Engineering System Security for a System of Systems

Carol Woody, Ph.D.
Technical Manager,
Cybersecurity Engineering

Software Engineering Institute (SEI)
Carnegie Mellon University (CMU)
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.

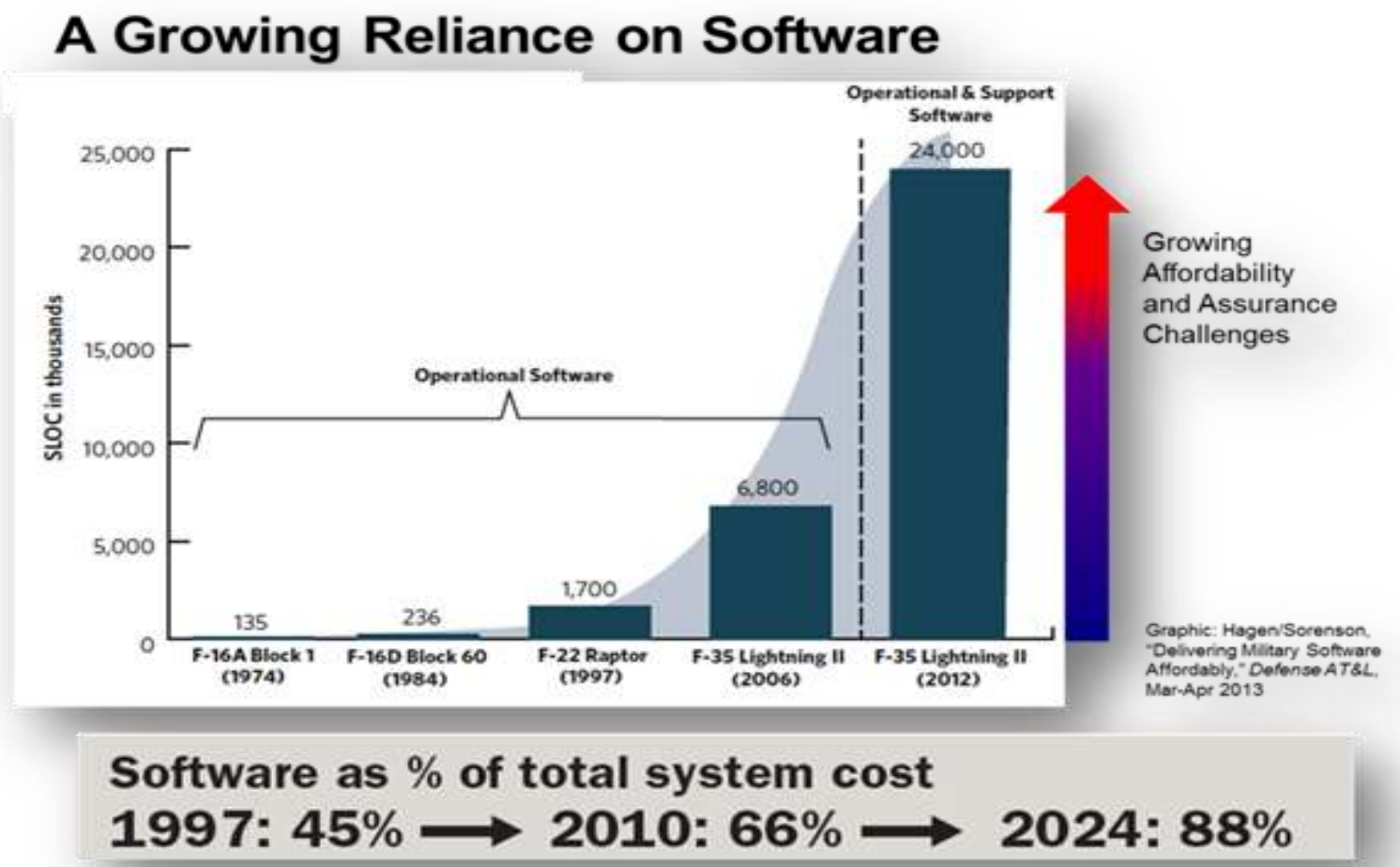
DM19-1279

Agenda

- Software in a System of Systems
- Security Challenges in a System of Systems
- Software Quality Improves System Security
- Security Engineering Risk Analysis (SERA) for a System of Systems
- Engineering Security into the System of Systems

Software in a System of Systems Context

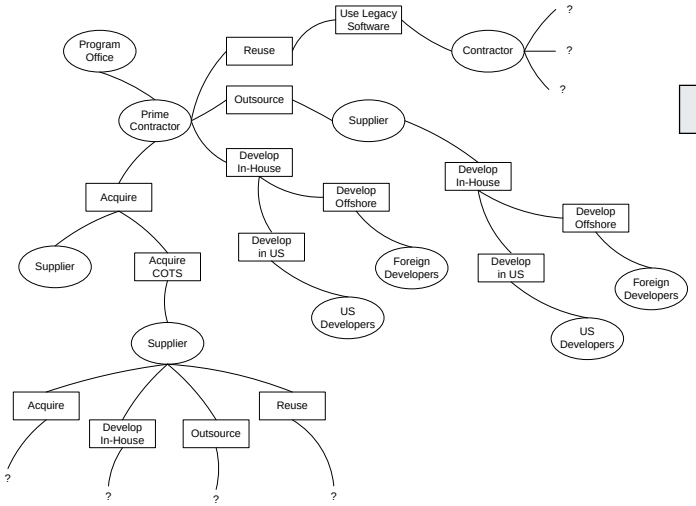
Software Reliance is Rapidly Expanding

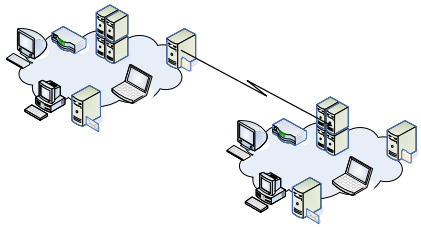


Source: U.S. Air Force Scientific Advisory Board. *Sustaining Air Force Aging Aircraft into the 21st Century* (SAB-TR-11-01). U.S. Air Force, 2011.

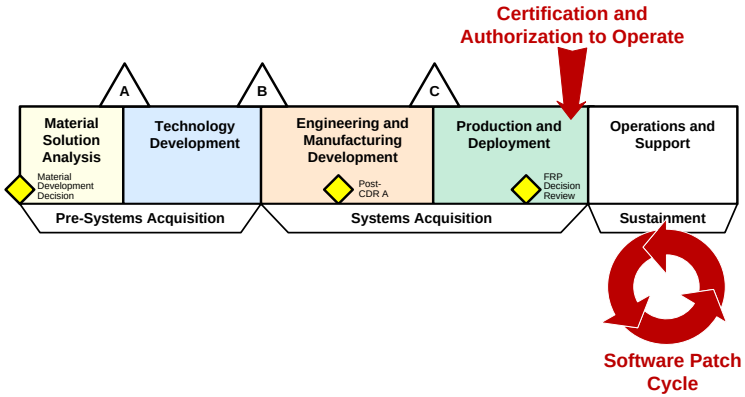
Systems are Built Independently by Many Hands

