and practise of secure systems;
4. understand the types of security services needed for network security; and
5. analyze or survey some aspect of computer security and cryptography in depth.

Text

We will be using draft chapters of a book in preparation (*Computer Security: Art and Science*). These will be handed out in class.

Computer Programs

The homework assignments, and your project, may require computer programs. Any computer programs written for this class must be well documented, cleanly written, and have a manual page or write-up describing how to use it, its input, and its output. Include sample runs. If you have C or C++ available, I would prefer you use one of those; if not, please check with me.

Course Web Page, Handouts, and Newsgroup

The web page *http://ecs253.ucdavis.edu* contains links to all course handouts (except for the published/copyrighted papers). If that is not available, go to my web page and follow the link in the "Quick Index" section.
Because we have some students without access to the UC Davis campus newsgroups, information about this class, homework assignments, office hours, and so forth, will be posted to the web page as well as to the *ucd.class.ecs253* newsgroup. Read this newsgroup (or web page) daily, especially near the time assignments are due. You are responsible for everything posted. This newsgroup is not for discussion about the class, for but information from the instructor to you.
If you want to post things about the class, please use the discussion newsgroup *ucd.class.ecs253.d.*, or send the instructor a mail message asking that something be posted. Discussing something in this group is perfectly fair! Postings from both newsgroups will be copied to the web page regularly.

Homework

There will be 5 homework assignments. The due date will be on each assignment. I will try to have your homework

graded as quickly as possible, usually within three class periods after I receive it.
Homeworks are due on the given date. If you have a reason for turning it in late, please discuss it with me. I'm quite liberal about due dates in a graduate class, but I do *not* want to have to grade lots of papers at the end of the term, and I cannot post answers until all the homework is in. If you turn homework in late without making arrangements first, I will take 10% off your score for every day it is late (including weekends, holidays, and any other classification you care to use!)
Please think your answers through before writing them down in final form; a request for a proof requires a proof, not a statement that "it's probably right, and here are 15,000 examples to show it;" a request for a discussion should be treated as an essay question, with a main theme and arguments for and against the answer. It is fair to present the factors that affect your answer; it is not acceptable to begin by giving one answer in the introduction and a different answer in the conclusion! (Yes, you'll lose points.) And, always show your work; if you simply write down a correct answer and do not show how you got that answer, you will not get any credit.
All homework must be submitted electronically, in text, Postscript, or PDF format. See the UCDisk handout for more information. Also, if you have a problem with UCDisk, send email as described on the UCDisk web page. Copy me on the mail, and email your homework. *Accepting homework through email is a pain, so I will not accept it unless you have sent a trouble report.*

Project

This class requires a term project requiring you to do outside reading, or apply what we've learned in class to a realistic situation, or extend your knowledge beyond what is done in class. The project is an integral part of the course, because it demonstrates you've learned enough to go beyond what we talked about in class. The section **Projects** describes the requirements in some detail and suggests possible projects, as well as the required intermediate reports.

Grading

50% Homework
50% Project
Note that there are no exams.

Recommended Reading

1. Edward Amoroso, *Fundamentals of Computer Security Technology*, Prentice-Hall ©1994
   Covers many topics but with little depth. This provides a very good overview of the subject, but you need to follow the references to appreciate much of what is said.
2. Edward Amoroso, *Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps, and Response*, Intrusion.net Books ©1999.
   An excellent introduction to one of the most exciting fields of computer security. If you're interested in this area, this book is a "must read."
3. Dorothy Denning, *Cryptography and Data Security*, Addison-Wesley ©1984
   Perhaps the best computer security text written so far; its only problem is being very out of date. Much of the cryptography is drawn from this book. If you can get a copy of it, I strongly encourage you to do so; it's a wonderful text.
4. Helen Fouché Gaines, *Cryptanalysis: a Study of Ciphers and their Solution*, Dover Publications, ©1956.
   A classic on cracking transposition and substitution ciphers, it does not cover more modern cryptography, but it shows the basics of cryptanalysis in a non-mathematical way.
5. Simpson Garfinkel and Gene Spafford, *Practical UNIX and Internet Security*, O'Reilly & Associates, ©1996.
   A marvelous book on UNIX security. Don't look for deep principles here; this book is a practicum.
6. Morrie Gasser, *Building a Secure Computer System*, Van Nostrand Reinhold, ©1985
   "The" book for practical and theoretical considerations in the design of a secure computer system. Not too rigorous, but quite comprehensive.
7. Katie Hafner and John Markoff, *Cyberpunk*, Simon & Schuster, ©1991.
   This book describes three of the better-known computer security incidents and the people behind them. It's not too technical, but a good study of hackers.
8. Lance J. Hoffman, *Rogue Programs: Viruses, Worms, and Trojan Horses*, Van Nostrand Reinhold, ©1990.

