The University of Texas at El Paso
Computer Science Department
Syllabus

MD Armanuzzaman

Course Information

CS 4390/5390: **System Security - Attack and Defense for Binaries**
Term: Spring 2026
Delivery Method: In-person
Meeting Days and Time: *TR, 3:00 PM - 4:20 PM*
Location: *EDUC 112*

Instruction Information

MD Armanuzzaman, Assistant Professor
Email: `marmanuzzaman@utep.edu`
Office Location: CCSB 3.1008
Office Hours: Teams Link

- In-person, or virtual on Teams: *TR, 1.30 PM to 3.00 PM*
- Or by appointment

Course Description

This course is designed to provide students with good understanding of the theories, principles, techniques and tools used for software and system **hacking** and **securing**. Students will study, in-depth, binary reverse engineering, vulnerability classes, vulnerability analysis, exploit and shell-code development, defensive solutions, etc. to understand how to **crack** and **protect native software**. In particular, this class covers **offensive** techniques including stack-based buffer over-flow, heap security, format string vulnerability, return-oriented programming, etc. This class also covers defensive techniques including canaries, shadow stack, address space layout randomization, control-flow integrity, seccomp, etc. A key part of studying security is putting skills to the test in practice. In this class the progress of students are evaluated by **hacking binary challenges**.

Course Schedule

The course schedule may be found on the course website: CS 4390/5390 Schedule. The schedule is subject to change depending on needs of the class.

Topics

1. Background knowledge
   (a) Principles of compiler, linker and loader
   (b) x86 and x86-64 architectures
   (c) x86 and x86-64 instruction sets
   (d) Embedded and IoT architectures, e.g., ARM Cortex-A and Cortex-M
   (e) x86 and x86-64 assembly and reversing engineering with Intel syntax
   (f) Linux file permissions
   (g) Set-UID programs
   (h) ELF file format, .plt, .got, lazy binding
   (i) Memory map of a Linux process (x86 and x86-64)
   (j) System calls
   (k) Environment variables
   (l) Linux shell commands and tricks, piping
   (m) Tools: `gcc`, `gdb`, `pwndbg`, `objdump`, `ltrace`, `strace`, `env`, `readelf`, `/proc/PID/maps`, `pmap`, `tmux`, `IDA Pro`, `Ghidra`, `binary ninja`, `tee`
2. Stack-based buffer overflow
   (a) C local and global variables
   (b) C calling conventions (x86 and x86-64)
   (c) How stack works
   (d) Overflow local variables
   (e) Overflow return addresses on stack
   (f) Injecting in local buffers, in environment variables, in program arguments
   (g) Frame pointer attack (overwrite saved EBP/RBP)
   (h) Return-to-libc attacks
   (i) Tools: `pwntools`
3. Defenses against stack-based buffer overflow
   (a) Base and bound check
   (b) Data Execution Prevention (DEP)
   (c) Stack Canaries: GCC implementation for x86 and x86-64
   (d) Shadow stack: (1) traditional; (2) parallel
   (e) SafeStack
   (f) Position Independent Executable (PIE)
   (g) Address space layout randomization (ASLR) and how to bypass it
   (h) FORTIFY_SOURCE
   (i) Seccomp (syscall firewall)

(j) Tools: `ldd`

4. Developing Shellcode
   (a) Writing, compiling, and testing x86 and x86-64 assembly code
   (b) System calls on Intel (x86 and x86-64)
   (c) Constraints on shellcoding: (1) zero-free shellcode; (2) ASCII-only shellcode; (3) Non-printable, non-alphanumeric shellcode
   (d) Research development: English shellcode, DNA shellcode
5. Format string vulnerability
   (a) C function with variable arguments
   (b) Difference with C++ function overloading
   (c) Format string, e.g., %n
   (d) Information leakage with format string vulnerability (memory read)
   (e) Format string vulnerability for memory write, e.g., .got, .fini
   (f) RELRO
   (g) Control-flow bending for Turing-complete printf-oriented programming
6. Heap security and exploitation
   (a) What is heap? Dynamic allocator and its interfaces, e.g., `malloc()`, `free()`
   (b) cache, chunks and metadata
   (c) Use after free (UAF) vulnerabilities
   (d) Double free vulnerabilities
   (e) Metadata corruption
   (f) Safe-linking
7. Integer overflow vulnerability
   (a) Integer overflow vulnerability
8. Return-oriented programming
   (a) ROP
   (b) Blind ROP
   (c) Jump-oriented programming
   (d) Control-Flow Integrity (CFI)
   (e) Hardware-supported CFI
   (f) Tools: `ROPgadget, pwntools`
9. Data-oriented attacks
   (a) Direct data manipulation
   (b) Data-oriented programming

(c) Data-Flow Integrity (DFI)

10. Race conditions

    (a) Race in file systems

    (b) Process and thread

    (c) Race in memory

Grading Policy

Students will be evaluated on their performance on the **homework and CTFs**. Attendance check will be performed in each class. Table 1 shows the grade breakdown.

Area	**No. Items**	**Points per Item**	**Points for Area**
Homework (CTFs)	9	70	630
Exams (CTFs)	2		360
Midterm Exam (CTFs)	1	160	
Final Exam (CTFs)	1	200	
Attendance	20	1	20
Anonymous Course Evaluation Bonus	1	20	20
Total			**1030**

Table 1: Grades Breakdown

Grading Chart

Table 2 presents the grading chart.