Software Supply Chain



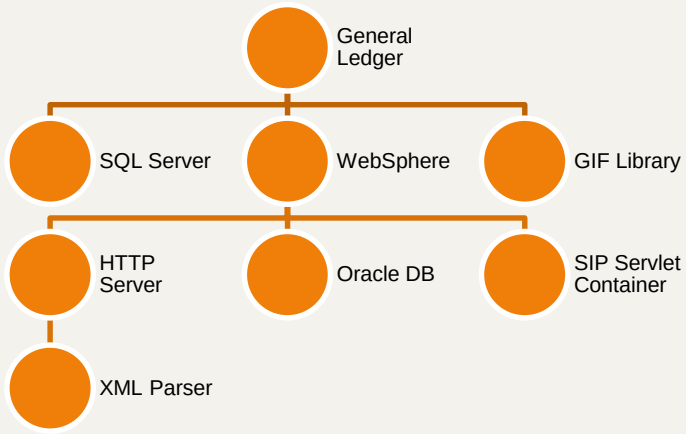


Integrated Software-Reliant System



Visibility and direct controls are limited to the top of the supply chain

Software Development is Primarily Assembly



Note: hypothetical application composition

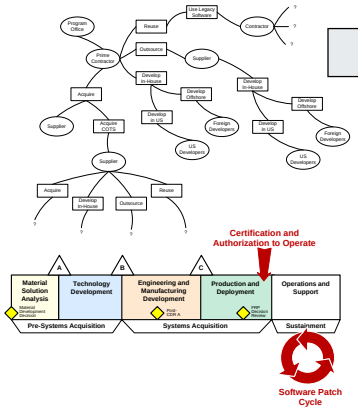
Process now involves assembly (reuse) using collective development. Each product has

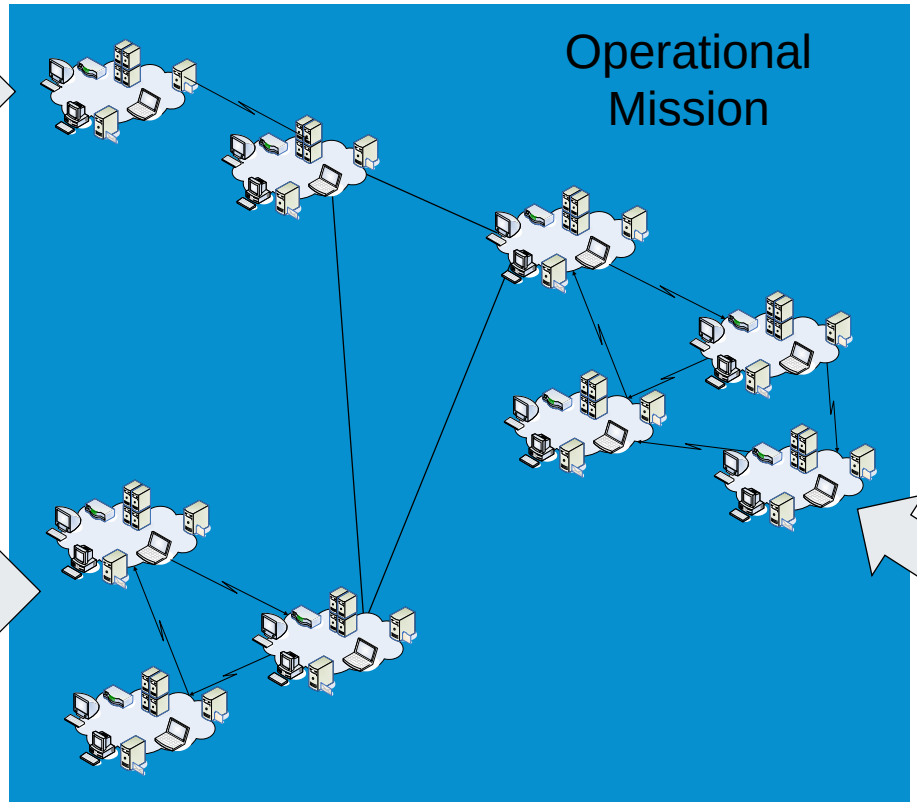
- too costly and time consuming for a single organization to build all pieces

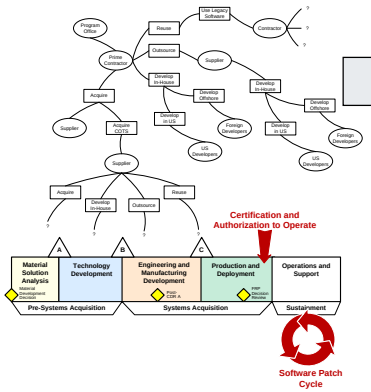
Each component is designed, built and maintained independently

Operational Missions Rely on Systems of Systems

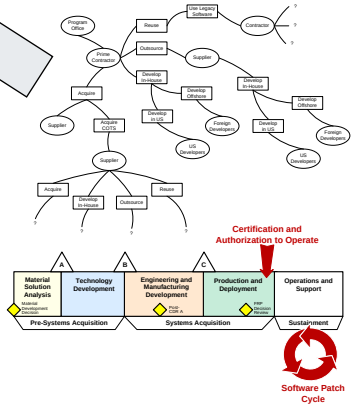
Acquisition 1





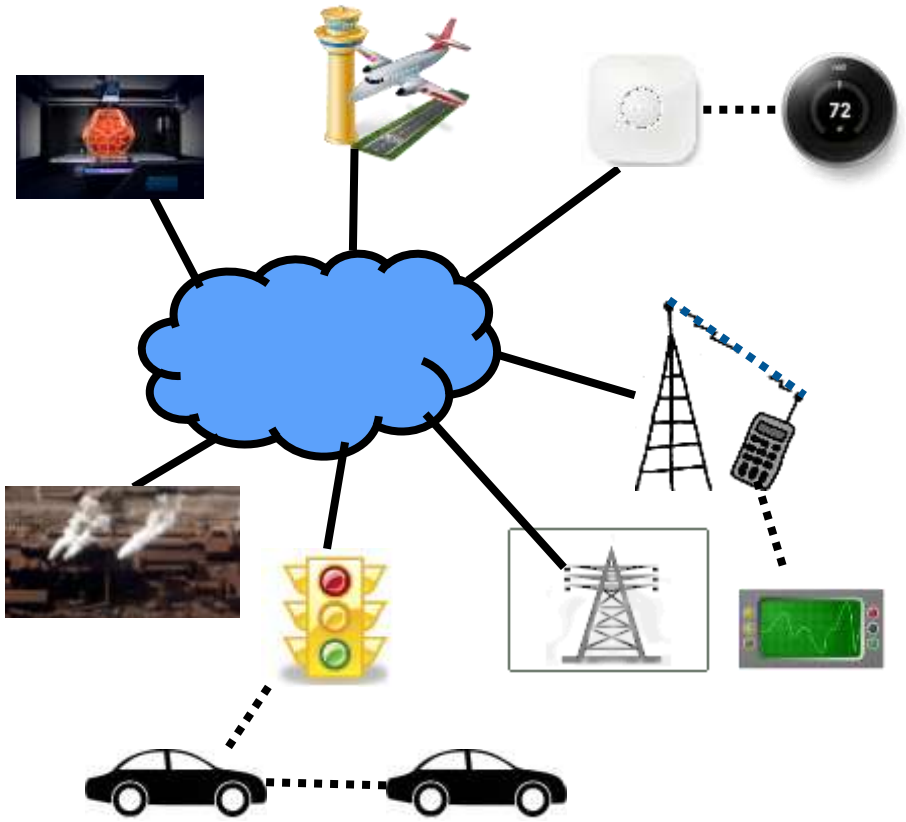


Acquisition 2



Acquisition 3

Software-Reliant Products are Connected using Software Driven Communications



Internet of Things (IoT)

- Cellular
  - Main processor
  - Graphics processor
  - Base band processor (SDR)
  - Secure element (SIM)
- Automotive
  - Autonomous vehicles
  - Vehicle to infrastructure (V2I)
  - Vehicle to vehicle (V2V)
- Industrial and home automation
  - 3D printing (additive manufacturing)
  - Autonomous robots
  - Interconnected SCADA
- Aviation
  - Next Gen air traffic control
  - Fly by wire
- Smart grid
  - Smart electric meters
  - Smart metering infrastructure
- Embedded medical devices

Anyone Can 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 1997 to 2012, software industry production grew from $149 billion to $425 billion

From 1990 to 2012, business investments in software grew at more than twice the rate of all fixed business investments; and from 2010 to 2012, software accounted for 12.2 percent of all fixed investment, compared to 3.5 percent for computers and peripherals

