The Time Is Now for College/Employer Partnerships for Secure Software Development Apprenticeships

CyberUSA

January 24, 2019

Girish Seshagiri
Unique Point in Time

- Software is the most important technology contributing to global high standards of living
- Cyber-attacks are increasing; 90% of attacks result from exploiting defects in software
- 50 billion devices coming online during the next 5 years (IOT)
- 1.8 million cybersecurity jobs unfilled by 2022
- Greatest economic development opportunity since the GI Bill post WW2
- High priority need for workforce capable of developing software that is secure from cyber-attacks
My Goals/Ask

• Make the case that there is an urgent need to connect education directly to a job and collaboration is needed to rapidly scale implementation of employer-driven dual model apprenticeship for secure software development (because nothing else works)

• Motivate you to collaborate with other employers, colleges and workforce intermediaries with common interest in Degrees/Certificates in Secure Software Development

• Invite you to join the national conversation on cybersecurity apprenticeships by joining the monthly meetings of NICE Apprenticeship Working Group
Federal IT Spend in 2015

The federal government spent more than 75 percent of the total amount budgeted for information technology (IT) for fiscal year 2015 on operations and maintenance (O&M) investments. Specifically, 5,233 of the government's approximately 7,000 IT investments are spending all of their funds on O&M activities.

Federal IT Spend in 2017

![Bar chart showing Federal IT Spend in 2017 with data from fiscal years 2010 to 2017. The chart indicates a decline in spending since 2010, with a breakdown of spending categories: Development, modernization, and enhancement (red) and Operations and maintenance (blue). Source: GAO analysis of agency data. | GAO-16-896T]
Cybersecurity

• **Defective software** is insecure
  • 90% of attacks are successful by exploiting defects in the software application layer
  • 1 in 20 software defects are vulnerabilities that can be exploited to launch cyberattacks
  • “If you have a quality problem, you have a security problem”

• **Consequences of poor quality software**
  • Impacts - Democracy, loss of life and limb besides just financial loss
  • Potentially more catastrophic than a bridge falling down

• High priority national goal to move from reactive to proactive – from threat detection to threat prevention
Cybersecurity Supply/Demand Heat Map

Cybersecurity talent gaps exist across the country. Closing these gaps requires detailed knowledge of the cybersecurity workforce in your region. This interactive heat map provides a granular snapshot of demand and supply data for cybersecurity jobs at the state and metro area levels, and can be used to grasp the challenges and opportunities facing your local cybersecurity workforce.

National level

<table>
<thead>
<tr>
<th>TOTAL CYBERSECURITY JOB OPENINGS</th>
<th>SUPPLY OF CYBERSECURITY WORKERS</th>
<th>GEOGRAPHIC CONCENTRATION</th>
<th>TOP CYBERSECURITY JOB TITLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>313,735</td>
<td>Very Low</td>
<td></td>
<td>Cyber Security Engineer</td>
</tr>
<tr>
<td>TOTAL EMPLOYED CYBERSECURITY WORKFORCE</td>
<td>Average Location Quotient</td>
<td></td>
<td>Cyber Security Analyst</td>
</tr>
<tr>
<td>715,715</td>
<td></td>
<td></td>
<td>Network Engineer / Architect</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Cyber Security Manager /</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Systems Engineer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Software Developer / Engineer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Systems Administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Vulnerability Analyst /</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Penetration</td>
</tr>
</tbody>
</table>
Dual Model Apprenticeship – “Because nothing else works”
Administration and Apprenticeship

TASK FORCE ON APPRENTICESHIP EXPANSION

Final Report to:

The President of the United States

May 10, 2018
Meanwhile, the American higher education system is churning out a pool of indebted job seekers who are not equipped to meet the skills needs of many employers in the modern American economy.
Nine Facts About Higher Education

• Students who enrolled in public colleges three years ago now face tuition as much as 50% to 80% higher in some states.

