Unit 5: Securing Applications and Data

Content Area: **Business**
Course(s): **Generic Course**
Time Period: **Semester 2**
Length: **6 weeks**
Status: **Published**

Standards

CS.9-12.8.1.12.CS.1	Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
CS.9-12.8.1.12.CS.2	Model interactions between application software, system software, and hardware.
CS.9-12.8.1.12.CS.3	Compare the functions of application software, system software, and hardware.
CS.9-12.8.1.12.CS.4	Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
CS.9-12.8.1.12.DA.2	Describe the trade-offs in how and where data is organized and stored.
CS.9-12.8.1.12.IC.1	Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
CS.9-12.8.1.12.NI.1	Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing.
CS.9-12.8.1.12.NI.2	Evaluate security measures to address various common security threats.
CS.9-12.8.1.12.NI.3	Explain how the needs of users and the sensitivity of data determine the level of security implemented.
CS.9-12.8.1.12.NI.4	Explain how decisions on methods to protect data are influenced by whether the data is at rest, in transit, or in use.
CAEP.9.2.12.C.3	Identify transferable career skills and design alternate career plans.
CAEP.9.2.12.C.7	Examine the professional, legal, and ethical responsibilities for both employers and employees in the global workplace.

Enduring Understandings

EU 5.1: Software Inherent Flaws as Malicious Vectors

Applications are only as secure as their code's ability to handle unexpected input. When software treats user-submitted data as executable code or fails to strictly enforce system boundaries, adversaries can manipulate databases (SQLi), hijack user sessions (XSS), crash systems (buffer overflows), or steal restricted files (directory traversal).

EU 5.2: Granular Access Control and Data Lifecycle

Securing data requires continuous protection that matches its classification level, its current state, and the specific role of the user trying to access it. Organizations must apply strict access rules (like Bell-LaPadula or least privilege) and technical permissions (like Linux strings) so that data remains shielded whether it is sitting on a hard drive, moving across a network, or active in memory.

EU 5.3: Shared-Secret Cryptographic Confidentiality

Symmetric encryption provides high-speed confidentiality for vast amounts of data, but its entire security model depends on the total secrecy of a single shared key. Algorithms like AES balance speed against brute-force resistance based on key length (128, 192, or 256 bits), meaning defenders must carefully manage keys to ensure encrypted data cannot be decrypted by unauthorized parties.

EU 5.4: Trust Architecture through Key Disparity

Asymmetric cryptography solves the problem of securely sharing secret keys by splitting encryption and decryption duties between two mathematically linked keys. By publicizing an encryption key while fiercely guarding a private decryption key, asymmetric algorithms (like RSA and ECC) allow completely untrusted parties to build encrypted tunnels and verify digital identities across the open internet.

EU 5.5: Proactive Software Defenses

True application security is baked into the software architecture from the very beginning, not patched on afterward. By adopting a "secure by default" mindset, developers can use strict input validation (checking if data fits the rules) and input sanitization (stripping out dangerous control characters) to strip malicious commands of their power before they ever reach a database or system memory.

EU 5.6: Immutable Evidence and Retrospective Discovery
When application defenses inevitably face attacks, cryptographic proofs and structured log metrics are required to uncover the breach. While server logs provide the clues needed to reconstruct web exploits, cryptographic hash functions generate digital fingerprints that instantly signal whether a file's integrity has been altered, allowing teams to separate legitimate software changes from silent data tampering.

Essential Questions

Topic 5.1: Application and Data Vulnerabilities & Attacks
How do flaws in software design—such as improper input handling or memory management—allow adversaries to execute SQL injection, XSS, buffer overflows, and directory traversals to compromise the CIA triad?

Topic 5.2: Protecting Applications and Data - Managerial Controls and Access Controls
How do data states, classification types, and structural access control models (like RBAC, MAC, and Linux permissions) work together to strictly enforce the principle of least privilege?

Topic 5.3: Protecting Stored Data with Cryptography
What are the mathematical, performance, and operational trade-offs between symmetric block and stream

ciphers (such as AES) when securing data at rest and in transit?

Topic 5.4: Asymmetric Cryptography
How does the mathematical relationship between public and private key pairs establish secure, uninterceptable communications, and what roles do RSA and ECC play in Public Key Infrastructure (PKI)?

