What Is Cybersecurity Engineering? And Why Do I Need It?

Carol Woody, Ph.D.
Principal Researcher

Rita Creel
Acting Deputy Director, CERT

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

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.

DM20-1061

Topics

What is Cybersecurity Engineering?

Today's Cybersecurity Context

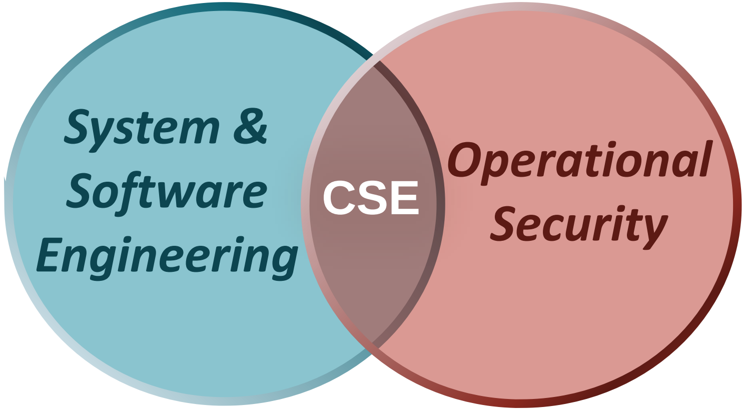
Cybersecurity Engineering Can Reduce Mission Risk

Final Thoughts

What Is Cybersecurity Engineering—and Why Do I Need It?

What Is Cybersecurity Engineering?

Cybersecurity Engineering (CSE)



What is it?

- Applying integrated principles, practices and rigor of **engineering** and **operational security** to build, configure, operate, and maintain systems and software for secure, resilient operation

Why do we need it?

- Operational security alone is insufficient in complex and highly interconnected technology environments.
- System and software security engineering requires more than security compliance checklists and tools.
- CSE applies threat-informed security risk analyses to reduce the systems and software attack surface.

Key Areas of Focus for Cybersecurity Engineering

These six activities are essential to building and fielding secure, resilient technology for today's contested environments:

1. Determining Risk
2. Defining and Monitoring System and Component Interactions
3. Evaluating Trusted Dependencies
4. Anticipating and Planning Responses to Attacks
5. Coordinating Security Throughout the Lifecycle
6. Measuring to Improve Cybersecurity

Reference: *Principles and Measurement Models for Software Assurance*, Nancy R. Mead, Dan Shoemaker (University of Detroit Mercy), Carol Woody, 2013 IJSSE Special Issue on Cybersecurity Scientific Validation, January 2013, http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=298843

1 Determining Risk

Cybersecurity engineering effectively considers threats and mission risk.

Perceptions of risk drive security engineering and assurance decisions.

- Cybersecurity expertise is essential to perceive security risk: threats, exploitable weaknesses in design and code, mission impact and likelihood of realized risk.
- Misperceptions are failures to recognize threats and impacts.
  - “How could it happen to us?” or, “It could not happen here!”

Acquirers, systems and software engineers, and developers must understand successful cyber attacks and impacts on the operational mission.

- Successful organizations learn from attacks on their organizations and on others’.
  - How can I rapidly anticipate and detect attacks?
  - How quickly can I respond and recover?
  - How can I be vigilant?

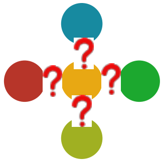
② Defining and Monitoring System and Component Interactions

Cybersecurity engineering exposes and manages risk to systems from the interaction among technology components and external systems.

Security risks must be analyzed and managed across all stakeholders and systems.

- Design interactions to avoid missing critical threats at various interaction points.
- Balance costs of security against the likely impact of a successful attack (security risk).
- Balance security risk with opportunities and needs (performance, reliability, usability, etc.).
- Consider interactions at all technology levels (network, security appliances, architecture, applications, data storage, etc.) to identify and address security control gaps.

3 Evaluating Trusted Dependencies



Cybersecurity engineering evaluates dependencies and inherited risk to ensure the appropriate level of trust is established.

Security dependencies relate to components others build that you use or connect with (inherited risk): What is a reasonable level of trust?

- Each dependency represents a risk (e.g. reuse, open source, collaboration environments).
- Dependency and trust decisions should be based on a realistic assessment of the risks and opportunities inherent in the dependency.
- Dependencies are not static so trust decisions must be re-evaluated regularly to identify changes that warrant reconsideration.
- Using shared components to build technology applications and infrastructure increases the dependency on and potential mismatch of other's decisions (supply chain risk).

④ Anticipating and Planning Responses to Attacks

Cybersecurity engineering identifies and plans for potential attacks that could interfere with mission critical functions. There are no perfect protections against attacks.

A growing and diverse population of adversaries uses both simple and increasingly sophisticated capabilities to compromise the confidentiality, integrity, and/or availability of technology assets.

- Adversaries often use a combination of technology, processes, standards, methods, and practices to craft a successful attack (socio-technical responses).
- Attacks are designed to take advantage of the ways we normally use technology or to contrive exceptional situations where defenses are circumvented.
- Attackers' profiles, capabilities, and methods are continually evolving.

⑤ Coordinating Security Throughout the Lifecycle

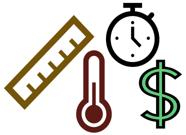
Cybersecurity engineering coordinates and integrates security across all aspects of acquisition and development throughout the lifecycle.

Assuring the security and resiliency of a system's mission-critical functions requires

- Planning for what might go wrong: developing requirements for misuse/abuse cases with corresponding system and software requirements for secure, resilient response and operation.
- Building security and resiliency into the system.
- Verifying that expected security and resiliency characteristics were actually built in.

Authority and responsibility for coordination must be clearly established at an appropriate level in the organization to ensure effective participation and coverage.

⑥ Measuring to Improve Cybersecurity



Cybersecurity engineering identifies measurement data and analysis procedures necessary to determine whether levels of cybersecurity assurance and risk are acceptable and to identify opportunities for improvement.