A collection of papers about malicious programs; the section on social and legal issues is very interesting,

9. David Kahn, *The Codebreakers*, Second Edition, Macmillan ©1996.
   Truly a classic, this book combines history with some basic cryptanalysis to show the evolution of codes and ciphers. This is the unabridged version, recently updated and re-released.
10. Alan Konheim, *Cryptography: A Primer,* John Wiley and Sons ©1981.
    Probably the best book yet on cryptanalysis; it does not have as much depth as Meyer and Matyas' book on some subjects (such as the DES), but it is much broader in scope. Beware of the notation, though: this can be a very hard book to understand!
11. Carl Meyer and Stephen Matyas, *Cryptography: A New Dimension in Computer Data Security*, John Wiley and Sons, ©1982.
    A very complete study of modern cryptography; the chapter on the DES is excellent.
12. National Research Council, *Computers at Risk: Safe Computing in the Information Age,* National Academy Press, ©1991.
    A study of how national policy should reflect problems, and advances, in computer security.
13. Donn Parker, *Crime by Computer,* Charles Scribner's Sons ©1976.
    Good discussion of what can happen if you ignore security considerations; it also considers ethics, something rarely seen but very badly needed.
14. Wayne Patterson, *Mathematical Cryptography for Computer Scientists and Mathematicians*, Rowman and Littlefield, ©1987.
    Highly mathematical, up-to-date treatment of many ciphers. Watch out for typographical errors and switches in notation, though!
15. Bruce Schneier, *Applied Cryptography,* Second Edition, John Wiley and Sons, ©1996.
    This book is a good but non-rigorous introduction to cryptography. The first edition had loads of errors, but (I am told) this version has eliminated most of them.
16. Abraham Sinkov, *Elementary Cryptanalysis: A Mathematical Approach*, The Mathematical Association of America, ©1966.
    A readable yet mathematical account of substitution and transposition ciphers.

Academic Integrity

Please see the Spring 2000 *Class Schedule and Room Directory* for a general discussion of this. In particular, for this course:

- All work submitted for credit must be your own. You may discuss your assignments with classmates, with instructors, or with readers in the course to get ideas or a critique of your ideas, but the ideas and words you submit must be your own. Unless explicitly stated otherwise in the assignment, collaboration is considered cheating and will be dealt with accordingly.
- For written homework, you must write up your own solutions and may neither read nor copy another student's solutions.
- For programs, you must create and type in your own code and document it yourself. Note that you are free to seek help while debugging a program once it is written.

A good analogy between appropriate discussion and inappropriate collaboration is the following: you and a fellow student work for competing software companies developing different products to meet a given specification. You and your competitor might choose to discuss product specifications and general techniques employed in your products, but you certainly would not discuss or exchange proprietary information revealing details of your products. Ask the instructor for clarification **beforehand** if the above rules are not clear.

Syllabus

#	Date	Topic, Readings, and Other Information
1.	Tuesday, April 4	Introduction to Computer Security
2.	Thursday, April 6	Foundations Part 1: Access Control Matrix, HRU
3.	Tuesday, April 11	Foundations Part II: Take-Grant, SPM
4.	Thursday, April 13	Security Policies
5.	Tuesday, April 18	Confidentiality Models
6.	Thursday, April 20	Integrity Models
7.	Tuesday, April 25	Other Models: Availability, ORCON, Role-Based, Non-Interference
8.	Thursday, April 27	Basic Cryptography: Ciphers, Protocols
9.	Tuesday, May 2	Applications of Cryptography: Key Management and Distribution
10.	Thursday, May 4	Access Control Mechanisms
11.	Tuesday, May 9	Information Flow
12.	Thursday, May 11	Security Kernels
–.	Tuesday, May 16	**no class** (2000 IEEE Symposium on Research in Security and Privacy)
13.	Thursday, May 18	Formal Methods for Assurance: Specification and Verification
–.	Tuesday, May 23	**no class** (National Colloquium on Information Systems Security Education)
14.	Thursday, May 25	Informal Methods for Assurance: Testing
15.	Tuesday, May 30	*Ad hoc* Methods for Assurance: Auditing, Intrusion Detection
16.	Thursday, June 1	Applied Methods for Assurance: Vulnerability Analysis and Secure Programming
17.	Tuesday, June 6	Network Systems Security
18.	Thursday, June 8	Distributed Systems Security *Reading*: *text*, chapter 28

We may schedule make-up classes for the two that I will miss. The exact date and time of the make-up classes depends upon the schedule of class members (all must agree to the dates and times!).

Projects

Why a Project?

This course covers a very large discipline, and – perhaps more so than many other areas of computer science – the discipline of computer security runs through many other areas. Because the class has a very limited amount of time, we will only touch the surface of many topics. The project gives you an opportunity to explore one of these topics, or some other area or application of computer security that interests you, in some depth.
The specific goal of the project is to produce a paper. The paper may document software (or hardware) work, so you may choose that kind of project. The paper must either be of publishable quality, or be publishable should some (small amount) of additional work be done.

