A. JAMES CLARK
SCHOOL OF ENGINEERING

Course: ENPM 808B – Secure Operating Systems
Semester: Spring 2014
Day(s): Tu
Time: 4:00-6:40 PM (DETS)
Location: JMP 2222
Instructor: Atif Memon
Phone:
Email: atif@cs.umd.edu

Course Description

Prerequisite: ENPM 808 Programming in C for Cybersecurity Applications, CMSC 106 Introduction to C Programming, or permission of the instructor.

Catalog Description: Operating systems are the basic building block on which programmers build applications and on which security-minded professionals rely, whether they are monitoring activity on a computer, testing applications for security, or determining how malicious code affected their network. This course covers advanced topics in operating systems including process management and communication, remote procedure calls, memory management (including shared memory and virtual memory), checkpointing and recovery, file system, I/O subsystem and device management, distributed file systems and security. The course consists of reading and discussing research papers and includes a course project. Please note: This course assumes knowledge of C programming and a previous operating systems class or knowledge in various issues such as process management, process synchronization, the critical section problem, CPU scheduling, memory management, secondary storage management.

Detailed Description: An operating system comprises of a core kernel surrounded/supported by modules that provide basic operating services (file system management, memory management, etc.). Some parts of the kernel and modules are relevant to the security of the overall operating system. For example, the memory space allocated to a specific user may need to be protected from other (possibly malicious) users – the memory management module provides this security. Similarly, persistent storage space of a user may need privileged access – the file system provides this facility. Closely related to security policies is understanding the behavior of attackers and malicious users. For example, one common attacker behavior is to predict target memory addresses of programs. Because of this behavior, one security mechanism typically implemented in the memory management module randomizes address spaces, thereby hindering such types of attacks by making it more difficult for an attacker to predict target addresses. Application developers need to understand these behaviors so that they can assess the security risks of running their application on a particular operating system. Understanding and using secure operating systems thus requires the study and understanding of 4 separate conceptual threads: (1) operating system fundamentals, (2) understanding security features of kernel and modules, (3) attacker behavior and attack patterns, and (4) practical security features provided by today's operating systems.

While it is important to understand the theoretical foundations of secure operating systems, it is equally important to study how the theory is realized in a practical, modern, and widely used operating system. Discussion of each security feature studied in this course is therefore, tightly coupled with a discussion of how the feature is implemented in the Linux and Android operating systems. For example, the lecture on memory protection includes a discussion of Android and Linux memory management security enhancements that make common security issues harder to exploit; extensions to OpenBSD *dlmalloc* to

prevent double *free()* vulnerabilities and to prevent chunk consolidation attacks (to exploit heap corruption) are discussed in the context of Android 1.5.

This course presents at least three perspectives relevant to secure operating systems: (1) operating system designer, (2) programmer, who develops code that run on the operating system base, and (3) end-user, who simply uses the operating system and applications. The operating system designer creates and implements the security model that the operating system provides; the programmer uses the security model to develop software applications that rely on the implementation to provide security; the end-user expects the operating system and applications to work together seamlessly, providing a reasonable level of security.

After a short "introduction to OS" thread, the course dives into content that is interwoven with 4 threads that cover theory as well as practice: (1) kernel and core modules of an operating system, (2) security aspects of the kernel and relevant core modules, (3) how attackers think, and (4) two practical case studies of Unix/Linux and the Android mobile platform which discuss the mechanisms these OS'es provide, and how programmers may leverage these mechanisms for secure software development and analysis.

Required/Recommended Textbooks

Recommended (but not required) books

1. *Operating Systems: Principles and Practice*, by Thomas Anderson and Michael Dahlin.
2. *Operating System Security (Synthesis Lectures on Information Security, Privacy, and Trust)*, by Trent Jaeger.
3. *Fundamentals of Secure Computer Systems*, by Brett Tjaden.