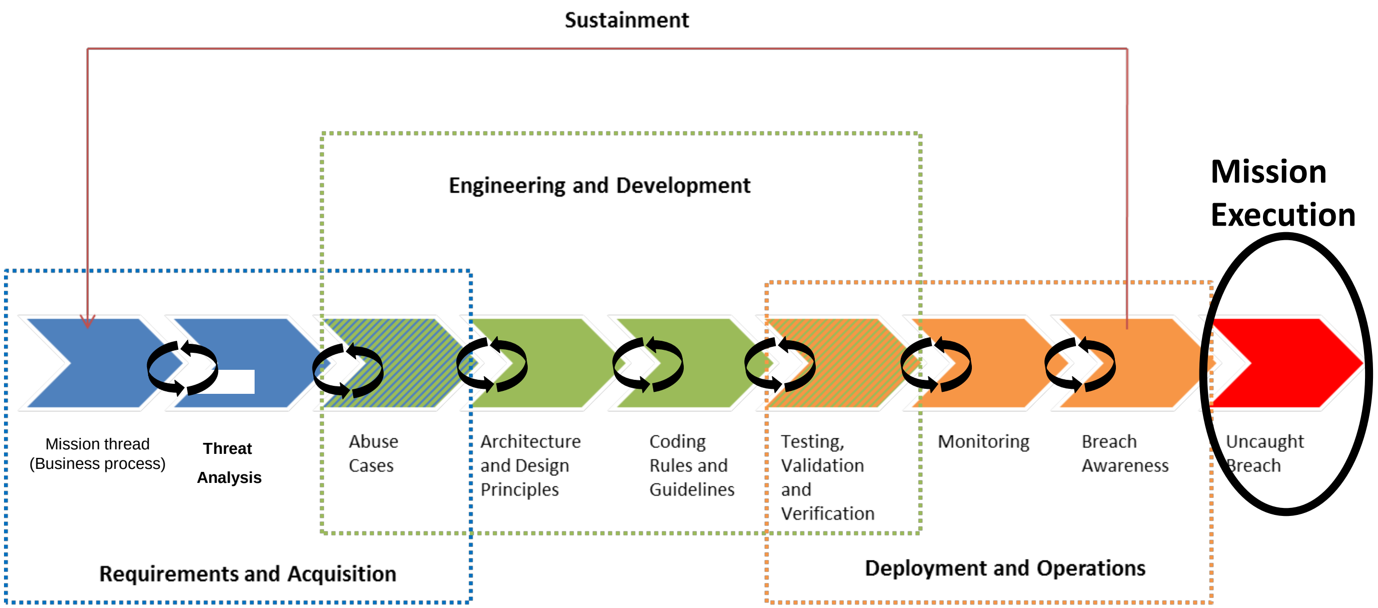
Cybersecurity attributes of critical products, processes, and resources are identified, measured, and monitored throughout the lifecycle.

- Use risk assessments that consider both current and potential future threats, vulnerabilities, and impacts to identify critical attributes to be measured and monitored.
- Develop and consistently apply well-formed measurement definitions and procedures to establish credibility of the measurement and analysis results.
- Include all elements of the socio-technical environment that touch engineering and acquisition activities (processes, procedures, products, resources, etc.).
- Support measurement with robust engineering planning: define a security measurement plan that spans the lifecycle, and develop requirements for any needed instrumentation.

What Is Cybersecurity Engineering—and Why Do I Need It?