CS4390 (Undergraduate)		**CS5390** (Graduate)	
Points	**Grade**	**Points**	**Grade**
850 -	A	900 -	A
750 - 849	B	800 - 899	B
650 - 749	C	700 - 799	C
550 - 649	D	600 - 699	D
0 - 549	F	0 - 599	F

Table 2: Final Letter Grades

Prerequisite

Students are expected to have a background in the C programming language. Solid background from the following classes helps significantly: (1) Computer Organization; (2) Introduction to Computer Security; and (3) Operating Systems. The instructor strives to keep this class self-contained.

Textbooks

There is no required textbooks for this course. The instructor will send out book chapters and other reading materials throughout the semester. Some of the teaching materials of this course are based on the following books:

- Randal Bryant, David O'Hallaron. **Computer Systems: A Programmer's Perspective**, 3rd Edition.
- Dennis Andriesse. **Practical Binary Analysis: Build Your Own Linux Tools for Binary Instrumentation, Analysis, and Disassembly**.
- Per Larsen, Ahmad-Reza Sadeghi. T**he Continuing Arms Race: Code-Reuse Attacks and Defenses**.

Papers

Throughout the semester, we will discuss the following papers:

- Laszlo Szekeres, Mathias Payer, Tao Wei, Dawn Song. **SoK: Eternal War in Memory**. IEEE Security and Privacy 2013.
- Yan Shoshitaishvili, Ruoyu Wang, Christopher Salls, Nick Stephens, Mario Polino, Andrew Dutcher, John Grosen, Siji Feng, Christophe Hauser, Christopher Kruegel, Giovanni Vigna. **SoK: (State of) The Art of War: Offensive Techniques in Binary Analysis**. IEEE Security and Privacy 2016.
- Hovav Shacham. **The Geometry of Innocent Flesh on the Bone: Return-into-libc without Function Calls (on the x86)**. ACM CCS 2007.
- Tyler Bletsch, Xuxian Jiang, Vince W. Freeh, Zhenkai Liang. **Jump-Oriented Programming: A New Class of Code-Reuse Attack**. ACM AsiaCCS 2011.
- Andrea Bittau, Adam Belay, Ali Mashtizadeh, David Mazieres, Dan Boneh. **Hacking Blind**. IEEE Security and Privacy 2014.
- N Carlini, A Barresi, M Payer, D Wagner, TR Gross. **Control-Flow bending: On the effectiveness of Control-Flow integrity**. USENIX Security Symposium, 2015

Course Drop Policy

According to UTEP Catalog, "At the discretion of the instructor, a student can be dropped from a course because of excessive absences or lack of effort. A grade of "W" will be assigned before the course drop deadline and a grade of "F" after the course drop deadline." See Policies and Regulations in the UTEP Undergraduate Catalog for a list of excused absences. Therefore, if I find that, due to excessive absences or nonperformance in the course, you are at risk of failing, I will drop you from the course. I will provide 24 hours advance notice via email.

Incomplete and Grade Policy

The University is committed to providing reasonable accommodations and auxiliary services to students, staff, faculty, job applicants, applicants for admissions, and other beneficiaries of University programs, services and activities with documented disabilities in order to provide them with equal opportunities to participate in programs, services, and activities in compliance with sections 503 and 504 of the Rehabilitation Act of 1973, as amended, and the Americans with Disabilities Act (ADA) of 1990 and the Americans with Disabilities Act Amendments Act (ADAAA) of 2008. Reasonable accommodations will be made unless it is determined that doing so would cause undue hardship on the University. Students requesting an accommodation based on a disability must register with the UTEP Center for Accommodations and Support Services (CASS). Contact the Center for Accommodations and Support Services at *915-747-5148*, email them at cass@utep.edu, or apply for accommodations online via the CASS portal.

Scholastic Integrity

Academic dishonesty is prohibited and is considered a violation of the UTEP Handbook of Operating Procedures. It includes, but is not limited to, cheating, plagiarism, and collusion. Cheating may involve copying from or providing information to another student, possessing unauthorized materials during a test, or falsifying research data on laboratory reports. Plagiarism occurs when someone intentionally or knowingly represents the words or ideas of another as ones' own. Collusion involves collaborating with another person to commit any academically dishonest act. Any act of academic dishonesty attempted by a UTEP student is unacceptable and will not be tolerated. All suspected violations of academic integrity at The University of Texas at El Paso must be reported to the **Office of Student Conduct and Conflict Resolution (OSCCR)** for possible disciplinary action. To learn more, please visit **HOOP: Student Conduct and Discipline**.

Course-specific Scholastic Integrity Policy

Students are allowed to discuss homework assignments. However, students are **NOT** allowed to share code, exploits, write-ups, and homework with each other. Plagiarism or any form of cheating in homework or exams (CTFs) is subject to serious academic penalty. All integrity violations will be reported to the UTEP Office of Student Conduct and Conflict Resolution. There is a zero tolerance policy in this class.

- A minor integrity violation of a specific homework assignment (i.e., plagiarism on one question) may result in a **0** on that assignment.
- More serious integrity violation may result in downgrade of the final grade and even an **F** on the final grade, e.g., falsification.
- The college and university have policies to upgrade the penalty, which is independent of the course policy.

Syllabus Update

Information in the syllabus may be subject to change with reasonable advance notice.