CSC415: Final

Exam Policies

- **You are allowed to bring in one piece of paper/two sides or two pieces of paper/one side with notes for the exam. Your study guide can be typed or handwritten but must be unique from every other student. You will turn in this sheet when you complete the exam.**
- **The exam is cumulative but emphasizes the material since the last midterm exam. I have made the new sections blue below.**
- Bring your computer - the exam will be administered partially through Gradescope, and partially on paper. You can only stay in Gradescope during the exam.
- You will have 2 ½ hours once you begin the exam to complete the exam.
- Collaboration will anyone else or using unauthorized resources is forbidden on the exam. You must work alone. If you violate the University Code of Student Conduct during the exam, you will receive a -100% for the exam and will be reported to the Office of Student Conduct. We also reserve the right to give an automatic F in the course and to make the course ineligible for a grade exclusion.

Preparing for the Exam

When preparing for the exam, the teaching staff recommends that you:

- Review the lecture slides
- Review any related readings linked from lecture topics
- Review the workshop exercises
- Review the lab activities
- Review your class notes

Types of questions:

- Multiple Choice, True/False, short answer
- Scenario questions
    - Based on the lectures and the workshops, a given situation or system will be described, and you'll need to answer the questions.
    - Example: Given a vulnerability, how can an attack exploit it? How can the vulnerability be mitigated?

1. **Course Introduction and Lecture 2**
    - **a.** explain the definition of a security vulnerability
    - **b.** name and define the most common security concepts / security objectives
    - **c.** classify a vulnerability as a design flaw or implementation bug
    - **d.** understand/name the three most popular vulnerability databases
    - **e.** understand what a CVE is and what a CWE is and how the two are related
    - **f.** describe the Common Vulnerability Scoring System (CVSS) and how a development team uses CVSS in relation to CVEs
    - **g.** understand how attackers can use GET and POST requests in attacks

h. understand how attackers can use HTML, hidden variables, GUI controls, and bypass client-side validation
i. where should input validation be done, client-side or server-side?

2. Vulnerability Rankings

a. describe the methodology for identifying the OWASP Top 10
b. describe the methodology for identifying the CWE Top 25
c. compare and contrast the CWE/SANS Top 25 with the OWASP Top 10
d. Who develops the KEV list and what is it based on? How is KEV different from the CWE Top 25 ranking? Why are people interested in the KEV list?
e. Who develops the EPSS list and what is it based on? How is EPSS different from the KEV list? Why are people interested in EPSS?
f. Define exploitability and explain why we care about exploitability.

3. **Broken Access Control (A1)**

a. compare and contrast discretionary, mandatory, role-based access controls (RBAC), attribute-based access control (ABAC)
b. create an access control matrix for a software system
c. describe how broken or incorrect authorization affects the security of software
d. describe techniques for mitigating risks associated with broken or improper authorization
e. Explain the role of session IDs in access control

4. **Cryptographic Failures (A2)**

a. identify and categorize sensitive system data using a classification scheme
b. describe how sensitive data exposure affects the security of software
c. describe techniques for mitigating risks associated with sensitive data exposure
d. describe how improper use of cryptography affects the security of software
e. explain the basics of how encryption and hashing are used to secure data
f. How is data protected from exposure in transit?
g. What is the risk to cryptography with quantum computing?

5. Injection (A3)

a. explain how injection attacks work in the general case
b. explain how to mitigate injection attacks in the general case. What software security principles apply to the mitigation of injection attacks?
c. create user input that produces a SQL injection attack
d. describe techniques for mitigating SQL injection attacks
e. explain how software may be vulnerable to cross-site scripting attacks
f. create user input that produces a cross-site scripting attack
g. describe techniques for mitigating cross-site scripting vulnerabilities
h. What is an allow list? What is a deny list? Which is preferable - filtering with an allow list or a deny list? Why?

i. Name several ways that input can enter a program from the “outside.”

6. **Insecure Design (A4)**
    a. Understand the 21 secure software design principles and the design flaws that can be prevented if these principles are considered.
        i. Hint: take a look at the exercise we did not do because of the snow day.

7. **Security Misconfiguration (A5)**
    a. What attack vectors are opened when web applications are configured incorrectly or insecurely
    b. Name several forms of misconfiguration
    c. Explain the secure-by-default security principle and how it relates to security misconfiguration
    d. Explain the least privilege security principle and how it relates to security misconfiguration
    e. What are some ways to prevent security misconfiguration vulnerabilities?

8. **Vulnerable and Outdated Components (A6)**
    a. How do vulnerable and outdated components increase security risk?
    b. How are attackers using components in the software supply chain to launch attacks?
    c. What is the purpose of the OpenSSF Scorecard project? What are the automated metrics in the Scorecard, and generally, how are they computed?
    d. What do Software Compositional Analysis tools do?
    e. What does it mean to have a pinned dependency? A floating dependency? What are the pros and cons of pinning or floating dependencies?
    f. What is dependency confusion, and what role does pinning/floating have in preventing dependency confusion?
    g. What is typosquatting? How can it be used by attackers?