Today’s Cybersecurity Context

Cybersecurity Is a Lifecycle Challenge

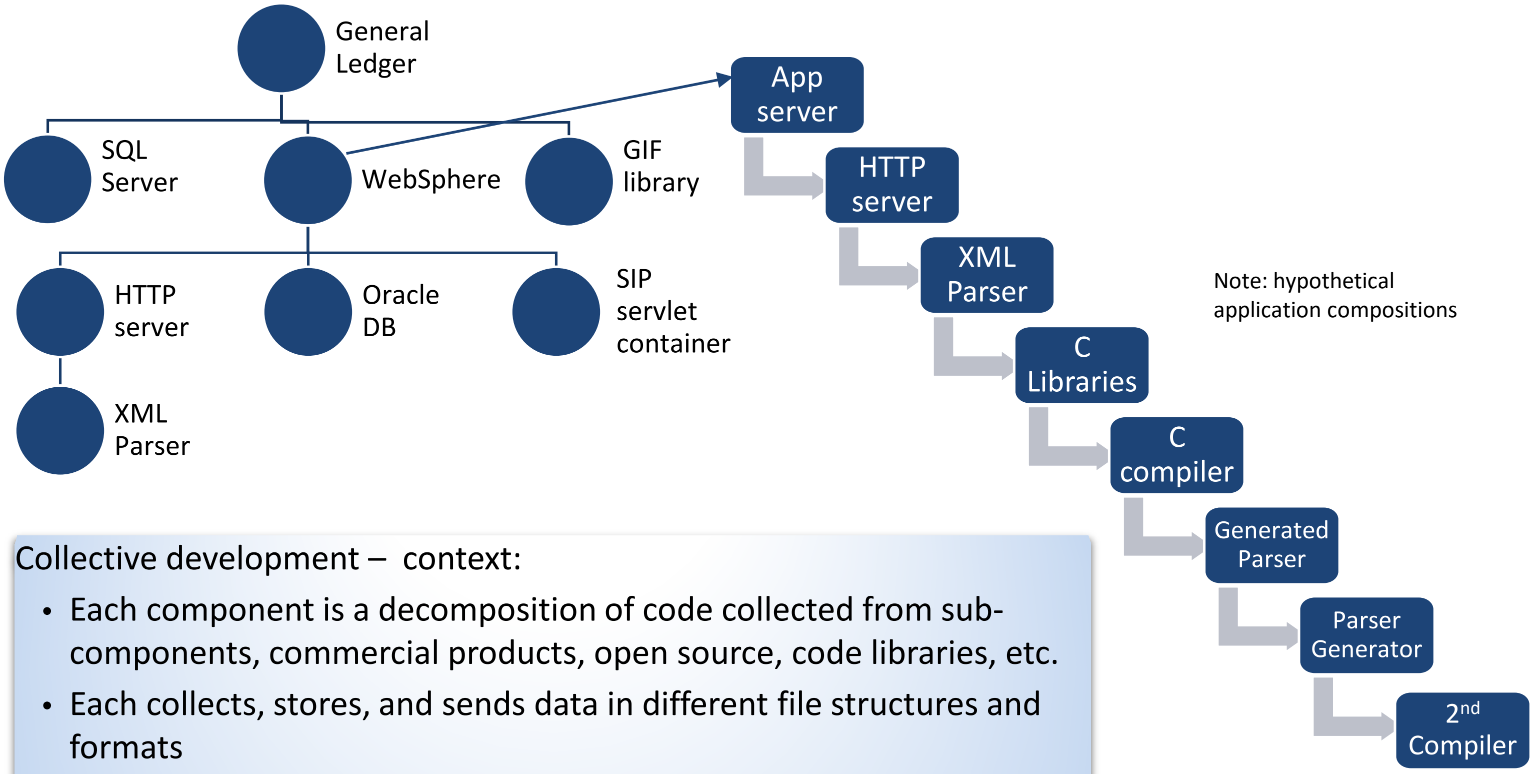


Design Weaknesses

Coding Weaknesses

Implementation Weaknesses

Software Development is Now Module Assembly



Note: hypothetical application compositions

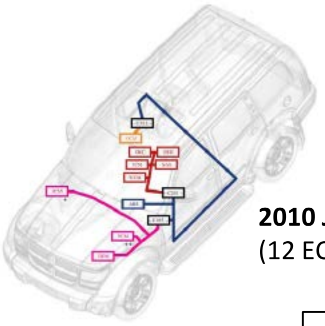
Collective development – context:

- Each component is a decomposition of code collected from sub-components, commercial products, open source, code libraries, etc.
- Each collects, stores, and sends data in different file structures and formats
- No one person, team, or organizations knows how all the pieces work

Reuse is rampant!

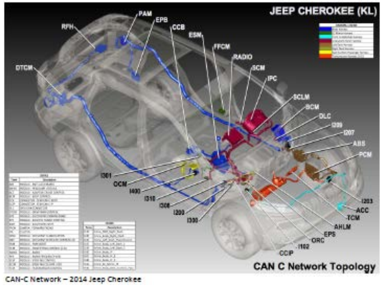
Modularity is Emphasized: Assemble from 3rd party components to reduce construction cost/schedule and increase flexibility

Example:
Vehicles are now Assembled from Engine Control Units (ECUs)



2010 Jeep Cherokee
(12 ECUs)

2014 Jeep Cherokee
(32 ECUs)



Expect, and engineer for, increases in supply chain risk

ECUs are prefabricated, software-driven components addressing select functionality and tailorable to a specific domain.

Modern high-end automotive vehicles have software and connectivity:

- Over 100 million lines of code
- Over 50 antennas
- Over 100 ECUs

Sources: Miller and Valasek, A Survey of Remote Automotive Attack Surfaces, http://illmatics.com/remote%20attack%20surfaces.pdf; https://www.cst.com/webinar14-10-23~?utm_source=rfg&utm_medium=web&utm_content=mobile&utm_campaign=2014series
https://en.wikipedia.org/wiki/Electronic_control_unit

Anyone Can and is Encouraged to Write Software . . .

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

http://lifehacker.com/how-and-why-to-teach-your-kids-to-code-510588878

From 1990 to 2012, software industry production grew from $149B to $425B: Business investments in software accounted for 12.2% of all fixed investment, compared to 3.5% for computers and peripherals.

. . . but *engineering* is required to build software to address a critical mission. Cybersecurity engineering aims to minimize attacker impact on the mission.

Software is Everywhere

You think you're building (or buying, or using) a product such as:

car or truck	satellite	mobile phone	development tools
home security system	aircraft	pacemaker	security tools
home appliance	financial system	bullets for a gun	

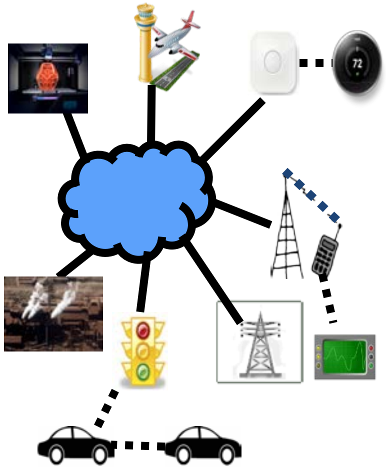
Actually you're getting ***a software platform:***

- Software is a part of almost everything we use.
- Software defines and delivers component and system communication.
- Software is used to build, analyze and secure software.

All software has defects:

- Best-in-class code has <600 defects per million lines of code (MLOC).
- Good code has around 1000 defects per MLOC.
- Average code has around 6000 defects per MLOC.

(based on Capers Jones research http://www.namcook.com/Working-srm-Examples.html)



The Attacker Needs Three Ingredients

Exploitable vulnerabilities

- Millions of lines of software code contain defects; up to 5% are vulnerabilities

  ref: Woody, Carol et al. *Predicting Software Assurance Using Quality and Reliability Measures.* http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=428589)

- Hundreds of thousands of known software vulnerabilities exist

  ref: NIST National Vulnerability Database, https://nvd.nist.gov/general/nvd-dashboard.

Access

- Increased connectivity links systems to other systems and connects new types of devices (IoT), which may be inadequately protected.
- Increased system and device connectivity with trusted connections provide security gaps that may be compromised.

Ability to exploit

- Attackers have access to software development tools and techniques as well as libraries of successful exploit software
- Attackers can apply reverse engineering to commercial and open source software to discover weaknesses.

Chasing Software Flaws is a Chronic Activity

The National Institute of Standards and Technology (NIST) National Vulnerability Database (NVD) contains **152,766 known vulnerabilities** – NVD received **15,911 new vulnerabilities in 2020** (as of 11/9).

