UNIT 12: Why Is Software Vulnerable

Estimated Time in Hours: 6

Big Idea(s)	Enduring Understandings	Projects & Major Assignments
1 Ethics 2 Establishing Trust 7 Risk 8 Implications	2.2	- Research the origin of specific vulnerabilities and how they detriment software. - Practice the basics of computer programming. - Investigate secure coding practices as a lead-in to the next unit.

Guiding Questions:

- Why are software development frameworks (DevOps, Agile) used?
- How does the principle of modularity relate to software development?
- How does the principle of simplicity relate to software development?
- Other than for security purposes, how does modularity and simplicity enhance software?
- How can an adversary alter a program's code?
- Why are software updates important?
- How can a developer's intentions differ from how their program is actually used by its consumer?
- Was the Internet designed with security in mind?
- What was the impact of the Morris Worm?

Learning Objectives & Respective Essential Knowledge Statements	Materials	Instructional Activities and Classroom Assessments
7.2.3 LO: Students will be able to explain how the logical malleability of software and hardware can allow an adversary to change a system to meet the adversary's goals rather than the systems original objective.	• Computer, lecture slides, projector, graphic organizers, access to Internet • Introduction to computer science as a career: "What Most Schools Don't Teach." *YouTube*, uploaded by Code.org, 26 Feb 2013,	• At the beginning of this unit, show the linked YouTube video which introduces computer science as a career. This is where students can begin simple computer programming (recommended IDE listed to the left). • Engage students in this unit by putting them in the shoes of a video game developer who rushes a game to launch and suffers from software vulnerabilities. Tie the solutions of these vulnerabilities to modularity and

	https://youtu.be/nKIu9yen5nc • Free, easy-to-use, web-based programming IDE: https://repl.it/ • DevOps: "The DevOps Revolution is Changing Cloud Security – Don't Get Left Behind." Check Point Software Technologies Ltd, *CheckPoint.com*, https://blog.checkpoint.com/2018/01/03/devops-revolution-changing-cloud-security-dont-get-left-behind/ • DevOps introduction video: "What is DevOps? – In Simple English." *YouTube*, uploaded by Rackspace Technology, 12 Dec 2013, https://youtu.be/_I94-tJlovg	simplicity to heighten the student's understanding of these principles. • Introduce DevOps (Development and Operations) or a similar development approach, such as Agile. Explain the main pieces of the lifecycle and anchor the rest of this unit to that development approach. • Ask students which stages of the DevOps/Agile framework and which piece(s) of the CIA Triad the vulnerabilities violate (e.g., if the game crashes frequently, this violates availability). • If given multiple software vulnerabilities, challenge the students to choose which problem is more important and should be addressed first.

2.3.7a EK: The principle of modularity says that individual components are capable of executing a unique part of the desired functionality and is achieved through system design. Because of this modular design, security upgrades can happen in one component without having to overhaul the entire system.		Note: modularity LO/EK is first referenced in Unit 7. • Review the principle of modularity. • Ask students to use modularity to repair one of the game's presented software vulnerabilities. • Other than fixing the software vulnerability, does adding modularity offer any other benefits?
2.3.7b EK: A system's components may be separated and recombined.		• Elaborate on enhancements modularity can provide the game, such as making it easier to add more game levels, new items, better communication features, etc. • If one of the modules breaks or experiences a vulnerability, do you necessarily need to repair the other modules?
2.2.1 LO: Students will describe the principle of simplicity, which is about ensuring that systems are easy to understand, maintain and test so as to be more secure. 2.2.1a EK: Simple designs are easier to understand, maintain and test for security problems.	• Vulnerability types: "Vulnerability Types – CompTIA Security+ SY0-501 – 1.6." *YouTube*, uploaded by Professor Messer, 14 Nov 2017, https://youtu.be/1UNCDsrDTu4	Note: simplicity LO/EK is first referenced in Unit 7. • Review the principle of simplicity. • Challenge students with using the principle of simplicity to repair one of the game's presented software vulnerabilities. • Task students to research the origin of specific vulnerabilities and why it can harm the software. You can task vulnerabilities from the linked YouTube video, or students can watch the video for ideas.

