This assignment challenges y'all to explore and demonstrate controlled exploitation techniques in a legal, ethical, and structured environment. Each task requires technical execution, documentation, and a risk assessment of the exploit or technique used. Pick 2 of the 7 below.

Option 1: Buffer Overflow Attack
- Develop or modify a vulnerable program that can be exploited via buffer overflow.
- Demonstrate an exploit that overwrites memory and causes a crash or unintended behavior.
- Analyze how ASLR and stack protections affect the exploit.
- Deliverables: Source code, memory analysis screenshots, explanation of behavior.

Option 2: Bypassing DEP & ASLR
- Identify and demonstrate a method to bypass DEP or ASLR on a provided vulnerable binary.
- Use ROP, return-to-libc, or other known techniques.
- Explain how modern defenses impact exploit development.
- Deliverables: Step-by-step process, screenshots, proof of concept.

Option 3: Controlled Web Exploitation
- Select a web vulnerability (SQL Injection, XSS, or CSRF) and demonstrate exploitation in a lab environment.
- Validate findings by manually crafting payloads and analyzing responses.
- Provide recommendations for mitigation.
- Deliverables: Attack methodology, request/response logs, proof of exploitation.

Option 4: Post-Exploitation & Privilege Escalation
- Perform post-exploitation techniques after gaining access to a vulnerable system.
- Techniques may include privilege escalation, credential dumping, or lateral movement.
- Explain how real-world attackers use these techniques after initial compromise.
- Deliverables: Walkthrough of the escalation method, captured credentials or access verification.

Option 5: Network Device Exploitation
- Identify a misconfigured or vulnerable network service (e.g., SMB, SNMP, Telnet).
- Use scanning tools like Nmap, CrackMapExec, or Metasploit to analyze weaknesses.
- Exploit the vulnerability while maintaining system integrity.
- Deliverables: Network scan results, exploit demonstration, remediation recommendations.

Option 6: Reverse Engineering a Binary
- Analyze a provided or self-selected binary using tools like Ghidra or Lady IDA Free.
- Identify security flaws, hardcoded credentials, or improper memory protections.
- Attempt to modify execution flow (e.g., bypassing authentication).
- Deliverables: Decompilation analysis, vulnerability report, optional modified binary.

Option 7: Wireless & IoT Exploitation
- Investigate and analyze the security of a wireless or IoT device.
- Methods may include Wi-Fi cracking, Bluetooth enumeration, or IoT firmware analysis.
- Test for weak authentication, misconfigurations, or open services.
- Deliverables: Findings report, screenshots, analysis of attack feasibility.

Additional Graduate Student Requirement

In addition to completing two tasks, graduate students must:
- Write a 3–5 page research analysis on modern security mitigations (e.g., Stack Canaries, RELRO, Control Flow Integrity) and how they impact exploitability.
- Evaluate a real-world case study where a mitigation was bypassed.

Submission Requirements
- A professional-style penetration test report covering both selected tasks.
- Supporting screenshots, scripts, and proof-of-concept code (if applicable).
- Please try to limit the number of submitted documents to the minimal required.
- Graduate students: Include a separate research analysis document.

Grading Rubric

Category	Points Per Task	Total
Technical Execution	50	100
Documentation & Report Quality	30	60
Risk Assessment & Mitigation	20	40
Total Per Task	100	200