• 45% of bachelor’s degree holders are underemployed.

• Only 28% of recent college graduates strongly agree that their university prepared them well for life outside of college.

• Only 37% of recent graduates strongly agree that their education was worth the cost.

• Only 24% of recent graduates say they had a good job waiting for them after they graduated.

• Only 11% of business leaders say college grads are prepared for the modern workforce while 96% of chief academic officers at higher education institutions say their institution is effective at preparing students for the world of work.

• By 2020, 65% of jobs will require a postsecondary education beyond high school. At current pace, the U.S. will fall at least 5 million degrees short.

Source: USCCF
Congress and Apprenticeships

Addressing the skills gap in America through apprenticeships

BY SEN. CHRIS COONS (D-DEL.), KEVIN O’CONNOR, NORM ABRAM, RICHARD TRETWEY, TOM SILVA, ROGER COOK, NATHAN GILBERT, OPINION CONTRIBUTORS — 05/08/18 09:30 AM EDT
THE VIEWS EXPRESSED BY CONTRIBUTORS ARE THEIR OWN AND NOT THE VIEW OF THE HILL

Dual Model Apprenticeship – Core Components

What are the Components of Registered Apprenticeship?

01 BUSINESS INVOLVEMENT
Employers are the foundation of every Registered Apprenticeship program.

02 STRUCTURED ON-THE-JOB TRAINING
Apprentices receive on-the-job training from an experienced mentor for typically not less than one year.

03 RELATED INSTRUCTION
Apprenticeships combine on-the-job learning with technical education at community colleges, technical schools, apprenticeship training schools, provided online or at the job site.

04 REWARDS FOR SKILL GAINS
Apprentices receive increases in wages as they gain higher level skills.

05 NATIONAL OCCUPATIONAL CREDENTIAL
Registered Apprenticeship programs result in a nationally-recognized credential – a 100% guarantee to employers that apprentices are fully qualified for the job.
Principles for Growing and Sustaining the Nation’s Cybersecurity Workforce

- Widen the aperture of candidate pipelines by adopting New Collar principles (e.g., stop making degrees a mandatory requirement for jobs).
- Revitalize job postings to be engaging and to focus on the core requirements; don’t “over-spec” the requirements.
- Simplify career models and build transparency; leverage the National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework where possible for consistency.
- Think about new ways of hiring and training, where the adjacent technical and professional skills take priority, and cybersecurity skills can be taught.
- **Launch apprenticeship programs to train candidate pipelines at scale.**
- Commit to employee development – support expanded and focused training in general.
- Adopt key principles for productive partnerships and programs – maximize impact by partnering with a focus on scale.
- Make cybersecurity everyone’s business – continue to advocate and make cybersecurity education widely available.

Source: Aspen Institute Cybersecurity Group
Why Apprenticeships?

Because Nothing Else Works
Categories of Cybersecurity Work

- Specialty Areas (33) – Distinct areas of cybersecurity work;
  - Work Roles (52) – The most detailed groupings of IT, cybersecurity or cyber-related work, which include specific knowledge, skills, and abilities required to perform a set of tasks.
    - Tasks – Specific work activities that could be assigned to a professional working in one of the NCWF’s Work Roles; and,
  - Knowledge, Skills, and Abilities (KSAs) – Attributes required to perform Tasks, generally demonstrated through relevant experience or performance-based education and training.