Topic 5.5: Protecting Applications
How do the philosophies of "secure by design" and "secure by default" leverage input validation and input sanitization to programmatically neutralize code injection and directory threats?

Topic 5.6: Detecting Attacks on Data and Applications
How do defenders use cryptographic hashing, server log analysis, and honeypots to retrospectively audit file integrity and identify active web-application exploits?

Knowledge and Skills

Topic 5.1: Application and Data Vulnerabilities & Attacks

- Identify the three core file/system vulnerabilities adversaries exploit before specific application attacks are needed (5.1.A)
- Explain how SQL injection, XSS (Type I and II), buffer overflow, and directory traversal each exploit application vulnerabilities, including the specific flaw each requires (5.1.B)
- Assess risks from application and data vulnerabilities using the High/Moderate/Low framework, identifying which CIA property is compromised in each case (5.1.C)

Topic 5.2: Protecting Applications and Data - Managerial Controls and Access Controls

- Explain how data state (at rest, in transit, in use) and classification (PII/PHI/PCI) determine the type and degree of security required (5.2.A)
- Identify the two managerial controls that govern application and data security: cryptography policy and web application security policy (5.2.B)
- Select the appropriate access control model for a given scenario: RBAC, RuBAC, DAC, or MAC (5.2.C)
- Apply Bell-LaPadula properties (WURD) and the principle of least privilege (5.2.C)
- Read, write, and interpret Linux file permission strings and chmod commands in both numeric and symbolic forms (5.2.D)

Topic 5.3: Protecting Stored Data with Cryptography

- Define plaintext, ciphertext, key, and keyspace; calculate keyspace size (2n) and average brute-force guesses (2n-1) for an n-bit key (5.3.A)
- Distinguish symmetric from asymmetric encryption and block ciphers from stream ciphers (5.3.A)
- Describe AES properties: symmetric, block cipher, 128-bit block size, key lengths 128/192/256, trade-

off between key length and performance (5.3.B)
- Interpret and write OpenSSL AES encrypt/decrypt commands (5.3.B.4)

Topic 5.4: Asymmetric Cryptography

- Explain how asymmetric key pairs work and determine the correct key to use when sending or receiving encrypted data (5.4.A)
- Explain why private key security is critical and what must happen if a private key is compromised (5.4.A)
- Calculate keyspace size (2n) and average brute-force guesses (2n-1); explain the security-speed trade-off for key length (5.4.B)
- Identify RSA and ECC as common asymmetric algorithms and interpret/write OpenSSL asymmetric commands (5.4.C)

Topic 5.5: Protecting Applications

- Identify the three principles of secure by design and the concept of secure by default (5.5.A)
- Explain what control characters are and how input sanitization protects applications against SQL injection, XSS, and directory traversal (5.5.B)
- Distinguish input sanitization from input validation (5.5.B)

Topic 5.6: Detecting Attacks on Data and Applications

- Explain how accounting and log analysis reveal malicious activity; describe honeypots and their limitations (5.6.A)
- Use hash commands in PowerShell, BASH, and zsh to verify file integrity; explain what a hash change indicates and what it cannot detect (5.6.D)
- Evaluate detective controls using cost, sensitivity, and classification criteria; distinguish real-time from retrospective detection; identify false negatives (5.6.B–C)

Transfer Goals

Students will be able to recognize the differing value of their personal data (like medical records, financial data, or login credentials) and consciously choose services that protect that data properly depending on whether it is stored or shared.

Students will be able to confidently navigate permission settings on any platform—from shared cloud folders and social media apps to local computer operating systems—ensuring that other people only have the exact access they truly need.

Students will be able to identify signs of compromised web applications (such as broken page elements or unexpected script behavior) and navigate online forms with an awareness of how their typed inputs can impact underlying systems.

Students will be able to use real-world clues—like browser security certificates, encryption locks, and digital signatures—to confidently verify the identity of a website or sender before trusting them with confidential

information.

Students will be able to use basic built-in tools (like hash checks) to verify that downloaded software, files, or documents are authentic, safe, and free from quiet tampering before opening or installing them.

Resources

AP Cybersecurity Course and Exam Description, Effective Fall 2026, College Board

AP Cybersecurity Course Framework, Effective 2026-2027 School Year, College Board

AP Cybersecurity Curriculum, by APCSExamPrep.com

Cybersecurity Curriculum Materials, by Cyber.org

Cyber Range, provided by Cyber.org