How do you make sure the code in your system is good?
Can will we compose mission reliable results with this software

Consumers Expect Cheap (often free) Easy-to-use Storage, Connectivity and Software

- Email, Google Docs, FaceBook, LinkedIn are essentially free
- Technology is a consumer commodity – we own multiple phones, multiple computers, multiple TVs, multiple cars
- Ap stores deliver code developed by anyone anywhere (many are free)
- Wi-fi hot spots are widely available and often free (airports, planes, trains, malls, restaurants, etc.)
- Open source software libraries provide developers with free tools and code for reuse

Vendors Aim for Technology Lock-in

- Vendor products are not designed to be interchangeable

  e.g. Cloud vendors provide very flexible and cheap data loading options and charge extensively for data extracting

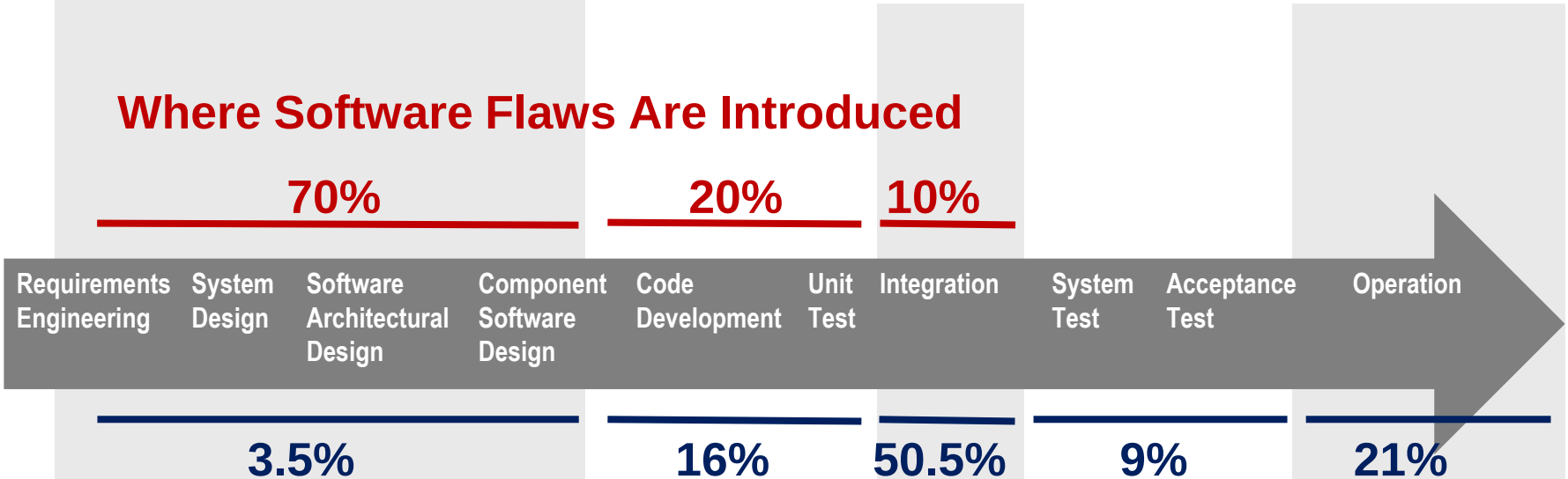
- Tools and features are built for a specific platform (e.g. iPhone or Android)

Vendors are not motivated to deliver quality

Speed is needed to capture the market share (lock in) and fix problems later (maybe)

Security Challenges in a System of Systems Context

Measuring the Growing Defects in Software



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

Sources: *Critical Code*; NIST, NASA, INCOSE, and Aircraft Industry Studies

Estimating Software Vulnerabilities

The **F-22** has 1.7 MLOC

- 1,020–10,200 defects (best – avg.)
- 51–510 vulnerabilities

The **F-35 Lightning II** has 24 MLOC

- 14,400–144,000 defects (best – avg.)
- 720–7,200 vulnerabilities

As software usage increases defects in each system will grow and the expected number of vulnerabilities will also increase

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.

5 % of defects are vulnerabilities.
Woody, Carol; Ellison, Robert; and Nichols, William. Predicting Software Assurance Using Quality and Reliability Measures. CMU/SEI-2014-TN-026. Software Engineering Institute, Carnegie Mellon University. 2014. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=428589)

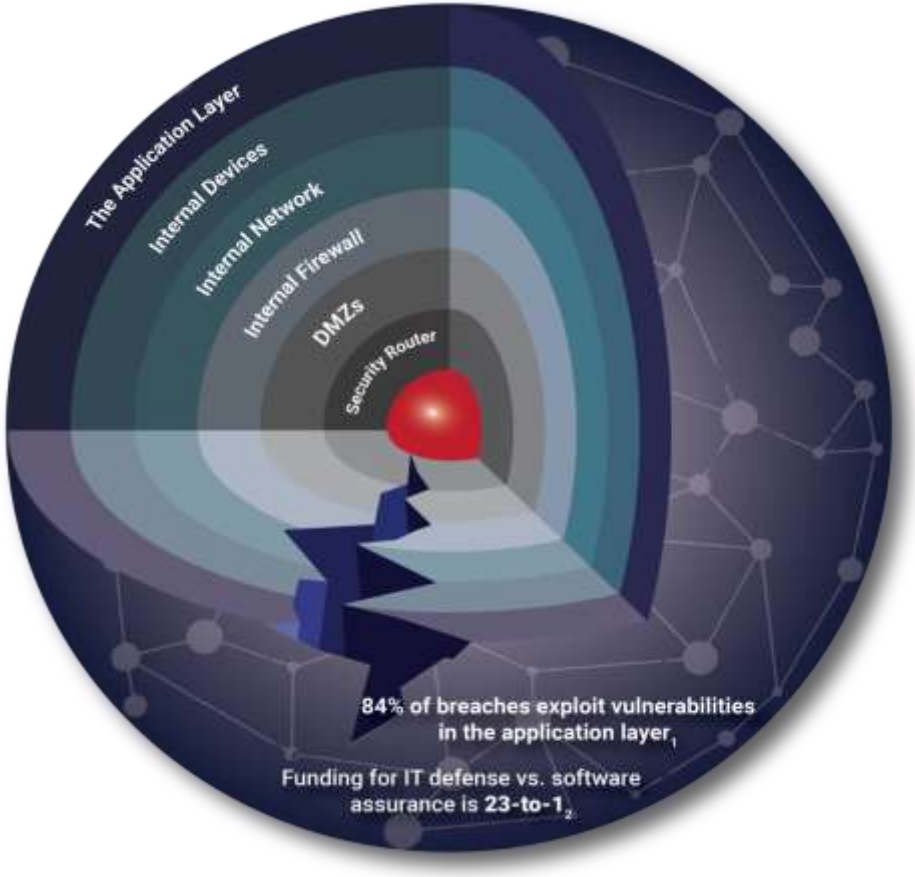
Software Vulnerabilities are Increasing

Software vulnerability is a weakness which allows an attacker to bypass security controls

The attacker needs 3 elements to make use of a weakness:

- Availability of a weakness
  - Millions of lines of software code, which contains defects, 5% are potential vulnerabilities
  - Thousands of known software vulnerabilities (NIST National Vulnerability Database)
- Attacker access to the weakness
  - Increased connectivity linking systems to other systems and connecting to new types of devices (IoT)
  - Increased system and device remote communication capability
- Attacker capability to exploit the weakness
  - Attackers have access to the same tools and techniques used to build software
  - Reverse engineering can be applied to commercial and open source software to discover weaknesses

84% of Attackers Exploit the Software Applications



"76% of U.S. developers use no secure application program process"[4]

"More than 40% of software developers globally say that security isn't a top priority for them"[4]