- Audience:
  - Employers
  - Current and Future Cybersecurity Workers
  - Training and Certification Providers
  - Education Providers
  - Technology Providers
## NICE Workforce Framework Categories

<table>
<thead>
<tr>
<th>NICE CYBERSECURITY WORKFORCE FRAMEWORK CATEGORY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECURELY PROVISION</td>
<td>Conceptualizes, designs, procures, and/or builds secure information technology (IT) systems, with responsibility for aspects of system and/or network development</td>
</tr>
<tr>
<td>OPERATE AND MAINTAIN</td>
<td>Provides the support, administration, and maintenance necessary to ensure effective and efficient information technology (IT) system performance and security</td>
</tr>
<tr>
<td>OVERSEE AND GOVERN</td>
<td>Provides leadership, management, direction, or development and advocacy so the organization may effectively conduct cybersecurity work</td>
</tr>
<tr>
<td>PROTECT AND DEFEND</td>
<td>Identifies, analyzes, and mitigates threats to internal information technology (IT) systems and/or networks</td>
</tr>
<tr>
<td>ANALYZE</td>
<td>Performs highly-specialized review and evaluation of incoming cybersecurity information to determine its usefulness for intelligence</td>
</tr>
<tr>
<td>COLLECT AND OPERATE</td>
<td>Provides specialized denial and deception operations and collection of cybersecurity information that may be used to develop intelligence</td>
</tr>
<tr>
<td>INVESTIGATE</td>
<td>Investigates cybersecurity events or crimes related to information technology (IT) systems, networks, and digital evidence</td>
</tr>
</tbody>
</table>
% of Job Openings by NICE Framework Category

- Collect and Operate: 6%
- Investigate: 1%
- Operate and Maintain: 25%
- Securely Provision: 24%
- Protect and Defend: 16%
- Analyze: 16%
- Oversee and Govern: 12%
## NICE Framework: Software Development Specialty Area

<table>
<thead>
<tr>
<th>Category</th>
<th>Specialty Area</th>
<th>Work Role</th>
<th>Work Role ID</th>
<th>Work Role Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Securely Provision (SP)</td>
<td>Software Development (DEV)</td>
<td>Software Developer</td>
<td>SP-DEV-001</td>
<td>Develops, creates, maintains, and writes/codes new (or modifies existing) computer applications, software, or specialized utility programs.</td>
</tr>
<tr>
<td></td>
<td>Secure Software Assessor</td>
<td>SP-DEV-002</td>
<td></td>
<td>Analyzes the security of new or existing computer applications, software, or specialized utility programs and provides actionable results.</td>
</tr>
</tbody>
</table>
Keyword Search

Search Descriptions

A0047: Ability to develop secure software according to secure software deployment methodologies, tools, and practices.

Abilities ID: A0047

Description: Ability to develop secure software according to secure software deployment methodologies, tools, and practices.

Work Roles:

Work Role ID: SP-DEV-001
Work Roles: Software Developer
Work Role Description: Develops, creates, maintains, and writes/codes new (or modifies existing) computer applications, software, or specialized utility programs.
Category: Securely Provision
Specialty Area(s): Software Development
Software Developer

Develops, creates, maintains, and writes/codes new (or modifies existing) computer applications, software, or specialized utility programs.

**Category:** Securely Provision  **Specialty Area:** Software Development

**Abilities**

- **A0007:** Ability to tailor code analysis for application-specific concerns.
- **A0021:** Ability to use and understand complex mathematical concepts (e.g., discrete math).
- **A0047:** Ability to develop secure software according to secure software deployment methodologies, tools, and practices.
- **A0123:** Ability to apply cybersecurity and privacy principles to organizational requirements (relevant to confidentiality, integrity, availability, authentication, non-repudiation).
- **A0170:** Ability to identify critical infrastructure systems with information communication technology that were designed without system security considerations.

**Knowledge**

- **K0001:** Knowledge of computer networking concepts and protocols, and network security methodologies.
- **K0002:** Knowledge of risk management processes (e.g., methods for assessing and mitigating risk).
- **K0003:** Knowledge of laws, regulations, policies, and ethics as they relate to cybersecurity and privacy.
- **K0004:** Knowledge of cybersecurity and privacy principles.
- **K0005:** Knowledge of cyber threats and vulnerabilities.
- **K0006:** Knowledge of specific operational impacts of cybersecurity lapses.
- **K0014:** Knowledge of complex data structures.
- **K0016:** Knowledge of computer programming principles.
- **K0017:** Knowledge of organizational enterprise information security architecture.
Tasks

T0009: Analyze information to determine, recommend, and plan the development of a new application or modification of an existing application.

