CS 261: Computer Security

Overview

CS 261 is a graduate class on computer security offered in the Fall 2002 semester.

Instructor: David Wagner.
Time: 12:30--2:00pm, Tuesdays and Thursdays.
Location: 310 Soda.
Course Control Number: 26944.
Office Hours: 2-3pm after class (tentative), or by appointment if you can't make that time slot.
Prerequisites: CS 162 or equivalent. Familiarity with basic concepts in operating systems and networking.
Web page: `http://www.cs.berkeley.edu/~daw/teaching/cs261-f02/`

For announcements, see the class newsgroup, ucb.class.cs261.

Course description

CS261: Security in Computer Systems. Prerequisite: CS162. Graduate survey of modern topics in computer security, including: protection, access control, distributed access control, Unix security, applied cryptography, network security, firewalls, secure coding practices, safe languages, mobile code, and case studies from real-world systems. May also cover cryptographic protocols, privacy and anonymity, and/or other topics as time permits. Term paper or project required. Three hours of lecture per week. (3 units)

Course topics

An approximate list of course topics (subject to change; as time permits):

Basic concepts
: Trust, trusted computing base, trusted path, transitive trust. Reference monitors. Policy vs. mechanism. Assurance. Lessons from the Orange Book.

Access control
: Authorization, policy, access matrix. Subjects and objects. ACLs, capabilities. Rings, lattices. Revocation. Groups. The role of crypto. Distributed access control. Mandatory vs. discretionary access control, compartmentalization, covert channels.

Protection
: Traditional OS centralized protection: address spaces, uids, resource management. The Unix security model: file permissions, the super-user, setuid programs, system calls, password security. How networks change the problem space.

Secure coding
: Design principles: code structure, least privilege, small security kernels, small interfaces. Tools: language support, type-safe languages, static checking. Common vulnerabilities: buffer overruns, setuid programs, the confused deputy, race conditions, improper canonicalization.

Cryptography
: Symmetric key, public key, certificates. Choosing an algorithm. Protocols. Integrity, authenticity confidentiality, availability. Non-repudiation.

Network security
: TCP/IP. Attacks on network protocols: address spoofing, hijacking, source routing, SYN floods, smurfing, etc. DNS attacks, routing vulnerabilities. Attacks on network daemons. The Internet Worm. TCP wrappers. Intrusion detection.

Firewalls
: Philosophy, benefits. Styles: packet filter, application proxying, stateful inspection. Performance, scalability. Fail-safety, assurance. Techniques. Do's and don'ts.

Confining untrusted code
: Motivation: the mobile code problem, implementing least privilege. Mechanisms: signed code, interpreted code, software fault isolation, proof-carrying code, virtualization, extensible reference monitors. Practical experience: ActiveX, Java, Javascript.

Case studies
: Kerberos. PGP and the web of trust. SSL and centralized certification authorities. SSH. IPSEC. Cellphones. Therac-25. Practical issues: risk management, key management, smartcards, copy protection systems, social engineering.

Extra topics
: Privacy: Anonymity and traffic analysis; remailers and rewebbers; practical experience. Cryptographic protocols: protocol failures, design principles; logics of authentication; Formal methods. Others as time permits and according to student interest.

Project presentations

Grading

Class project: 45%
Problem sets: 40%
Paper summaries and class discussion: 15%

There will be no final or midterm.

Projects

There will be a term project. You will do independent research in small groups (e.g., teams of 2--3). Projects may cover any topic of interest in systems security, interpreted broadly (it need not be a topic discussed in class); ties with current research are encouraged. A conference-style report and a project presentation on your results will be due at the end of the semester.

You are encouraged to start thinking of topics of interest early. Be ambitious! I expect that the best papers will probably lead to publication (with some extra work).

More details will be made available here in the future.

Problem sets

There will be approximately three to five homework assignments throughout the semester, to appear on the course webpage as they are assigned.

Reminder: Turn in your homeworks on paper at the beginning of class on the appropriate day. This deadline will be enforced strictly. Late homeworks will not be accepted.

You may discuss the questions on the homeworks with others, but the writeup you turn in must be your own. You may use any source you like (including other papers or textbooks), but if you use any source not discussed in class, you must cite it.

Readings

There is no required textbook. All reading will be from papers. Whenever possible, handouts and papers will be placed online on the web page; papers not available online will be handed out in class. A schedule of assigned readings is available online.

You will be required to write a *very brief* summary of *each* paper you read, to be handed in at the beginning of the class when the reading is due. Your paper summary need only list the one or two most significant new insight(s) you took away from the paper and its one or two most significant flaw(s); incomplete sentences are fine.

Returned paper summaries are available outside 765 Soda.

Warning

From time to time, we may discuss vulnerabilities in widely-deployed computer systems. This is *not* intended as an invitation to go exploit those vulnerabilities. It is important that we be able to discuss real-world experience candidly; students are expected to behave responsibly.

Berkeley's policy (and my policy) on this should be clear: you may not break into machines that are not your own; you may not attempt to attack or subvert system security. Breaking into other people's systems is inappropriate, and the existence of a security hole is no excuse.

David Wagner, daw@cs.berkeley.edu, http://www.cs.berkeley.edu/~daw/