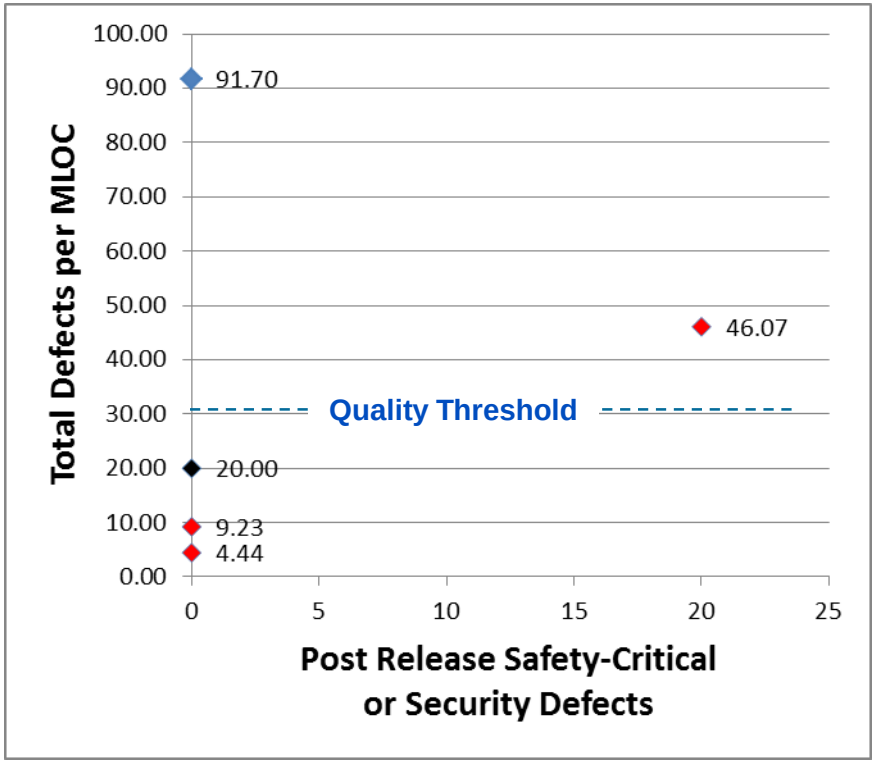
1. Clark, Tim, *Most cyber Attacks Occur from this Common Vulnerability,* Forbes. 03-10-2015
2. Feiman, Joseph, *Maverick Research: Stop Protecting Your Apps; It's Time for Apps to Protect Themselves,* Gartner. 09-25-2014. G00269825
3. *Horvath, Mark, Neil MacDonald, Ayal Tirosh: Integrating Security Into the DevSecOps Toolchain, Gartner. 11-16-2017. G00334264*
4. Microsoft[1]– http://visualstudiomagazine.com/articles/2013/07/16/majority-of-us-devs-dont-practice-secure-coding.aspx

Software Quality Improves System Security

Data Shows Increased Quality can Reduce Security Risk

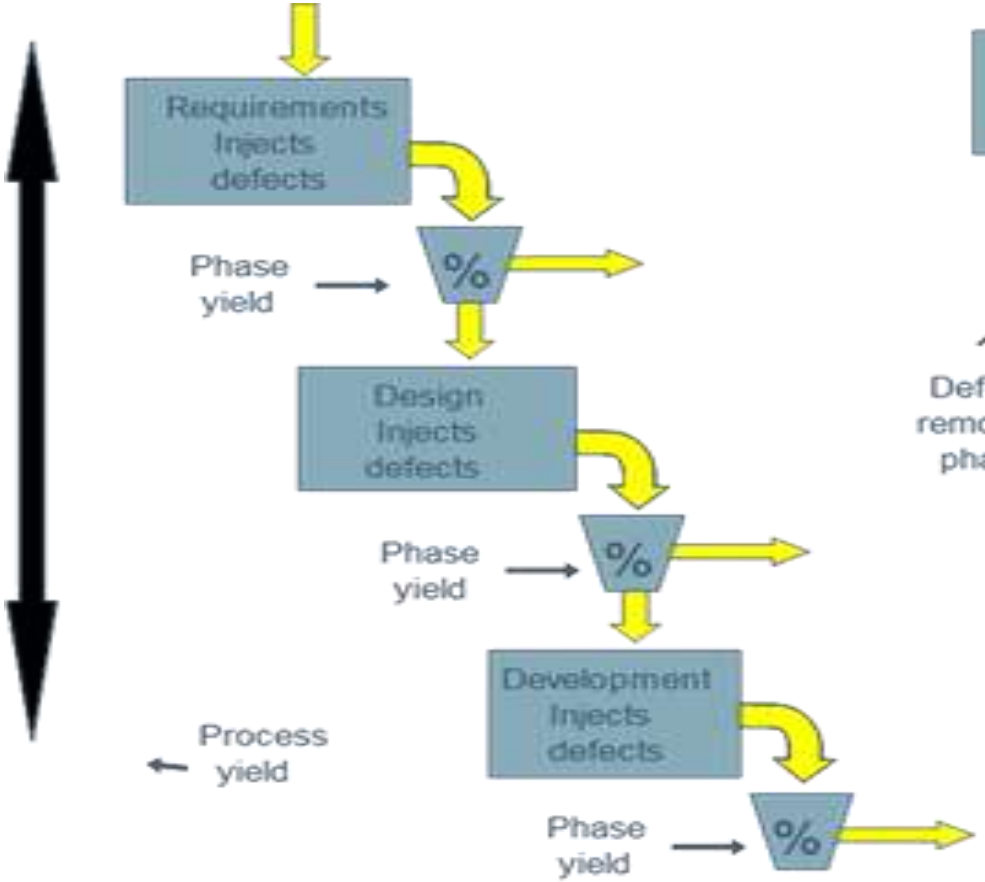
Org.	Project	Type	Secure or Safety Critical Defects	Defect Density	Size
D	D1	Safety Critical	20	46.07	2.8 MLOC
D	D2	Safety Critical	0	4.44	.9 MLOC
D	D3	Safety Critical	0	9.23	1.3 MLOC
A	A1	Secure	0	91.70	.6 MLOC
T	T1	Secure	0	20.00	.1 MLOC

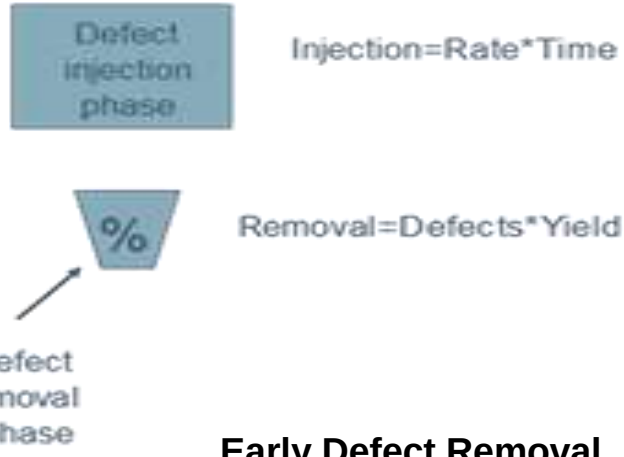
Data from five projects with low defect density in system testing reported very low or zero safety critical and security defects in production use.