2.2 EU: The simpler you can make the design or implementation of a system, the better you can check whether or not it can be exploited. 2.2.1b EK: Simplicity is also known as "Economy of Mechanism."		• Other than the security benefits of simplicity, how does the principle of simplicity benefit the game in other ways? Students should relate the principle of simplicity to the scenario.
2.2.1c EK: A simple design incorporates a careful analysis of what is needed.	• UI comparison example: "New vs Old UI." Florida International University | Information Technology | PantherSoft, *FIU.edu*, 25 May 2018, https://panthersoft.fiu.edu/uiredesign/new-vs-old-ui/	• Explain the importance of simplicity in the design and functionality of user interfaces. • Ask students to judge which UI is best (example to the left) and determine which uses simplicity.
2.3.7 LO: Students will recognize that the cybersecurity often applies to a system that consists of individual self-sufficient components and the overall security is dependent on the security properties of the components.		• Relate this unit's scenario to the bigger picture. For example, the game consists of several smaller components: the user accounts, game servers, chat systems, etc. • Why do we need to protect each of these individually?
7.2.3a EK: Software is frequently updated to correct both	• "Why it's important to update your software | Update your phone [How	• Now that the student's game code has been repaired, the game itself must be tested, released and deployed (DevOps). Relate this to your chosen framework.

functional errors and security problems.	To] | Support on Three." *YouTube*, uploaded by Three UK, 29 Apr 2016, https://youtu.be/LkToKpX9ZWQ	• Explain the essential importance of software updates. • Show the linked YouTube video to your students. Ask them to list the reasons to update software.
1.2.2a EK: The designer assumptions and user assumptions could differ. Another way to say this, the user may not know the assumptions of the designer for using the tool, leading the user to use the tool in a way the designer never intended. 1.2.2 LO: Students will give examples of where/how tools are used in ways that were not intended by the system designer.	• List of ideas for activity: orange swan. "You've got an old computer, your're crafty, and you spent way too much time watching 'Transformers' as a kid." MetaFilter | Community Weblog, *MetaFilter.com*, 22 Oct 2012, https://www.metafilter.com/121155/Youve-got-an-old-computer-yourre-crafty-and-you-spent-way-too-much-time-watching-Transformers-as-a-kid	• Describe how software developers have to defensively code as users may use the software in ways the designer never considered. • Ask whether the students think it's a good or bad thing for people to find new ways to use software. Why or why not? • As an activity, show them photograph examples of things used in unintended ways. Ask them to identify the specific ways they are uniquely used.
7.2.3b EK: Software changes could come from an adversary that intentionally inserts code to meet the goals of the adversary. 7.2.3c EK: Changes in software code are common and those	• Insider Threat: "Insider Threats in 2 Minutes." *YouTube*, uploaded by Security Innovation, 11 Jul 2018, https://youtu.be/QXnNkSeT6dM	• Explain ways an attacker may attempt to change the source code itself through supply chain attacks or attacks against code repositories. • Show the linked YouTube video about insider threats and ask students these questions (a video viewing guide may be used).

introduced by an adversary are often not easily detected.		- How are insider threats dangerous? - Could an insider threat cause damage to source code? - How? - What are some ways to defend against insider threats?
8.1.1e EK: The violation of system integrity can alter the behavior of critical infrastructure.		• Changing software source code is an attack on its integrity. How would this impact critical infrastructure? • Assign students scenarios and have them research how a breach of integrity would harm the systems (e.g., source code damage to a COVID-19 case tracking app).
8.1.2b EK: Security was not seen as a concern until much of the "infrastructure" for computer networks was in place.	• Morris Worm: "Morris Worm explained." *YouTube*, uploaded by Cyber Security Entertainment, 7 Jul 2018, https://youtu.be/sm3wN8rLi8U	• Demonstrate the Internet's former lack of security by showcasing the Morris worm. Show students the linked YouTube video. • Use a video viewing guide to ask questions about the worm, such as how much monetary damage it caused, where it was launched from, and the type of modern cyber attack it inspired.
8.1.1h EK: Cybersecurity events have led to the development of various cybersecurity career paths and various needs in order to prepare people for these new types of jobs.		• Explore a relevant career, such as software developer.