T0011: Analyze user needs and software requirements to determine feasibility of design within time and cost constraints.

T0013: Apply coding and testing standards, apply security testing tools including "fuzzing" static-analysis code scanning tools, and conduct code reviews.

T0014: Apply secure code documentation.

T0022: Capture security controls used during the requirements phase to integrate security within the process, to identify key security objectives, and to maximize software security while minimizing disruption to plans and schedules.

T0026: Compile and write documentation of program development and subsequent revisions, inserting comments in the coded instructions so others can understand the program.

T0034: Confer with systems analysts, engineers, programmers, and others to design application and to obtain information on project limitations and capabilities, performance requirements, and interfaces.

T0040: Consult with engineering staff to evaluate interface between hardware and software.

T0046: Correct errors by making appropriate changes and rechecking the program to ensure that desired results are produced.

T0057: Design, develop, and modify software systems, using scientific analysis and mathematical models to predict and measure outcome and consequences of design.

T0077: Develop secure code and error handling.

T0100: Evaluate factors such as reporting formats required, cost constraints, and need for security restrictions to determine hardware configuration.

T0111: Identify basic common coding flaws at a high level.

T0117: Identify security implications and apply methodologies within centralized and decentralized environments across the enterprise's computer systems in software development.

T0118: Identify security issues around steady state operation and management of software and incorporate security measures that must be taken when a product reaches its end of life.

T0171: Perform integrated quality assurance testing for security functionality and resiliency attack.

T0176: Perform secure programming and identify potential flaws in codes to mitigate vulnerabilities.
Contact

Girish Seshagiri

girish.seshagiri@ishpi.net

703 426-2790
Software Assurance Curriculum for Two-Year Associates in Applied Science Degree

Carol Woody, Ph.D.
Copyright 2019 Carnegie Mellon University. All Rights Reserved.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

CERT® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM18-1161
Importance of Software Assurance to Today’s Environment
Software Reliance is Rapidly Expanding


From 1997 to 2012, software industry production grew from $149 billion to $425 billion
Anyone Can Write Software but is it Good?

How To Raise The Next Zuckerberg: 6 Coding Apps For Kids
http://readwrite.com/2013/04/19/how-to-raise-the-next-zuck-6-coding-apps-for-kids/

TYNKER - We Empower KIDS to Become Makers
https://www.tynker.com/

How and Why to Teach Your Kids to Code

Best-in-class code:  <600 defects per MLOC
Very good code:   600 to 1,000 defects per MLOC
Average quality code:  6000 defects per MLOC
Up to 5% of defects are vulnerabilities
84% of Security Breaches Exploit Software Applications

Breaking this cycle will require engineering of the software we use to handle the realities of the operational environment. All fielded software needs good cybersecurity. However,

- “76% of U.S. developers use no secure application program process”\(^3\)
- “More than 40% of software developers globally say that security isn't a top priority for them”\(^4\)

1. Clark, Tim, Most cyber Attacks Occur from this Common Vulnerability, Forbes. 03-10-2015
2. Feiman, Joseph, Maverick Research: Stop Protecting Your Apps; It's Time for Apps to Protect Themselves, Gartner. 09-25-2014. G00269825
Preparing the Incoming Workforce to Handle Software Assurance
Software Assurance Curriculum Project

Goals: Develop software assurance curricula
Define transition strategies for implementation

Community Outreach
- 20+ Published Papers
- 7 SEI reports
- 15+ talks, webinars, podcasts, media
- Thousands of downloads
- LinkedIn group of 500+ members
- Course materials and videos

Integrated into course offerings
- Carnegie Mellon University
- Stevens Institute of Technology
- US Air Force Academy
- University of Detroit Mercy
- University of Houston
- (ISC)²