Woody, Carol et al. *Predicting Software Assurance Using Quality and Reliability Measures.* CMU/SEI-2014-TN-026. Software Engineering Institute, Carnegie Mellon University. 2014. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=428589

Quality Focuses on Defect Injection and Removal

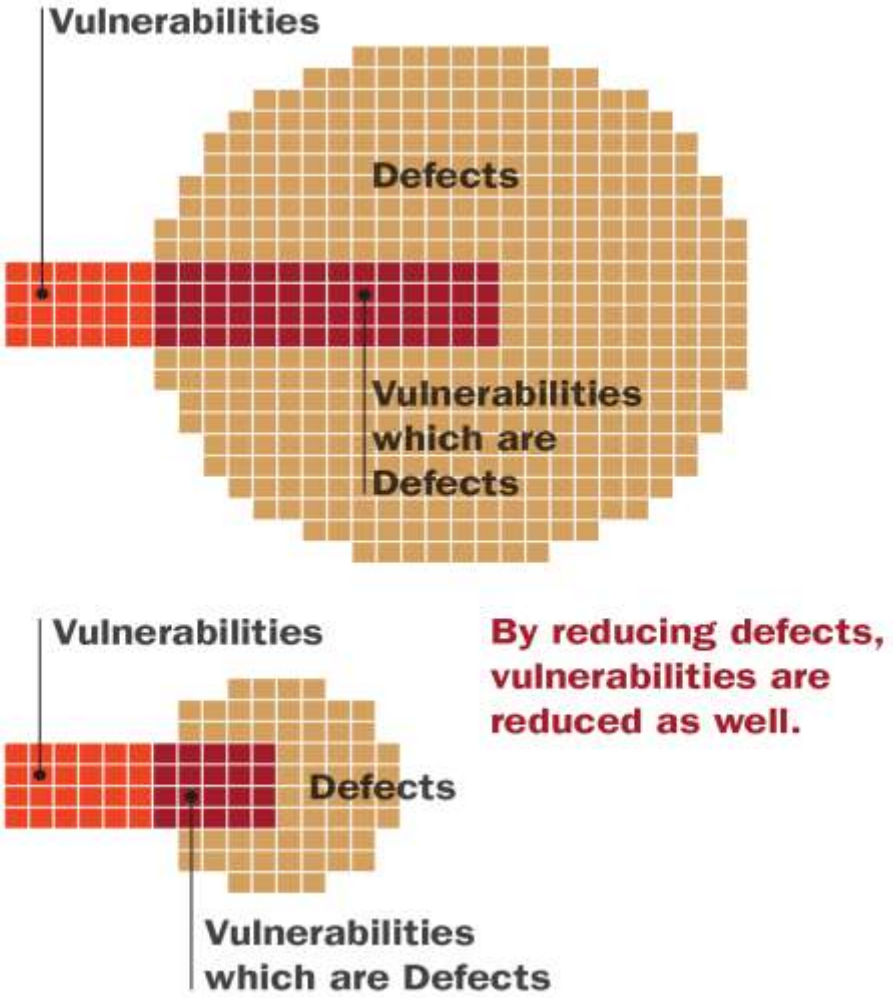




Early Defect Removal Across Lifecycle Reduces Cost

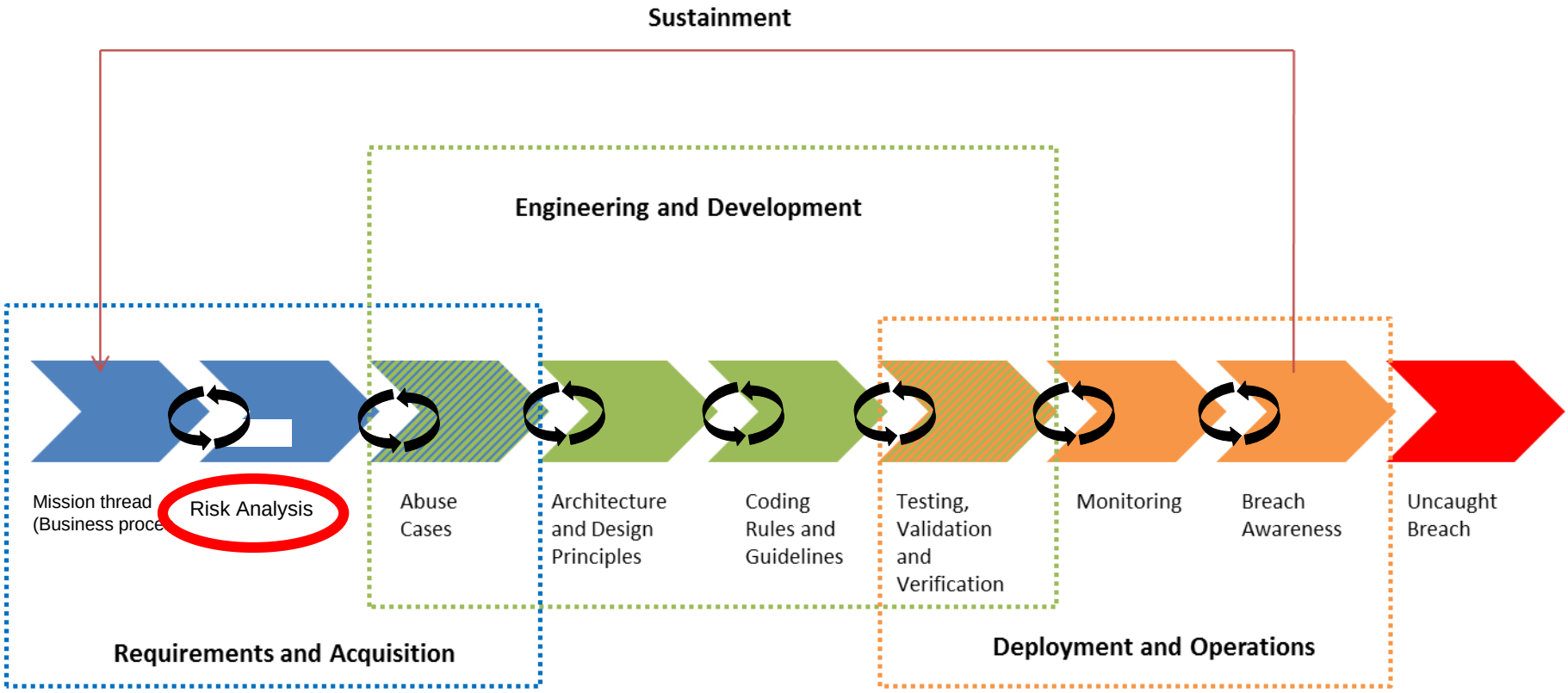
Poor quality predicts poor security
Effective quality removes defects at each step

Reducing Defects Reduces Potential Vulnerabilities



Security Engineering Risk Analysis (SERA) for a System of Systems

Software Security Is a Lifecycle Challenge



Design Weaknesses

Coding Weaknesses

Implementation Weaknesses

Security Engineering Risk Analysis (SERA)

What

- A systematic approach for analyzing complex security risks in a system of systems context

Why

- Build security into software-reliant systems by addressing design weaknesses as early as possible (e.g., requirements, architecture, design)
- Assemble a shared organizational view (business and technical) of system of system security risk

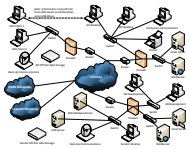
Benefits

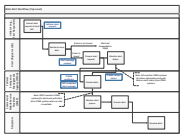
- Identify and correct design weaknesses before a system is deployed
- Reduce residual security risk in deployed systems