Suggestions for How to Proceed

First, choose a topic. Good ways to find a topic are to think about an area of computer science you enjoy, and try to relate it to computer security (or vice versa); talk to some other graduate students and see if what they are doing suggests any ideas; think of ways security of the system you're working on could be made better; go to the library and browse for an interesting-looking paper; and so forth. The major computer security journals are *Computers & Security* and *Journal of Computer Security,* but articles appear in almost all journals; the major conferences are *Crypto* and *Eurocrypt* (for cryptography), *Symposium on Research in Security and Privacy, National Computer Security Conference,* and the *Annual Computer Security Applications Conference.* If you need more help or have questions, feel free to talk to me.

Some Suggestions for Project and Report Topics

The following are just to get you thinking. You will need to do much refinement for each!

- Analyze your favorite Internet or network protocol with respect to specific security requirements. Is it adequate, or should changes be made to enhance its ability to meet stated goals?
- Do a historical survey of computer viruses or worms. You will need to examine the differences of types of viruses (or worms) as well as giving a chronology.
- We have several copies of an attack kit called *rootkit*. Analyze its genealogy – which version came first, can you trace their evolution, and how, *etc.*
- UC Davis has an electronic mail security policy. Is it reasonable or realistic? What are the legal implications? Could you improve it from the point of view of system administration?
- Look at attack signatures and derive a little language to capture some class of them. Can you generalize your language to include as many attacks as possible? Focus on the temporal aspects.
- Add temporal logic to the Take-Grant Protection Model.
- The non-interference and non-deducibility results are related to multi-level security used to protect confidentiality. Can you either extend those results to the Biba integrity model, or set up a similar notion for integrity-based or availability-based models?
- How would you look for non-secure settings of environment variables in an executing program? Can you develop a wrapper that will check those values whenever a subprocess is spawned? (The motive here is that we may not have access to the source code, but can wrap the program so when it executes, the wrapper controls execution and can stop the wrapped program to check state.) You may need to hack a kernel to do this.
- Design and implement Karger's Trojan Horse checking scheme. Be sure you check *login*, *mail*, *etc.* because those are the programs attackers will instrument.
- Pick a class of vulnerabilities, analyze it, and design tools to check for those problems in program. Substantiate any claims of success by implementing a prototype and using it.

What Is Due When

All submissions are to be made through UCDisk.

Thursday, April 13	By this time you should have chosen your project. Turn in a 2–3 paragraph write-up of what you want to do, and why; list several sources (at least 3), and describe how you plan to go about completing the project. Your submission is to be in HTML format; a template is on the web page. I will post this to the web page *as soon as I get it*.
Tuesday, May 9	By this time your project should be well underway. Turn in a *detailed* outline or design document. Be specific about what you are doing, how, and what you expect (hope!) will be the result. Motivation is important; why should anyone other than you care about your result? Again, your submission is to be in HTML format; a template is on the web page. I will post this to the web page *as soon as I get it*.
Thursday, June 8	Your completed project is due.

All About UCDisk

This term, we will be experimenting with the University's shared file system, called *UCDisk*. All homework is to be submitted through this shared file system. This is a brief introduction to using it.

What Is UCDisk?

UCDisk is a large shared disk facility built on AFS (a descendent of the Andrew File System). It provides fine-grained access control, which will play a part in how you will use UCDisk. It also is accessible from a large number of places on campus, including from the *isun* systems and most Windows labs. If you have a home computer running Windows 95 or 98, and you connect over the Internet (either from another ISP or through the campus modem bank using PPP), you can mount the file system onto your home machine.

Unfortunately, you cannot yet connect from the CSIF. If this experiment works out well, we will try to make it available next term. But currently AFS software is available to the University only for SGI systems (although a native Linux client is being written).

Accessing UCDisk

To access UCDisk, you first need a campuswide login (called your "UCD login") and a password. The authentication protocol Kerberos is used to authenticate you, and your identity in turn controls what you can access. This means your password is *never* sent over the network (in the clear or enciphered). If you do not have a UCD login, *telnet* to the host mothra.ucdavis.edu, log in as *services* (no password needed) and choose N from the menu.

Your "UCDisk home directory" has your UCD login as its name. For example, my UCDisk home directory is called "bishop". DCAS has created UCDisk home directories for everyone in the class who has a UCD login. If you do not have a UCD login, please get one and email me when it is selected, and I'll ask DCAS to make you a directory.

Once you have a directory, you can access it through Windows (any of 95, 98, or NT will work[1]), the *isun*s, or the ECE UNIX computer systems. (Sorry, you can't do it through a Mac. The Mac doesn't handle the right versions of Kerberos.[2]) The web page http://ucdisk.ucdavis.edu describes how to do this from all systems; go to the "Getting Started" link. Briefly, from a UNIX system, if your UCD login is *ucdlogin*, you log into the system, and run a command to authenticate yourself to UCDisk. From a Windows system, log into UCDisk using the web interface and then perform the network mounting as described on the web page; your home directory will be represented by a network drive (the exact letter depends on your system setup).

UCDisk Web Page

This is an abbreviated version of the information available from http://ucdisk.ucdavis.edu. I recommend you check that web page out for more information.

1. The standard method doesn't seem to work for Windows 2000. We hope to get it working this term.
2. But stay tuned … we hope to get one sometime during the term!