Transition
- Degree offerings
  - MSwA Curriculum Design:
    - Polytechnic University of Madrid
  - Community College Programs:
    - Illinois Central College
    - Alamo Colleges
    - Lincoln Land Community College
- SwA Courses
  - Assurance Management
  - Assured Software Dev’t
  - Exec Course

Professional Society Recognition

https://www.sei.cmu.edu/education-outreach/curricula/software-assurance/index.cfm
SwA Curriculum Project Objectives

Improve the state of software assurance education

Develop a Master of Software Assurance Reference Curriculum (Volume I)

Identify educational offerings at other levels

• Undergraduate (Volume II)
• MSwA Syllabi (Volume III)
• Community College (Volume IV)
• Integration with IS Curricula (Technical Note)
Professional Society Recognition

IEEE Recognition

The MSwA curriculum was recognized by the IEEE Computer Society. Its notification follows:

At the meeting of the IEEE Computer Society Board of Governors it was passed:

MOVED, that the IEEE Computer Society Board of Governors recognizes the SEI CMU/SEI-2010-TR-005 Reference Curriculum as appropriate for a Masters Program in Software Assurance for a period of 5 years beginning in 1 August 2010.

Statement: The curriculum recommendation could contain a statement similar to “The IEEE Computer Society recognizes this curriculum recommendation as appropriate for a Masters Program in Software Assurance,” signifying that the Society considers it suitable for its stated purpose. If the curriculum recommendation is appropriate as a model for similar efforts, the statement should indicate that designation.

IEEE published an article about its recognition of the MSwA curriculum at http://www.computer.org/portal/web/pressroom/20101213MSWA.

ACM Recognition

The MSwA curriculum was also recognized by the Association for Computing Machinery (ACM) Education Board. This is identical to the IEEE recognition.
Community College Report

An ACM committee on two-year degree programs, led by Elizabeth Hawthorne, partnered with the SEI team to develop this report.

The report includes

• discussion of existing curricula related to software security that are suitable for community colleges
• target audience
• course outlines
• identification of resources
Community College Courses

**Target audience:** Students planning to transfer to a four-year program, students with prior undergraduate technical degrees who wish to become more specialized in software assurance

**Courses:**
- Computer Science I, II, and III
- Introduction to Computer Security
- Secure Coding
- Introduction to Assured Software Engineering

Students will not necessarily take all six courses, nor will the courses necessarily match courses in four-year colleges with similar titles.
## Computer Science I

<table>
<thead>
<tr>
<th>Topic</th>
<th>Bloom's Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure coding (2 hours): data protection techniques of data encapsulation, information hiding and integrity, and strict data typing</td>
<td>A</td>
</tr>
<tr>
<td>Fundamental programming constructs (11 hours): basic syntax and semantics of a higher-level language; variables (scope and lifetime), types, expressions, and assignment; self-documentation; standard and file input/output; conditional and iterative control structures; structured decomposition; pseudo-random number generator</td>
<td>A</td>
</tr>
<tr>
<td>Fundamental algorithms and problem-solving (6 hours): problem-solving strategies; the role of algorithms in the problem-solving process; implementation strategies for algorithms; debugging strategies; the concept and properties of algorithms</td>
<td>A</td>
</tr>
<tr>
<td>Fundamental data structures (6 hours): primitive types, arrays, records, strings, references</td>
<td>A</td>
</tr>
<tr>
<td>Object-oriented principles (6 hours): abstraction, objects, classes, methods, parameter passing, encapsulation, inheritance, polymorphism</td>
<td>A</td>
</tr>
<tr>
<td>Program development (3 hours): program development phases, with emphasis on design, implementation, and testing and debugging strategies</td>
<td>A</td>
</tr>
<tr>
<td>Software tools and integrated development environment (IDE) (2 hours): compiling, interpreting, linking, executing, testing, and debugging</td>
<td>A</td>
</tr>
<tr>
<td>Programming languages (1 hour): comparison of object-oriented, procedural, functional programming</td>
<td>C</td>
</tr>
<tr>
<td>Human-computer interaction (1 hour): sound design concepts and fundamental graphical interface design; standard API graphics</td>
<td>C</td>
</tr>
<tr>
<td>Machine-level representation of data (1 hour): overview of the storage of instructions, numbers, and characters in a Von Neumann machine</td>
<td>C</td>
</tr>
<tr>
<td>Ethical conduct (1 hour): codes of ethics and responsible conduct; intellectual property, copyright, and plagiarism; “Ten Commandments for Computer Ethics”</td>
<td>C</td>
</tr>
<tr>
<td>Overview of operating systems (1 hour): role and purpose of operating systems; simple file management</td>
<td>C</td>
</tr>
<tr>
<td>Historical context of computing (1 hour): history of computing ideas, computing, and programming</td>
<td>K</td>
</tr>
</tbody>
</table>
## Computer Science II