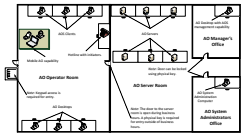
SERA Method: *Four Tasks*

1. Establish operational context.
2. Identify risk.
3. Analyze risk.
4. Develop control plan.

Modeling Techniques







Risk Identification Worksheet

Risk Evaluation Criteria

Risk Analysis Worksheet

Control Approach Worksheet

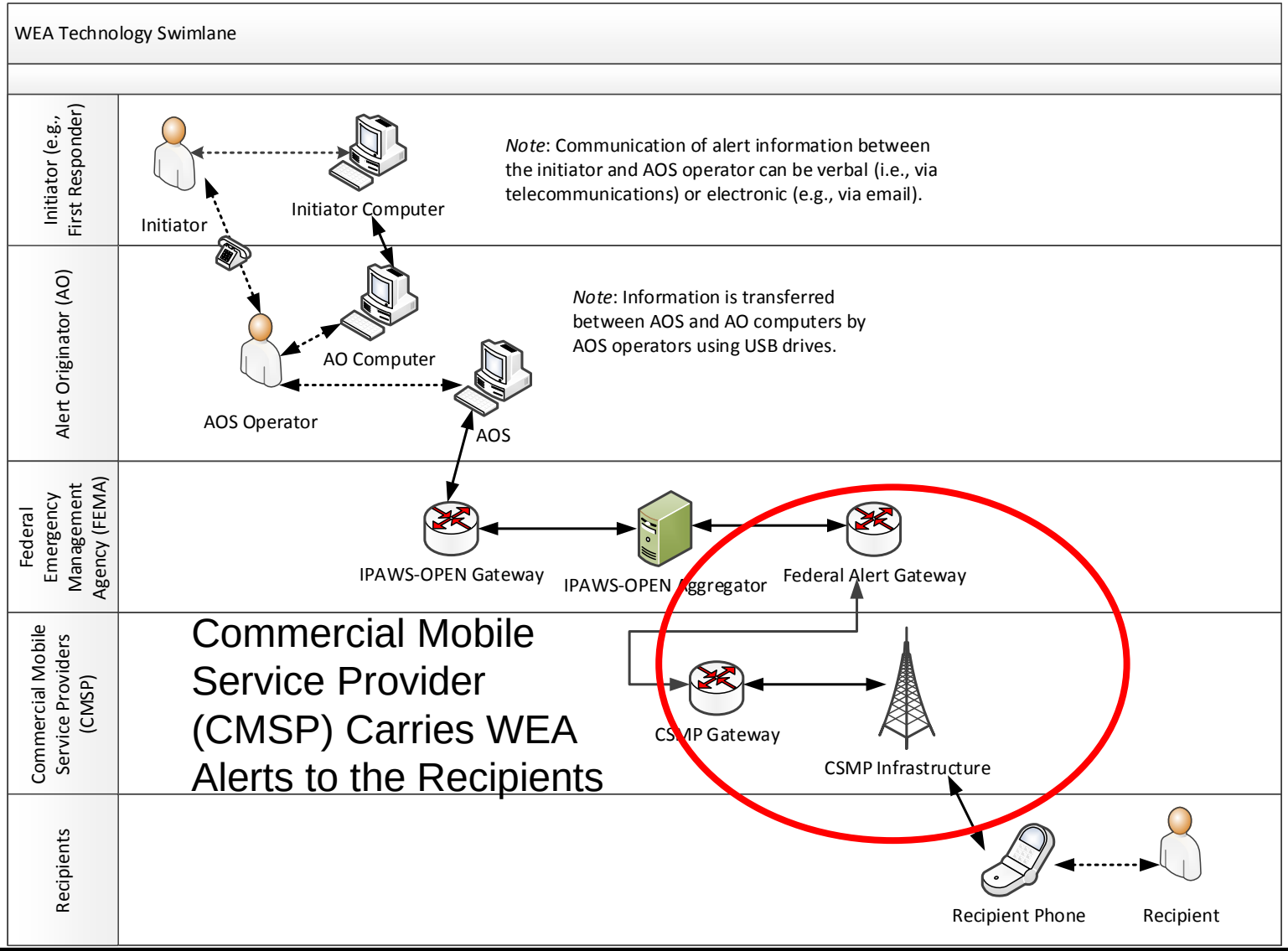
Control Plan Worksheet

Example: *Wireless Emergency Alerts (WEA)*

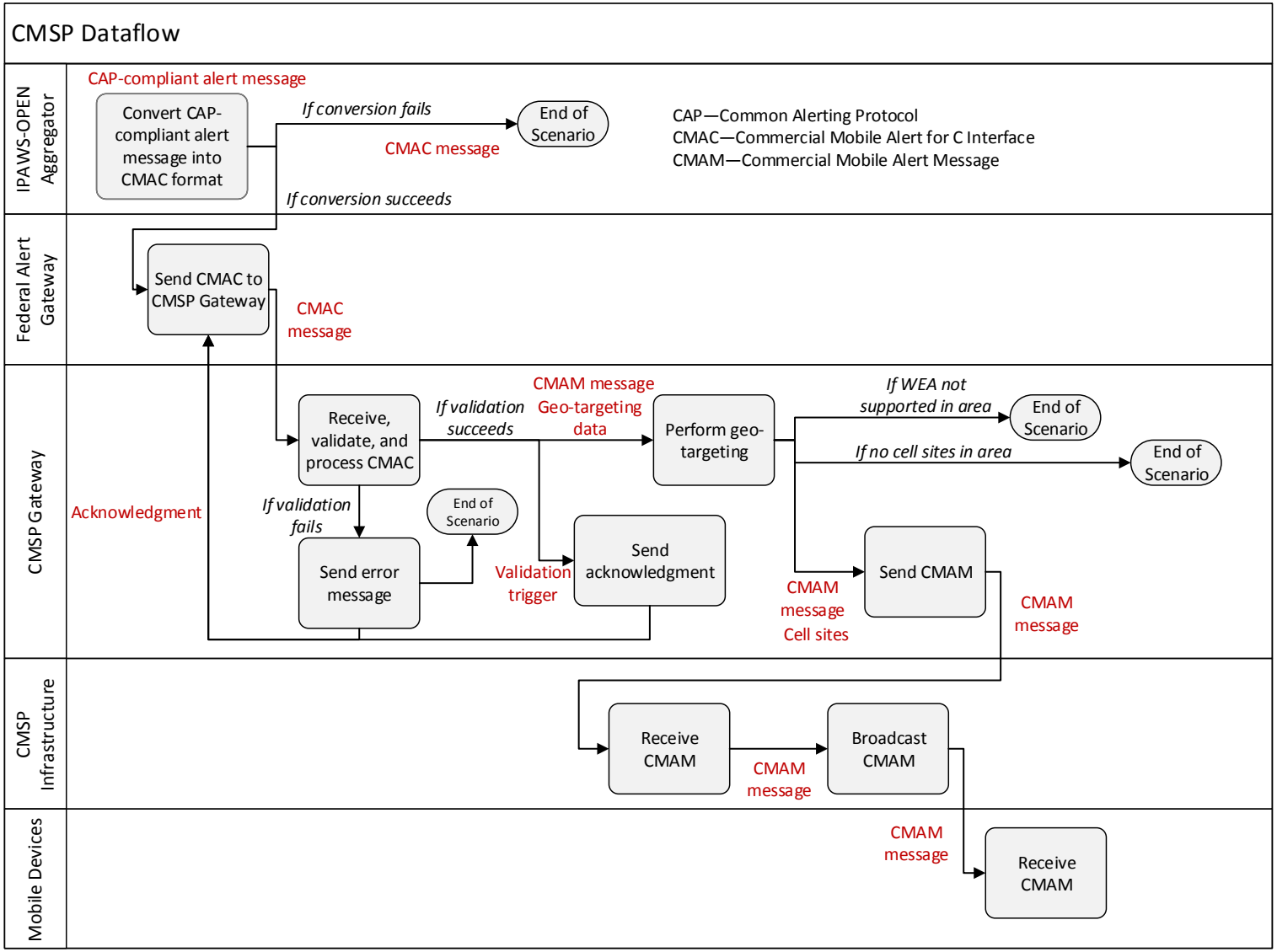
WEA is a major component of the Federal Emergency Management Agency (FEMA) Integrated Public Alert and Warning System (IPAWS).

- Enables federal, state, territorial, tribal, and local government officials to send targeted text alerts to the public via commercial mobile service providers (CMSPs).
- Customers of participating wireless carriers with WEA-capable mobile devices will automatically receive alerts in the event of an emergency if they are located in or travel to the affected geographic area.