9. **Identification and Authentication Failures (A7)**
    a. describe how broken or missing authentication affects the security of software
    b. describe techniques for mitigating attacks resulting from broken or missing authentication
    c. describe how allowing excessive authentication attempts affects the security of software
    d. describe techniques for mitigating attacks resulting from improper restriction of excessive authentication attempts
    e. explain why authentication may need to be confirmed on each page or for sensitive transactions.

10. **Software and Data Integrity Failures (A8)**
    a. What is an integrity violation/failure?
    b. How was the Solarwinds Orion attack (of 2020) an integrity failure?

c. What are some ways to protect from software and data integrity failures?

11. Security Logging and Monitoring Failures (A9)

a. What are the types of CWEs commonly associated with logging? Understand CWE related to not enough and too much information.
b. Why is logging not enough, and why must proactive monitoring be done?
c. What security objective(s) is logging/monitoring important for?
d. What is non-repudiation? How can non-repudiation be mitigated?
e. What types of transactions must be logged?

12. Server Side Request Forgery (SSRF) (A10)

a. Describe how an SSRF attack can occur
b. Describe ways to prevent SSRF attacks

13. Other Input Validation Vulnerabilities

a. describe how unrestricted file uploads can affect the security of software; and how can these types of attacks be mitigated?
b. describe how untrusted inputs in security decisions can affect the security of software; and how can these types of attacks be mitigated?
c. describe how path traversal can affect the security of software; and how can these types of attacks be mitigated?
d. Improper limitation of a pathname to a restricted directory
e. describe how URL direction to an untrusted can affect the security of software; and how can these types of attacks be mitigated?

14. Cross-Site Request Forgery (CSRF)

a. explain how software may be vulnerable to cross-site request forgery attacks
b. explain the condition(s) that must be true for a CSRF attack to occur
c. explain how to create a cross-site request forgery attack for a web application
d. describe techniques for mitigating cross-site request forgery vulnerabilities in software

15. Automated Vulnerability Detection

a. Understand how Dynamic Application Security Testing (DAST) works
b. Understand the pros and cons of using DAST
c. Understand how Static Application Security Testing (SAST) works
d. Understand the pros and cons of using SAST
e. What vulnerability discovery techniques are most effective/most efficient for detecting implementation bugs versus design flaws?
f. Compare vulnerability detection techniques (SAST, DAST, security testing) in the areas of: number of false positive vulnerabilities, exploitability, fix time, and relative numbers of vulnerabilities it will likely find?

16. **Security Testing**
    a. What is penetration testing?
    b. What is the difference between opportunistic penetration testing and systematic penetration testing?
    c. What are the elements of the black box test case template?
    d. What is the difference between functional security testing and adversarial security testing?
    e. Explain the purpose of and how to use the OWASP Application Security Verification Standard (ASVS)

17. **Software Secrets / Checked-in Credentials**
    a. What are the types of (hard-coded) software secrets/credentials that can be checked in?
    b. How prevalent are software secrets committed? Is the industry seeing more or fewer checked-in credentials over time?
    c. How are checked-in credentials detected? How can they be prevented from being checked in?
    d. Do all checked-in credentials pose the same security risk? If not, what makes a secret have more or less security risk?
    e. What are the best practices for storing and managing credentials?
    f. How should discovered checked-in credentials be cleaned up?

18. **Adversarial Behavior**
    a. Be able to define adversary and threat.
    b. Understand why it is important to think about the adversaries and their motivations when designing a system.
    c. What does the MITRE ATT&CK framework contain? What is a tactic? A technique? A procedure? A mitigation? A detection?
    d. What is CAPEC? How does MITRE ATT&CK compare with MITRE CAPEC?

19. **Attack Trees**
    a. Explain the value of attack trees when building security into software
    b. Explain the value of defense trees when building security into software
    c. Create an attack tree for a given attack goal
    d. Create a defense tree for a given attack goal
    e. Be able to estimate and propagate values for cost/difficulty up a tree, such that they can be used for prioritization.
    f. How did we use data from MITRE AT&CK to populate attack and defense trees?

20. **Threat modeling**
    a. Understand the threat modeling 4 questions and when/how they are used.
    b. Know when in the software development lifecycle threat modeling is done
    c. Know how to draw a dataflow diagram (DFD) and elements that are on a DFD.

d. Know the STRIDE model threats and the corresponding desirable properties and mitigations.
e. Know how the STRIDE threats are used with a DFD

21. **Large Language Models (LLMs) and security**
    a. What is vulnerable code versus malicious code?
    b. What does it mean when a LLM hallucinates?
    c. How well do LLMs work for finding malicious code?
    d. How can prompt engineering be used to improve the detection of malicious code?

22. **Software bill of materials (SBOM)**
    a. What is an SBOM, and what is the purpose of an SBOM?
    b. How can an SBOM help software supply chain security?
    c. What is the purpose of a VEX?