<table>
<thead>
<tr>
<th>Topic</th>
<th>Bloom’s Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure coding (3 hours): buffer overflows; memory leaks; malicious code; unauthorized and back-door access; security-aware exception handling</td>
<td>A</td>
</tr>
<tr>
<td>Software development (4 hours): software life cycle; test case design; software tools; debuggers and simulators; characteristics of maintainable software; program code verification and data validation; software inspection</td>
<td>A</td>
</tr>
<tr>
<td>Object-oriented programming (7 hours): encapsulation and information hiding; inheritance; class hierarchies; polymorphism; abstract and interface classes</td>
<td>A</td>
</tr>
<tr>
<td>Object-oriented design and modeling (5 hours): class constructors and destructors; abstract data types (ADTs); reusable software components; APIs; modeling tools; class diagrams</td>
<td>A</td>
</tr>
<tr>
<td>Intermediate programming constructs (3 hours): cohesion and decoupling; assertions, including pre/post conditions and loop invariants; software reuse; self-documentation</td>
<td>A</td>
</tr>
<tr>
<td>Intermediate computing algorithms (5 hours): searching; sorting; recursive algorithms; complexity of algorithms</td>
<td>A</td>
</tr>
<tr>
<td>Intermediate data structures (7 hours): built-in; programmer-created; dynamic</td>
<td>A</td>
</tr>
<tr>
<td>Event-driven programming (4 hours): graphics API; event creation; event-handling methods; exception handling</td>
<td>A</td>
</tr>
<tr>
<td>Human-computer interaction (2 hours): sound design concepts; interfaces between people and technology</td>
<td>C</td>
</tr>
<tr>
<td>Simple database integration (1 hour): database I/O; embedded SQL queries; SQL injection</td>
<td>C</td>
</tr>
<tr>
<td>Societal and professional issues (1 hour): computing and the internet; social impact of computing; privacy</td>
<td>C</td>
</tr>
</tbody>
</table>
## Computer Science III