Course Outline

Grading: The grade of the course will be determined as follows: 50% in-class tests (we will have 2 in-class tests; 25% each), 25% quizzes (we will have 12 quizzes of which 8 best scores will be considered), 25% final exam.

Date	Topic	Purpose	Readings/Comments
Jan. 28	Course Overview. What is a secure OS?	To introduce the course and basic terminology. By the end of this lecture, students will understand the need for a secure operating system and be familiar with the terms commonly used in the secure operating systems literature.	Various resources, textbooks, and Internet.

Feb. 4	Secure against what? Nature of threats/attacks	To introduce common classes of threats/attacks and instances of real attacks. By the end of this lecture, students will understand the nature of real attacks so that they can appreciate the mechanisms used to defend against them.	*Quiz 1 (in Class)* Is Apple's iMac Leopard Operating System Secure under ARP-Based Flooding Attacks? By Surisetty, S.; Kumar, S., Internet Monitoring and Protection (ICIMP), 2010 Fifth International Conference on , vol., no., pp.60,64, 9-15 May 2010. Various lists of attacks from the Internet.
Feb. 11	Parts of an OS	The significant modules that comprise an OS; the separation between spaces; how spaces interact in an OS. By the end of this lecture, students will understand the parts of an OS, the security requirements of each part, and how a vulnerability in one part may have an impact on another part.	*Quiz 2 (in Class)* *kGuard: lightweight kernel protection against return-to-user attacks*. By Vasileios P. Kemerlis, Georgios Portokalidis, and Angelos D. Keromytis. In Proceedings of the 21st USENIX conference on Security symposium (Security'12). Less is More -- A Secure Microkernel-Based Operating System, By Lackorzynski, A.; Warg, A., SysSec Workshop (SysSec), 2011 First , vol., no., pp.103-106, July 2011.
Feb, 18	Processes & Threads	The fundamental element of execution in an OS is the process and thread. In this lecture, students will learn about how an OS manages processes and threads.	*Quiz 3 (in Class)* Operating system textbook.

Feb. 25	Secure handling of Processes & Threads, Concurrency	Security in an OS starts with isolation of execution artifacts such as processes and threads. Students will learn about ensuring that processes and threads remain secure.	*Quiz 4 (in Class)* *ASIST: architectural support for instruction set randomization.* By Antonis Papadogiannakis, Laertis Loutsis, Vassilis Papaefstathiou, and Sotiris Ioannidis. In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security (CCS '13). ACM, New York, NY, USA, 981-992. Even more patterns for secure operating systems. By Eduardo B. Fernandez, Tami Sorgente, and Maria M. Larrondo-Petrie. In Proceedings of the 2006 conference on Pattern languages of programs (PLoP '06). ACM, New York, NY, USA, Article 10, 9 pages. Additional material on concurrency, locks, mutexes, etc., those ensure security by separation of access to critical sections of program data.
Mar. 4	Memory management	Many security vulnerabilities are related to exploitation of memory management in the OS. By the end of this lecture, students will understand the core of memory management and why it is important to have effective management practices for memory in an OS.	**First InClass Test** Operating system textbook.
Mar. 11	Secure memory management	Students will learn about practical memory management approaches that are designed for security.	*Quiz 5 (in Class)* *Enhanced operating system security through efficient and fine-grained address space randomization.* By Cristiano Giuffrida, Anton Kuijsten, and Andrew S. Tanenbaum. 2012. In Proceedings of the 21st USENIX conference on Security symposium (Security'12).