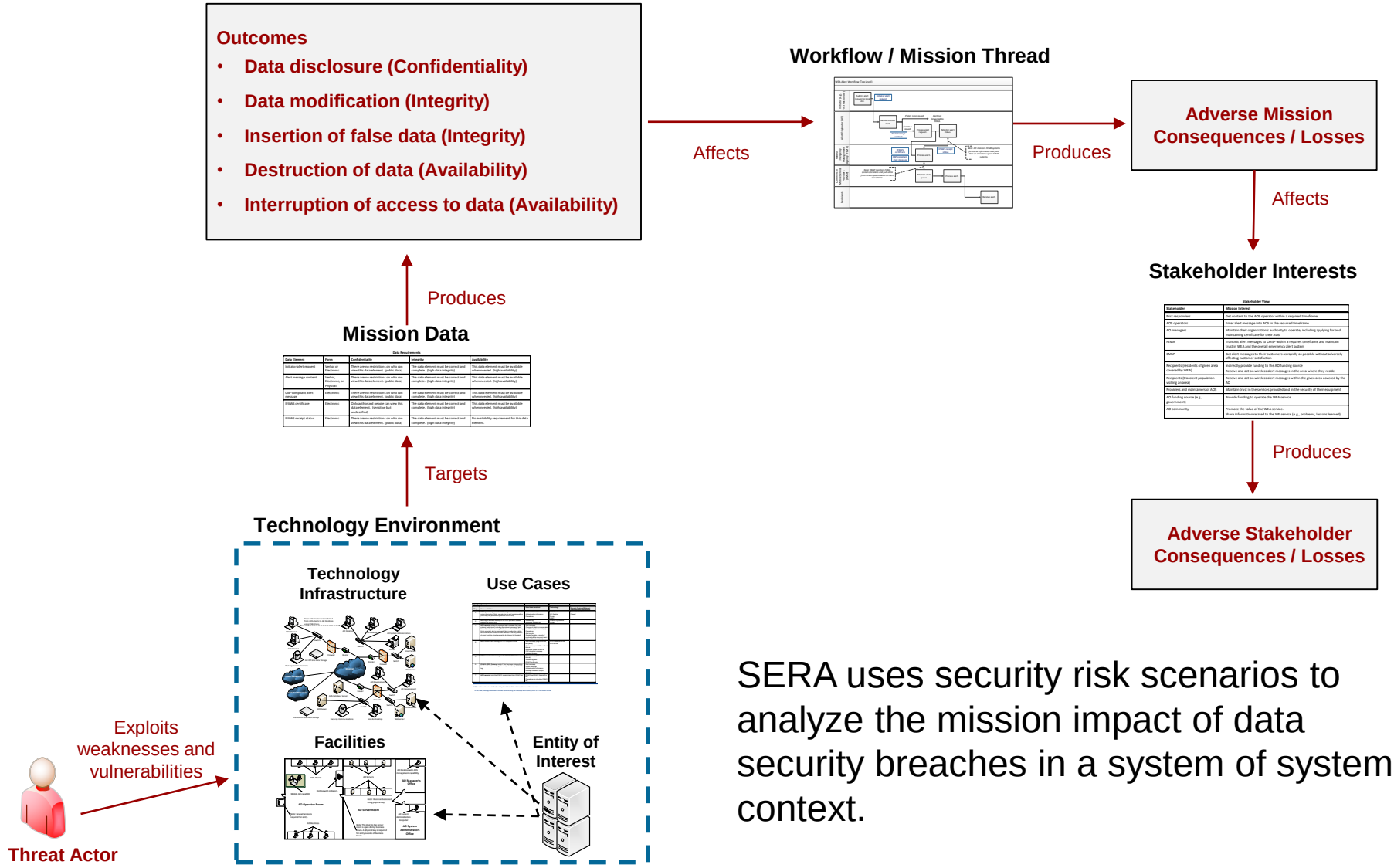
SERA Task 1: *WEA System of Systems*



SERA Task 1: *CMSP Dataflow*



SERA Task 2: *Mission Security Risk Scenarios*



SERA uses security risk scenarios to analyze the mission impact of data security breaches in a system of systems context.

SERA Task 2: *Elements of Security Risk Scenario*

Threat Components

- Actor – Motive – Goal – Outcome – Means – Threat Complexity

Threat Sequence

- Threat Step – Enabler(s)

Workflow Consequences

- Consequence – Amplifier(s)

Stakeholder Consequences

- Consequence – Amplifier(s)

SERA Task 2: *Security Risk Scenarios -1*

Example:

R1. Insider Sends False Alerts

- **IF** an insider with malicious intent uses the CMSP infrastructure to send nonsense alert messages repeatedly, **THEN** customers could become annoyed with the carrier; the carrier could incur considerable costs to recover from the attack; the carrier's reputation could be tarnished; and public trust in the WEA service could erode.

SERA Task 2: *R1 Threat Sequence*

T1. The insider is upset upon learning that he is not receiving a bonus this year and has been passed over for a promotion.

T2. The insider begins to behave aggressively and abusively toward his coworkers.

T3. The insider develops a logic bomb designed to replay a nonsense CMAM message repeatedly.

T4. The insider uses a colleague's workstation to check-in the modified code with the logic bomb.

T5. Seven months later, the insider voluntarily leaves the company for a position in another organization.

T6. Twenty-one days after the insider leaves the carrier, the logic bomb is activated automatically.

T7. The malicious code causes the carrier's WEA service to send a nonsense WEA alert repeatedly to people across the country.

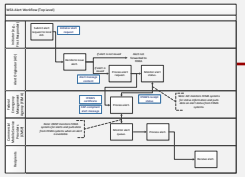
SERA Task 3: *Risk Measures*

Outcomes

- Data disclosure (Confidentiality)
- Data modification (Integrity)
- Insertion of false data (Integrity)
- Destruction of data (Availability)
- Interruption of access to data (Availability)

Affects

Workflow / Mission Thread

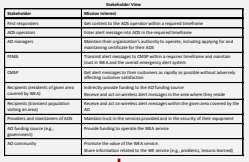


Produces

Adverse Mission Consequences / Losses

Affects

Stakeholder Interests

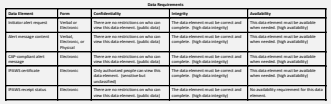


Produces

Adverse Stakeholder Consequences / Losses

Produces

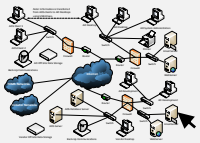
Mission Data



Targets

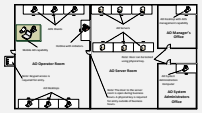
Technology Environment

Technology Infrastructure



Use Cases

Facilities



Entity of Interest

Exploits weaknesses and vulnerabilities

Threat Actor

Probability

Impact

Risk Exposure

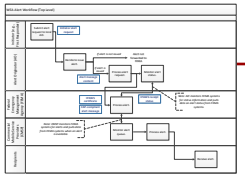
SERA Task 4: *Controls*

Outcomes

- Data disclosure (Confidentiality)
- Data modification (Integrity)
- Insertion of false data (Integrity)
- Destruction of data (Availability)
- Interruption of access to data (Availability)

Affects

Workflow / Mission Thread

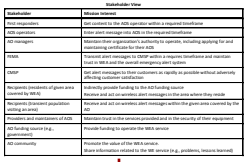


Produces

Adverse Mission Consequences / Losses

Affects

Stakeholder Interests

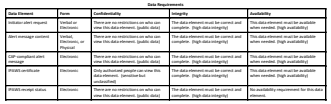


Produces

Adverse Stakeholder Consequences / Losses

Produces

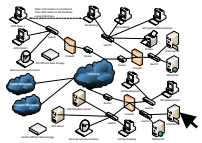
Mission Data



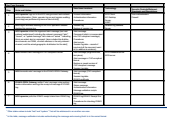
Targets

Technology Environment

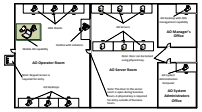
Technology Infrastructure



Use Cases



Facilities



Entity of Interest

Threat Actor

Exploits weaknesses and vulnerabilities

Reduce mission consequence amplifiers

Reduce stakeholder consequence amplifiers

Controls

Reduce threat enablers (e.g., weaknesses, vulnerabilities)