<table>
<thead>
<tr>
<th>Topic</th>
<th>Bloom's Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software assurance (3 hours): conformance with assurance coding standards and practices, trustworthiness, and predictable execution testing; quality reviews; engineering and security tradeoffs; risks and liabilities of computer-based systems; fault prevention in software life-cycle stages; intentional and unintentional software security vulnerabilities.</td>
<td>A</td>
</tr>
<tr>
<td>Formal computing algorithms (8 hours): efficiency of various sorting and searching algorithms; hashing; collision-avoidance strategies; binary search trees; depth- and breadth-first traversals; shortest-path algorithms; minimum spanning tree; transitive closure; topological sort</td>
<td>A</td>
</tr>
<tr>
<td>Canonical data structures (7 hours): stacks; queues; linked lists; hash tables; trees; graphs</td>
<td>A</td>
</tr>
<tr>
<td>Recursion (7 hours): recursive mathematical functions; divide-and-conquer, first-and-rest, and last-and-rest strategies; backtracking; recursion with linked lists; trees; graphs</td>
<td>A</td>
</tr>
<tr>
<td>Software reuse (3 hours): design patterns; parametric polymorphism (templates or generics); code libraries; container classes and iterators</td>
<td>A</td>
</tr>
<tr>
<td>Human-computer interaction (2 hours): universal principles; human-centered considerations; usability testing and verification; design tradeoffs; secure user interfaces</td>
<td>C</td>
</tr>
<tr>
<td>Software engineering (4 hours): standard approaches and implementation tools for analysis and design; measurement and metrics; software life-cycle stages, processes, and documentation; software process maturity scale</td>
<td>C</td>
</tr>
<tr>
<td>Algorithmic strategies (2 hours): brute-force; greedy; branch-and-bound; heuristics; pattern matching; string/text</td>
<td>C</td>
</tr>
<tr>
<td>Basic algorithmic analysis (3 hours): asymptotic analysis of upper and average complexity bounds; best, average, and worst case behaviors; big O and little o notations; standard complexity classes; empirical measurements of performance; time and space tradeoffs; recurrence relations</td>
<td>C</td>
</tr>
<tr>
<td>Concurrency (2 hours): threads; scheduling, synchronization and timing; multi-threaded programs; race conditions</td>
<td>C</td>
</tr>
<tr>
<td>Professionalism (1 hour): standards of professional behavior; professional computing societies and publications; professional responsibilities and liabilities; ACM Code of Conduct; career paths in computing</td>
<td>C</td>
</tr>
</tbody>
</table>
# Introduction to Computer Security

<table>
<thead>
<tr>
<th>Topic</th>
<th>Bloom’s Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security goals and fundamentals: confidentiality, integrity, availability, reliability, etc.</td>
<td>K</td>
</tr>
<tr>
<td>Secure systems: types, models, design, changes to non-secure systems; comparative analysis</td>
<td>C</td>
</tr>
<tr>
<td>Access controls: controlling access to resources, access matrix model, access control lists and capability lists; mandatory controls, originator controls</td>
<td>C</td>
</tr>
<tr>
<td>Networks and security: internet security architecture, internet protocols, implementation considerations; firewalls</td>
<td>C</td>
</tr>
<tr>
<td>Integrity: cryptographic checksums, malicious logic, viruses, Trojan horses; defenses, prevention</td>
<td>K</td>
</tr>
<tr>
<td>Cryptography fundamentals: classical, public key; implementation problems</td>
<td>K</td>
</tr>
<tr>
<td>Authentication: passwords</td>
<td>C</td>
</tr>
<tr>
<td>Attacks: software attacks (malicious code, buffer overflows, social engineering, injection attacks, and related defense tools); network attacks (denial of service, flooding, sniffing and traffic redirection, defense tools and strategies); website attacks (cross-site scripting)</td>
<td>K</td>
</tr>
<tr>
<td>Management: planning for security; introduction to risk assessment and management; business cases; regulatory compliance and legal issues; Federal Information Security Management Act; and business continuity/disaster planning</td>
<td>K</td>
</tr>
<tr>
<td>Security standards in government and industry: NIST 800-39 (risk management), NIST 800-53 (security controls), ISO 27001, and ISO 27002; sample corporate and institutional security policies</td>
<td>K</td>
</tr>
<tr>
<td>Security issues in requirements, architecture, design, implementation, testing, operation, maintenance, acquisition, and services</td>
<td>K</td>
</tr>
<tr>
<td>Ethics and professionalism as related to computer security</td>
<td>K</td>
</tr>
</tbody>
</table>
## Secure Coding