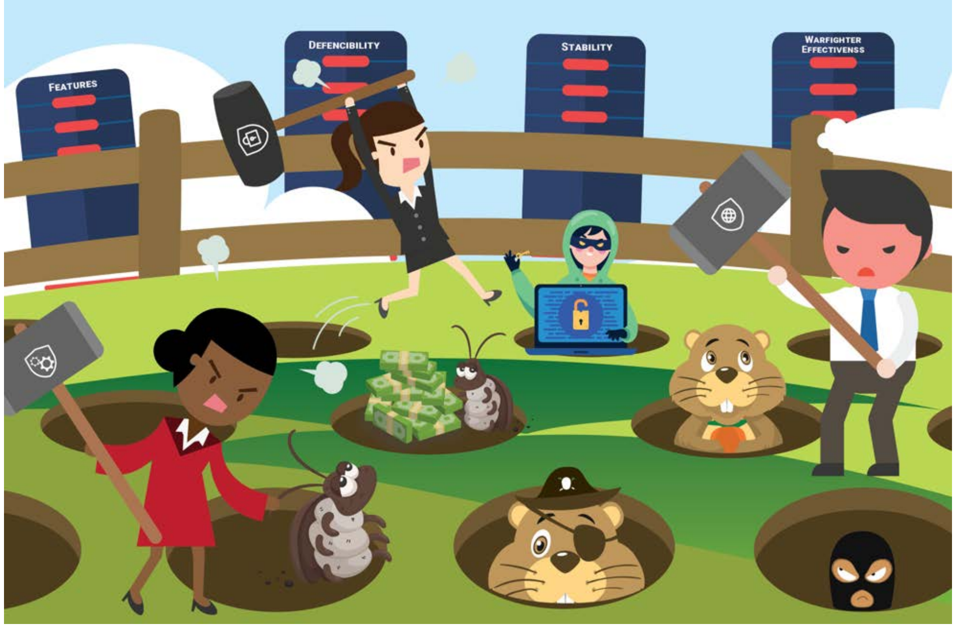
Just a few items from **SANS NewsBites** (published Tuesdays & Fridays) and **SANS @Risk** (published Thursdays) https://www.sans.org/newsletters/ (a few of hundreds from 14 August through 12 November 2020)

⋮

- Microsoft Patch Tuesday updates address at least 120 vulnerabilities in Windows and other products and services, including two actively exploited vulnerabilities
- "BLESA" flaw in Bluetooth devices could allow attackers to spoof connections to mobile and internet-of-things devices.
- Universal Health Services (UHS) Ransomware Attack Affects All 400 U.S. Health Systems
- Improperly Configured AWS S3 Bucket Exposes 10 Million Hotel Guest Records
- University of Vermont (UVM) Health Network Cyberattack Impacts Chemotherapy, Mammograms
- Google Drive Collaboration Feature is Being Exploited by Bad Actors
- Apple iOS memory corruption vulnerability may lead to arbitrary code execution
- Oracle WebLogic Server Unauthenticated Remote Code Execution Vulnerability

⋮

Today, Operations Plays Whack-a-mole Chasing Attacks



Rapid delivery of features is prioritized over defensibility, reliability, and stability.

Operational missions are jeopardized by weak designs that allow attackers to leverage the many vulnerabilities.

Once software's in an operational system, vulnerabilities can be difficult (or impossible) to mitigate.

What Is Cybersecurity Engineering—and Why Do I Need It?

Cybersecurity Engineering Can Reduce Mission Risk

Cybersecurity Begins with Good Engineering!

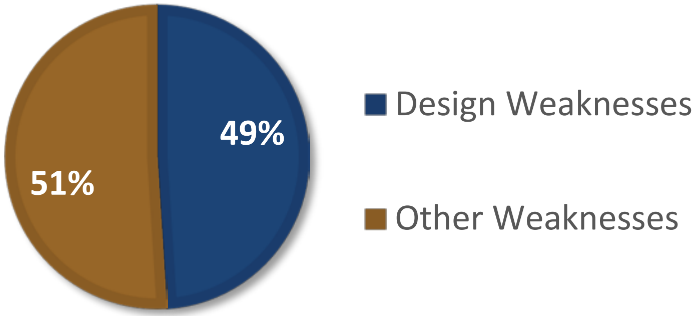
- A Cybersecurity by Design
- B Security Requirements: An Essential Foundation
- C Security Engineering Risk Analysis: Addressing Requirements Gaps
- D Integration of Cybersecurity Risks With Program Risk Management
- E Measurement for Cybersecurity Improvement
- F Lifecycle Practices for Software and System Cybersecurity

Ⓐ Cybersecurity By Design

Design weaknesses are a significant contributor to software vulnerabilities.

Cybersecurity Engineering can help identify and remove design weaknesses before they are implemented in software and become vulnerabilities.

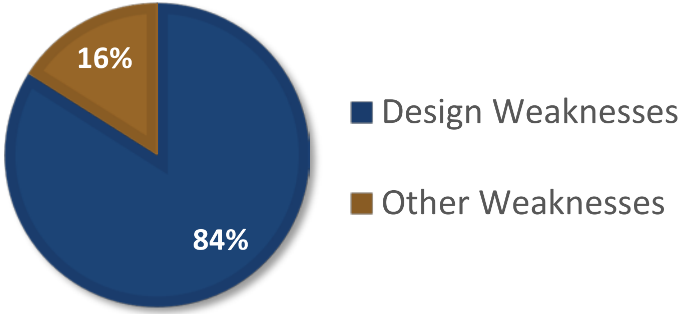
COMMON WEAKNESS ENUMERATION



Almost half the 891 weaknesses in the Common Weakness Enumeration, ***are design weaknesses***[1]

[1]Source: https://cwe.mitre.org/data/definitions/701.html
as of Nov 9, 2020

2020 TOP 25 MOST DANGEROUS SOFTWARE WEAKNESSES



The 21 design weaknesses in the Top 25 account for ***over 16,000 entries in the National Vulnerability Database (NVD)***[2].

[2]Source: https://cwe.mitre.org/top25/archive/2020/2020_cwe_top25.html
as of Nov 9, 2020

Ⓑ Security Requirements: An Essential Foundation

Design weaknesses and software vulnerabilities frequently result from missing or insufficient cybersecurity requirements.