Engineering Security into the System of Systems

Software in Systems of Systems - 1

SoS Characteristic (Maier 1998)	Growing Insecurity	Engineering Software to be Secure
Operational Independence	Acquirers/Integrators assemble software from many vendors to seamlessly deliver end-to-end mission capability	Acquirers must identify and mitigate vulnerabilities in software performing mission-critical functions
Managerial Independence	Vendors focus on functionality and time to market to capture maximum market share; customers identify vulnerabilities through use	Acquirers must continually monitor, upgrade and patch all components to address known vulnerabilities
Evolutionary Development	Vendors release new functionality, which will include known fixes, to capture market share and drop support of older versions	Acquirers must upgrade critical software quickly to reduce the attack potential

Software in Systems of Systems - 2

SoS Characteristic (Maier 1998)	Growing Insecurity	Engineering Software to be Secure
Emergent Behavior	Acquirer's focus on least cost and speed of delivery; Vendors drive down costs through standardized interfaces (e.g. TCP/IP), reuse and early releases to dominate their niche markets; Vendor demand licenses that absolve them of liabilities	Acquirers must monitor quality and security related requirements in their vendor contracts and ensure vendors manage their software supply chains effectively (increased costs and increased oversight)
Geographic Distribution	Vendors deliver insecure-by-default software (faster and easier) with extensive connectivity	Acquirer must impose secure-by-default and increased quality requirements

Understand the System and Mission Security Risks

Weak perceptions of security risk lead to poor security decisions for systems and systems of systems

- Perceptions are primarily based on knowledge about successful attacks
  - the current state of security is largely reactive
  - successful organizations learn from attacks and figure out how to react and recover faster and be vigilant in anticipating and detecting attacks

Develop security attack scenarios to determine mission impact using SERA to evaluate "what if" possibilities

Interconnections Expand Access to Software Vulnerabilities

Highly connected systems require alignment of risk across all participants in the system of systems to ensure critical security risks are not ignored

- Decisions for reduced quality and accepted software vulnerabilities in one system can increase security risks for others in the system of systems
- Security must also be balanced with other critical qualities (performance, reliability, usability, etc.)
- Interactions occur at many technology levels (network, security appliances, architecture, applications, data storage, etc.) and are supported by a wide range of roles

Collaborative choices among participants in the system of systems are needed to address mission risk

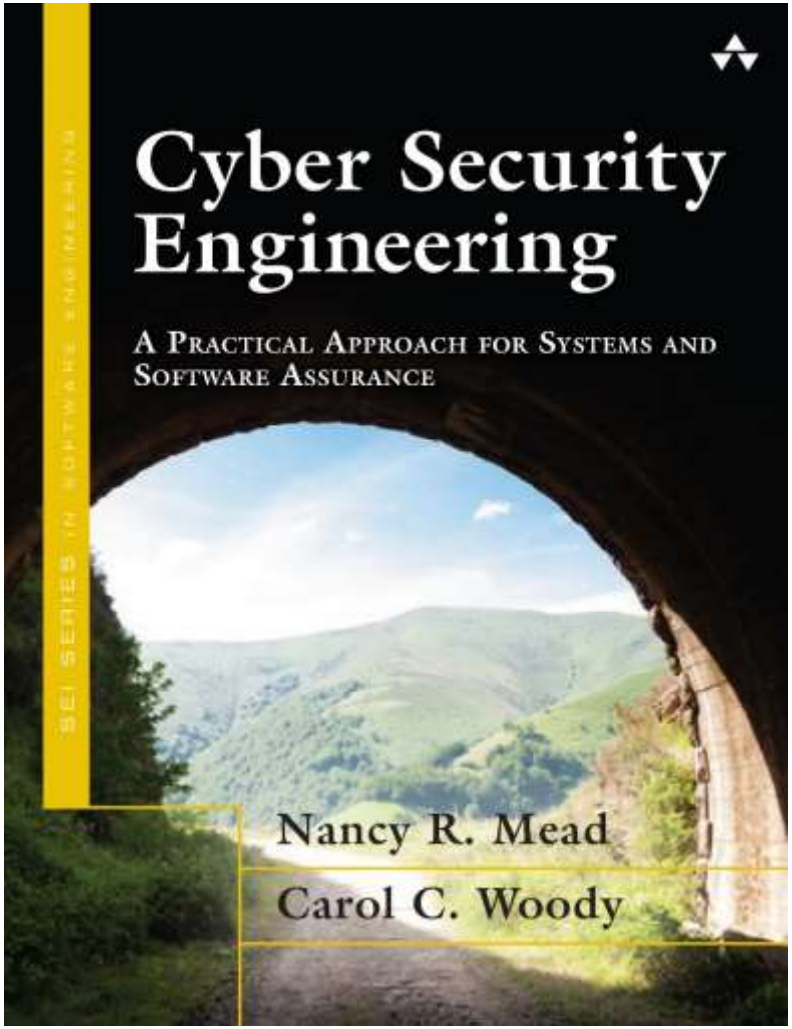
Attackers Do Not Respect System and Organizational Boundaries

There are no prefect protections against attacks.

Emphasize quality of software to reduce potential vulnerabilities.

There exists a broad community of attackers with growing technology capabilities able to compromise the confidentiality, integrity, and availability of any and all of your technology assets and the attacker capabilities are increasing.

Additional Materials



Released November 2016 as part of the SEI Book Series

Paperback and Kindle versions available from Amazon
https://www.amazon.com/Cyber-Security-Engineering-Practical-Assurance/dp/0134189809/ref=sr_1_3?crid=36L3HLV3PSL67&keywords=cyber+security+engineering&qid=1575936108&s=books&sprefix=cybersecurity+engineeri%2Caps%2C188&sr=1-3

CERT Cybersecurity Engineering and Software Assurance Professional Certificate

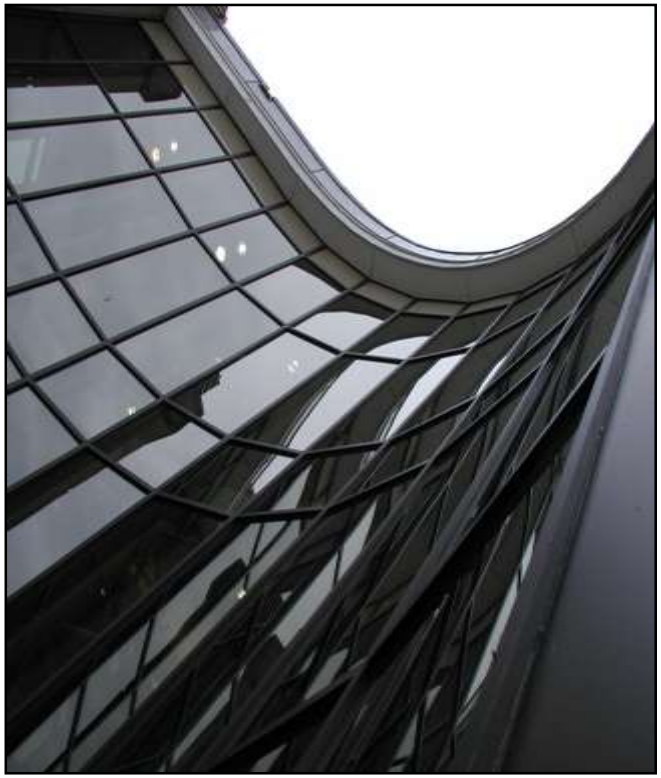
The CERT Division designed this program to arm software acquirers and developers, software and system assurance managers, systems engineers, and software engineers, with the skills and know-how to tackle the challenges of cybersecurity in acquired systems.

To learn more, visit https://sei.cmu.edu/education-outreach/credentials/credential.cfm?customel_datapageid_14047=33881.

Topics covered include:

- Software Assurance Methods
- Security Quality Requirements
- Security Risk Analysis
- Supply Chain Risk Management
- Threat Modeling

Contact Information



Carol Woody, Ph.D.

cwoody@cert.org

Web Resources (SEI)

www.sei.cmu.edu/go/cybersecurity-engineering

www.sei.cmu.edu/