<table>
<thead>
<tr>
<th>Topic</th>
<th>Bloom’s Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview of security vulnerabilities and risks in software: Common Weakness Enumeration (CWE), Open Web Application Security Project (OWASP) Top 10</td>
<td>C</td>
</tr>
<tr>
<td>Data protection: methods for preventing unauthorized access or manipulation of data</td>
<td>AP</td>
</tr>
<tr>
<td>Input validation and user authentication</td>
<td>AP</td>
</tr>
<tr>
<td>Memory management: buffer overflow, memory corruption, and privilege violations</td>
<td>AP</td>
</tr>
<tr>
<td>Integer overflow and misuse of strings and pointers</td>
<td>AP</td>
</tr>
<tr>
<td>Communication vulnerabilities: concurrency, secure inter-process communication and authorization, authentication and networking protocols</td>
<td>AP</td>
</tr>
<tr>
<td>Unit testing for security vulnerabilities: fuzzing, abuse cases</td>
<td>AP</td>
</tr>
<tr>
<td>Code review: formal inspections and static analysis</td>
<td>AP</td>
</tr>
<tr>
<td>Vulnerabilities in modern languages: insecurities in Java and hypertext preprocessor (PHP)</td>
<td>C</td>
</tr>
<tr>
<td>Standard risk mitigation strategies and resources: coding standards, enterprise security API (ESAPI)</td>
<td>C</td>
</tr>
<tr>
<td>Professional development: OWASP, certification</td>
<td>C</td>
</tr>
</tbody>
</table>
# Introduction to Assured Software Engineering

<table>
<thead>
<tr>
<th>Topic</th>
<th>Bloom’s Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to software project management: project planning, estimation, configuration management, risk management; and software security process models: Building Security In Maturity Model (BSIMM), OWASP Software Assurance Maturity Model (SAMM), Microsoft Software Development Lifecycle (SDL)</td>
<td>C</td>
</tr>
<tr>
<td>Role of assured software engineering: software engineering for assurance and its place as an engineering discipline</td>
<td>C</td>
</tr>
<tr>
<td>Requirements analysis: requirements analysis for functional and quality requirements</td>
<td>AP</td>
</tr>
<tr>
<td>Introduction to software architecture: introduction to software architecture, including architectural patterns (pipe &amp; filter, MVC), client-server computing</td>
<td>C</td>
</tr>
<tr>
<td>Use and misuse cases: use cases, misuse cases, and user-centered design</td>
<td>C</td>
</tr>
<tr>
<td>Design patterns: abstraction-occurrence, composite, player-role, singleton, observer, delegation, facade, adapter, etc.</td>
<td>C</td>
</tr>
<tr>
<td>UML: Review of object-oriented principles, UML class diagrams, and object-oriented analysis</td>
<td>AP</td>
</tr>
<tr>
<td>Domain modeling: examples of building class diagrams to model various domains</td>
<td>C</td>
</tr>
<tr>
<td>Reusable technologies: Review of reusable technologies as a basis for software engineering, risks associated with reuse (e.g. Ariane)</td>
<td>C</td>
</tr>
<tr>
<td>Software behavior: representing software behavior: sequence diagrams, state machines, activity diagrams, correctness under all conditions of use</td>
<td>AP</td>
</tr>
<tr>
<td>Verification and validation: Inspections and reviews, integration, system, and acceptance testing</td>
<td>AP</td>
</tr>
</tbody>
</table>
Additional Resources
Publication to Support the Curriculum

Released November 2016 as part of the SEI Book Series

For more information see https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=483667
CERT Cybersecurity Engineering and Software Assurance Professional Certificate

Released March 2018

The program consists of five components

- Software Assurance Methods in Support of Cybersecurity Engineering
- Security Quality Requirements (SQUARE)
- Security Risk Analysis (SERA)
- Supply Chain Risk Management
- Advanced Threat Modeling

To learn more, visit
www.sei.cmu.edu/go/cybersecurity-engineering
Contact Information

Carol Woody
cwoody@cert.org

Web Resources (CERT/SEI)
www.sei.cmu.edu/go/cybersecurity-engineering