Typical problems

- Minimized or ignored in requirements elicitation because stakeholders
  - have a limited understanding of the impact of cybersecurity risk on mission success, and
  - are unable to state their cybersecurity requirements
- Reliance on compliance mandates (e.g. Risk Management Framework (RMF)) and standards (e.g. ISO/IEC 27001) in place of mission-focused cybersecurity risk analysis
- Narrow focus only on common security practices (e.g. authentication & authorization) or mechanisms (e.g. Secure Socket Layer (SSL) for Web communication)

Good requirements, architecture and design activities and effective trade-off decisions that include cybersecurity are critical to operational mission success

Ⓒ Security Engineering Risk Analysis (SERA): Addressing Requirements Gaps

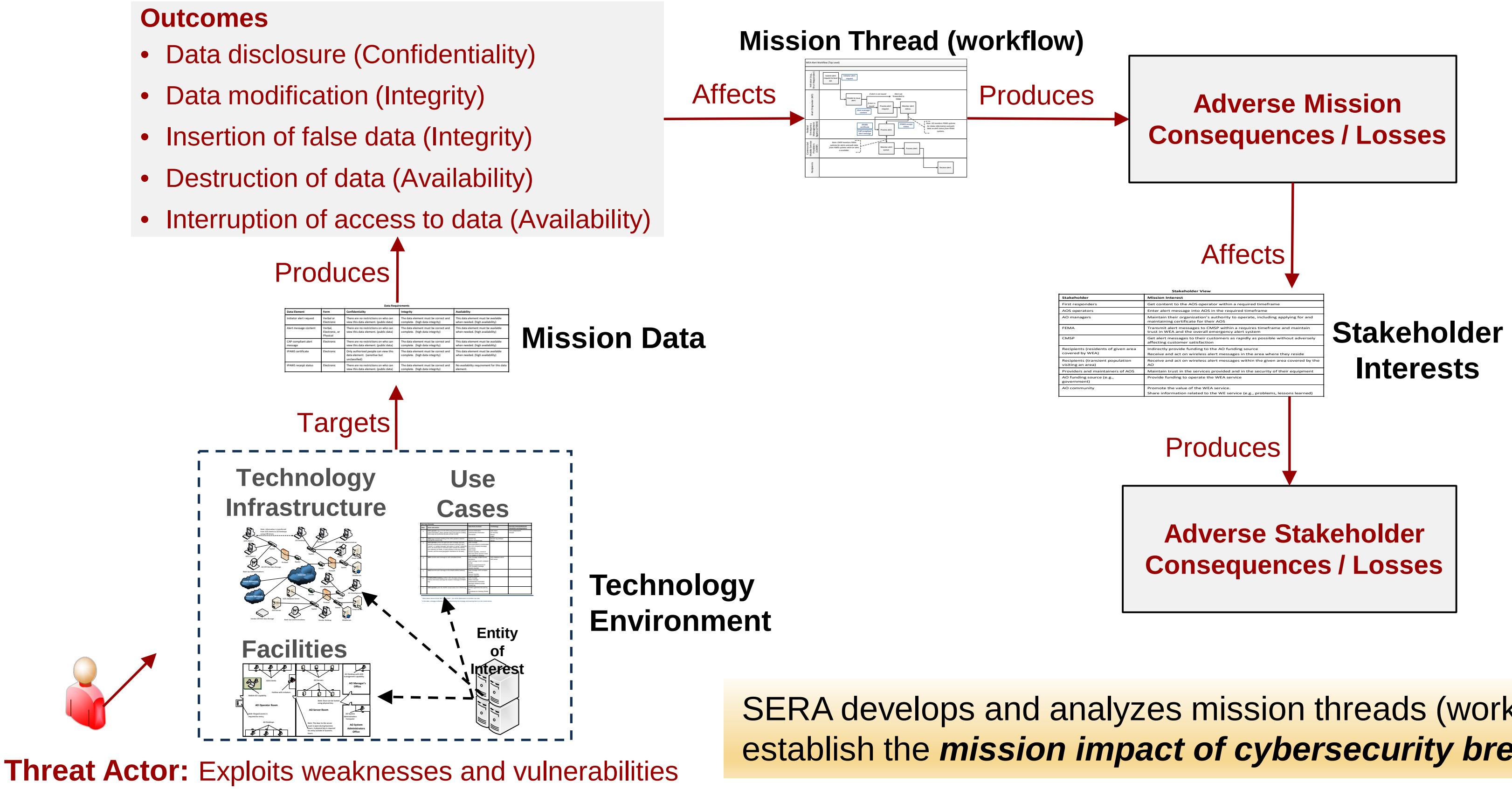
What is SERA?

- Systematic method for analyzing complex cybersecurity risks in software-reliant systems and systems of systems
- Considers *intended operational aspects* of a system to identify missing requirements and key design and architecture trade-offs with unacceptable mission impacts