Mar. 25	File systems	The file system is responsible for, among other things, persistent data storage. By the end of this lecture, students will understand the requirements for a secure file management system in an OS.	*Quiz 6 (in Class)* Operating system textbook.
Apr. 1	Secure file systems	Students will learn about practical new approaches to design secure strategies for file systems.	*Quiz 7 (in Class)* Application-level isolation and recovery with solitude. By Shvetank Jain, Fareha Shafique, Vladan Djeric, and Ashvin Goel. 2008. SIGOPS Oper. Syst. Rev. 42, 4 (April 2008), 95-107. Enhancing file data security in Linux operating system by integrating secure file system, Pal, R.K.; Sengupta, I., IEEE Symposium on Computational Intelligence in Cyber Security, 2009. CICS '09, vol., no., pp.45-52, March 30 2009-April 2 2009.
Apr. 8	Secure communication and messaging	By the end of this lecture, students will understand the need for securing the communication/messaging layer; and they will study a few approaches for securing messaging.	*Quiz 8 (in Class)* Permission re-delegation: attacks and defenses. By Adrienne Porter Felt, Helen J. Wang, Alexander Moshchuk, Steven Hanna, and Erika Chin. In Proceedings of the 20th USENIX conference on Security (SEC'11). Simplifying security policy descriptions for internet servers in secure operating systems. By Toshihiro Yokoyama, Miyuki Hanaoka, Makoto Shimamura, and Kenji Kono. 2009. In Proceedings of the 2009 ACM symposium on Applied Computing (SAC '09). ACM, New York, NY, USA, 326-333.

Apr. 15	Untrusted OS	Until now, the course would have covered approaches to improve operating systems security. By the end of this lecture, students will learn to handle situations in which the OS is compromised. In such a case, they will learn how applications must take care of security themselves.	**Second InClass Test** INVISIOS: A Lightweight, Minimally Intrusive Secure Execution Environment. By Divya Arora, Najwa Aaraj, Anand Raghunathan, and Niraj K. Jha. ACM Trans. Embed. Comput. Syst. 11, 3, Article 60 (September 2012), 20 pages. InkTag: secure applications on an untrusted operating system. Owen S. Hofmann, Sangman Kim, Alan M. Dunn, Michael Z. Lee, and Emmett Witchel. SIGPLAN Not. 48, 4 (March 2013), 265-278.
Apr. 22	Security perspective: end-user	By the end of this lecture, students will understand what security means from an end-user perspective. They will study several issues that surround end-user security, including passwords, access control, etc.	*Quiz 9 (in Class)* Mandatory access control with a multi-level reference monitor: PIGA-cluster. By Mathieu Blanc, Damien Gros, Jérémy Briffaut, and Christian Toinard. In Proceedings of the first workshop on Changing landscapes in HPC security (CLHS '13). ACM, New York, NY, USA, 1-8. Aspects of security that the end user sees. Various Internet resources.

Apr. 29	Hardware/Architecture support for OS security	Various new architectural features are being provided at the hardware level to design more secure operating systems. By the end of this lecture, students will understand these features and how they are used. They will understand the benefits of using hardware-based security versus software-based security.	*Quiz 10 (in Class)* SHARK: Architectural support for autonomic protection against stealth by rootkit exploits. By Vikas R. Vasisht and Hsien-Hsin S. Lee. 2008. In Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture (MICRO 41). IEEE Computer Society, Washington, DC, USA, 106-116. SecureME: a hardware-software approach to full system security. By Siddhartha Chhabra, Brian Rogers, Yan Solihin, and Milos Prvulovic. 2011. In Proceedings of the international conference on Supercomputing (ICS '11). ACM, New York, NY, USA, 108-119. Secure virtual architecture: a safe execution environment for commodity operating systems. John Criswell, Andrew Lenharth, Dinakar Dhurjati, and Vikram Adve. 2007. SIGOPS Oper. Syst. Rev. 41, 6 (October 2007), 351-366.
May 6	Leveraging Android security for programs	By the end of this lecture, students will understand the Android OS security features upon which programmers rely when writing their apps.	*Quiz 11 (in Class)* From Android developers site; Android Security
May 13	Putting it all together – a case study app	By the end of this lecture, students will understand a full example Android app that uses Android's security features.	*Quiz 12 (in Class)*