Value of SERA

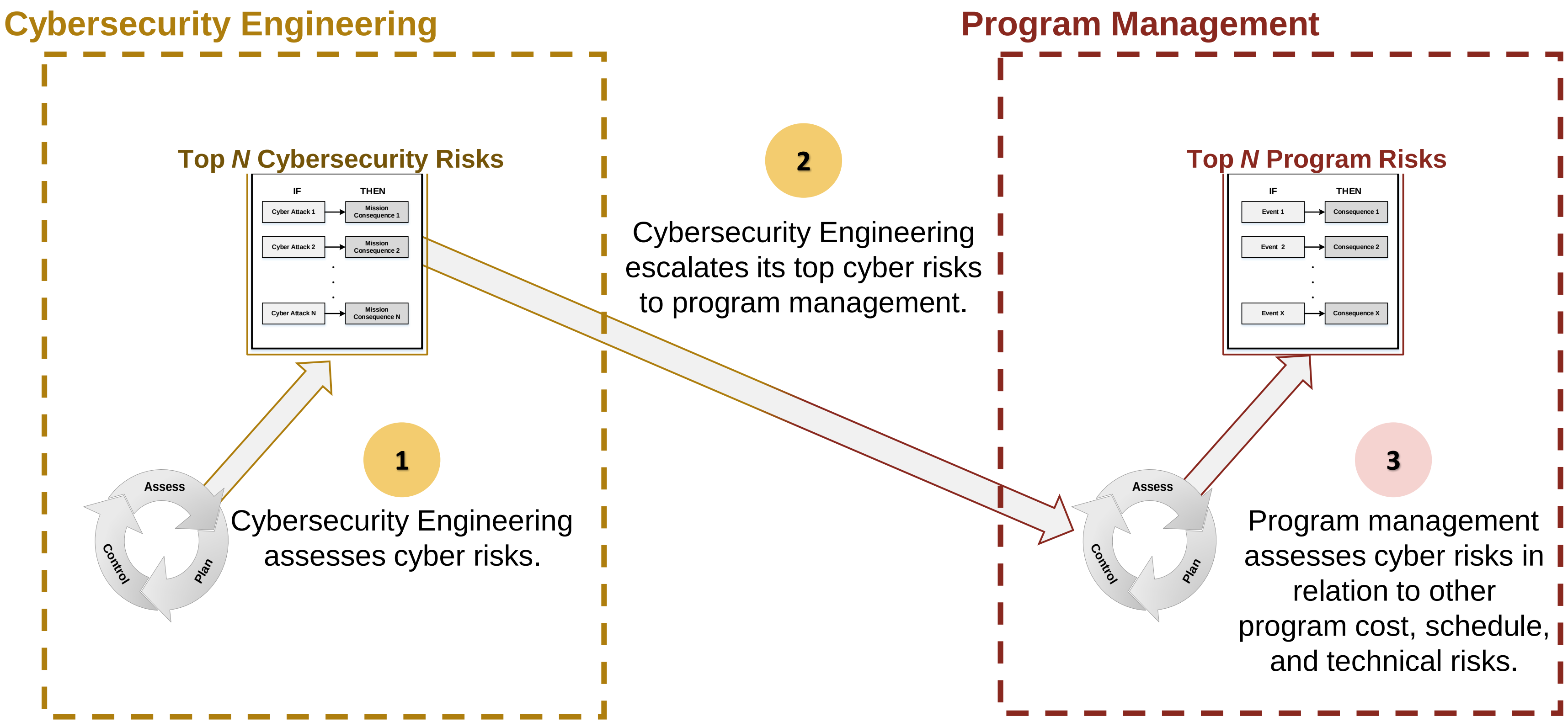
- Identifies cybersecurity weaknesses that, when addressed, improve the ability of a system to support mission success
- Assembles a shared operational view (mission and technical perspectives) to connect cybersecurity threats with mission risk

ⓒ SERA: Requirements for Mission Cybersecurity Risk



SERA develops and analyzes mission threads (workflows) to establish the ***mission impact of cybersecurity breaches***.

Ⓓ Integration of Cybersecurity Risks with Program Risk Management



Ⓔ Measurement for Cybersecurity Improvement

Cybersecurity Engineering needs to establish an assurance goal and identify the measures that will be useful in reaching the goal[1]

- Decompose the goal into sub goals that establish a practice, outcomes and possible metrics (use Goal/Question/Metric[2])
- To begin assembling data for evaluation, identify a subset of possible metrics that
  - address the goal, and
  - can be produced using data that are already available, or can be collected with minimal additional effort
- Add/adjust metrics building on what you are already doing

[1]Woody, Carol; Ellison, Robert; & Ryan, Charles. *Exploring the Use of Metrics for Software Assurance.* CMU/SEI-2018-TN-004. Software Engineering Institute, Carnegie Mellon University. 2019. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=540881
[2]Basili, V.R. Software Modeling and Measurement: The Goal/Question/Metric Paradigm. https://www.cs.umd.edu/~basili/publications/technical/T78.pdf

Ⓕ Lifecycle Practices for Software and System Cybersecurity

Requirements. Does the program/project define and manage security requirements?

Architecture and Design. Does the program/project appropriately address security in its system and software architecture and design?

Implementation. Does the program/project minimize the number of weaknesses inserted into the operational software?

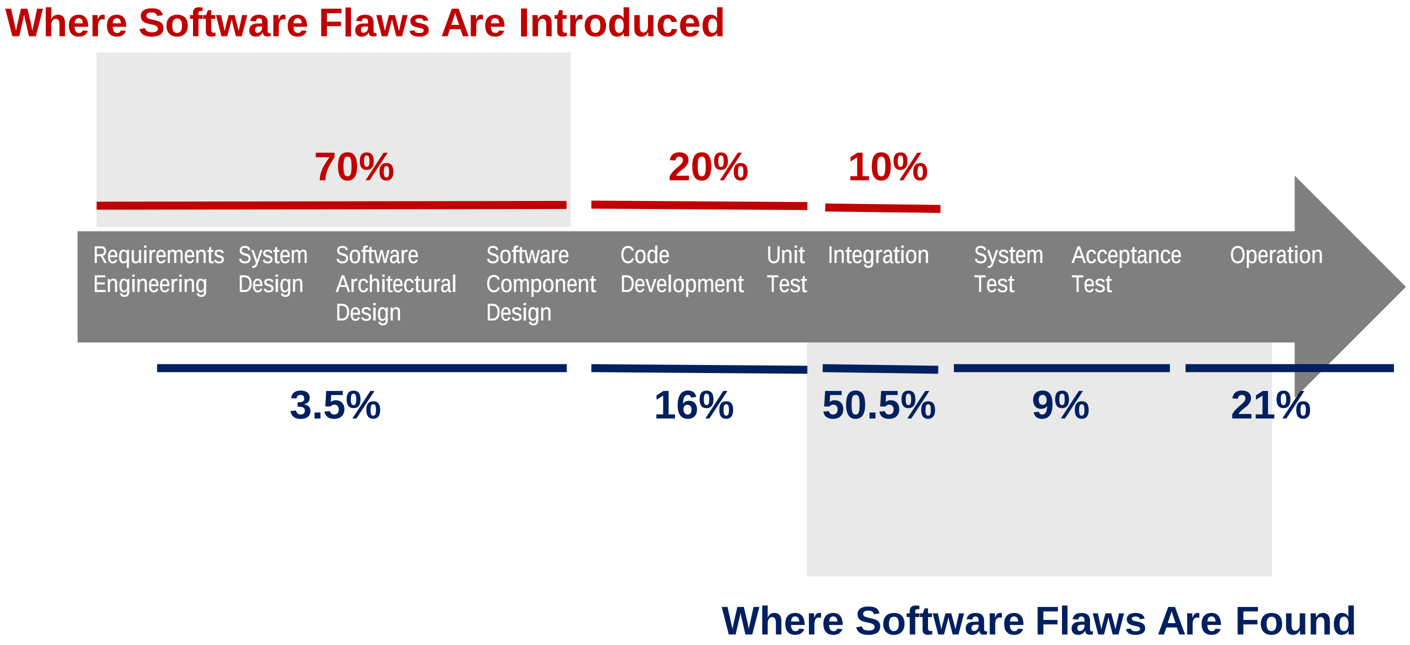
Testing, Validation, and Verification. Does the program/project test, validate, and verify security in its software and system components?

Support Tools and Documentation. Does the program/project develop tools and documentation to support secure configuration and operation of components?

Deployment. Does the program/project consider security during the deployment and maintenance of software and system components?

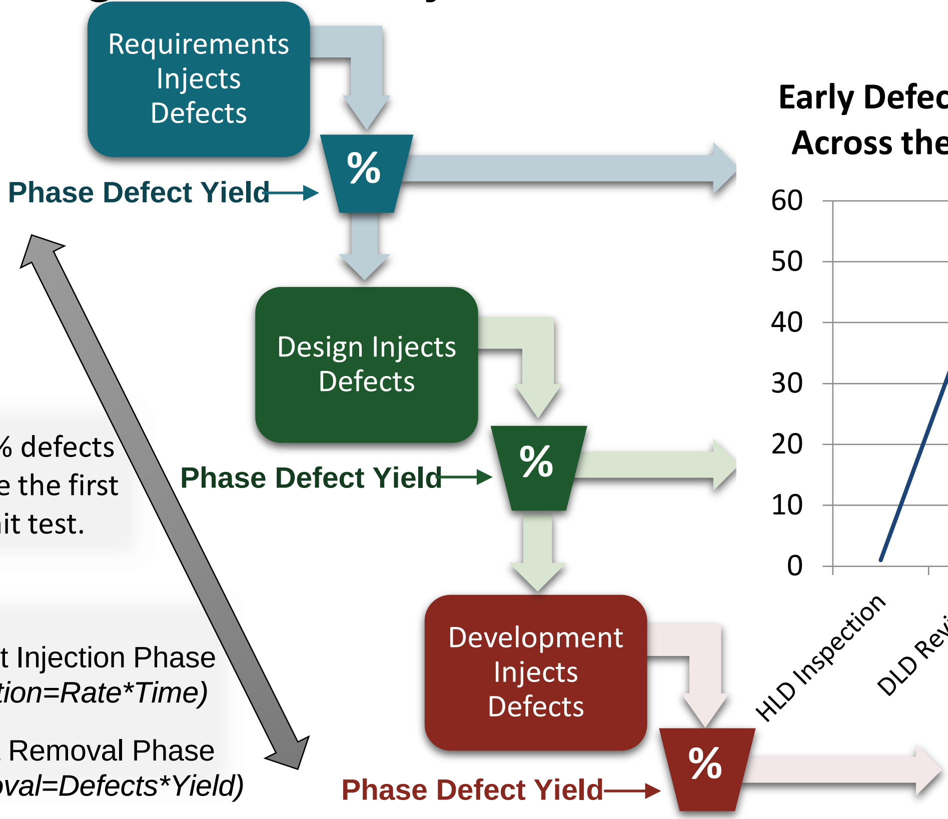
Alberts, Christopher; & Woody, Carol. *Prototype Software Assurance Framework (SAF): Introduction and Overview.* CMU/SEI-2017-TN-001. Software Engineering Institute, Carnegie Mellon University. 2017. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=496134

Ⓕ Most Software Flaws Are Found Long After They Are Introduced



Sources: *Critical Code*; NIST, NASA, INCOSE, and Aircraft Industry Studies
**Woody et al.* http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=428589)

Ⓕ Manage Defect Injection and Removal for Early Detection

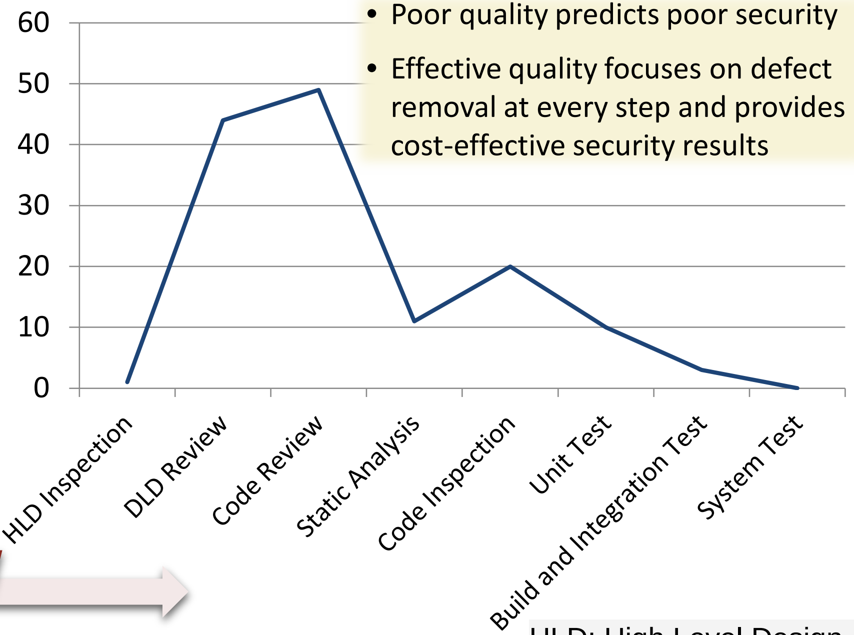


Process yield: % defects removed before the first compile and unit test.

Defect Injection Phase
*(Injection=Rate*Time)*

Defect Removal Phase
*(Removal=Defects*Yield)*

Early Defect Removal Across the Lifecycle



- Poor quality predicts poor security
- Effective quality focuses on defect removal at every step and provides cost-effective security results

HLD: High Level Design
DLD: Detailed Level Design

What Is Cybersecurity Engineering—and Why Do I Need It?

Final Thoughts

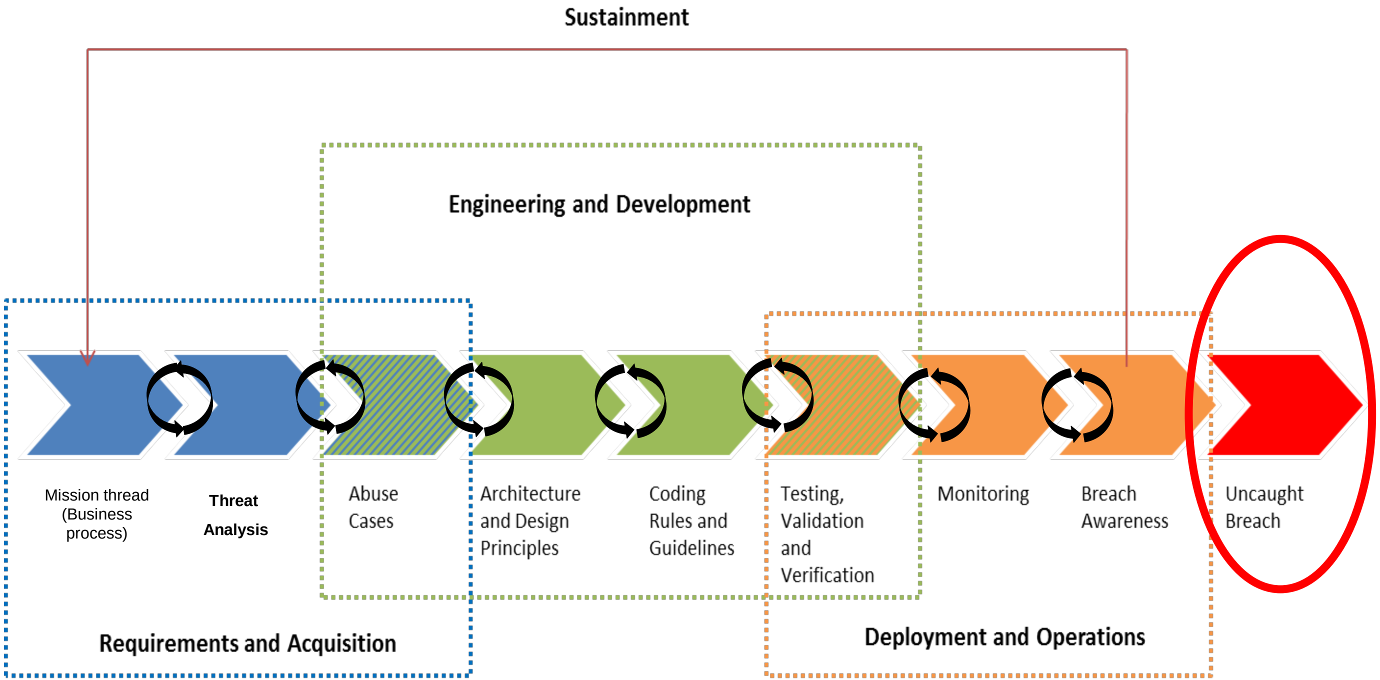
Build a Cybersecurity Strategy

Establish a plan for sufficient system and software cybersecurity engineering to ensure the operational mission(s) continue, even under cyber attack.

Elements in the strategy include consideration of:

- Appropriate security requirements to ensure confidentiality, integrity, availability (CIA)
- Ongoing monitoring of CIA in operational systems and software
- Sufficient resiliency to recognize, resist, and recover from attacks
- Operational security under all circumstances including designed in methods of denying critical information to an adversary to avoid/minimize mission impact
- Evaluate of alternatives to determine the level of accepted cybersecurity risk
- Appropriate lifecycle processes and practices to reduce operational vulnerabilities

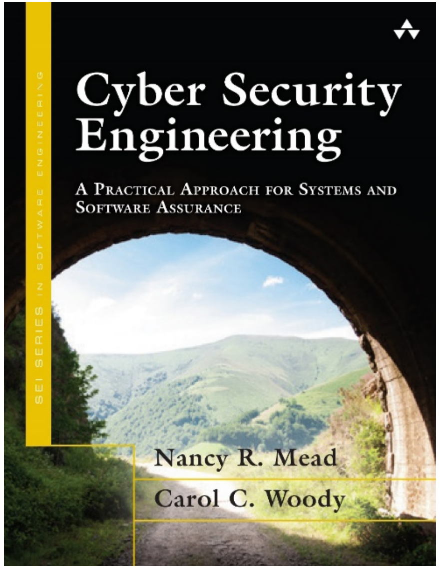
Continuous Focus on Cybersecurity Risk Across the Lifecycle is Critical to Operational Mission Success



Opportunities to Learn More

Textbook

Cybersecurity Engineering



SEI Book Series

Professional Certificate

CERT Cybersecurity Engineering and Software Assurance

https://sei.cmu.edu/education-outreach/credentials/credential.cfm?customel_datapageid_14047=33881

Online training in 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

Contact Information

Carol Woody, Ph.D.

cwoody@cert.org

Rita Creel

rc@cert.org

Web Resources

Building security into application lifecycles

https://sei.cmu.edu/research-capabilities/all-work/display.cfm?customel_datapageid_4050=48574

CMU SEI Home Page

https://sei.cmu.edu/