SECURE CODE
ALLIANCE

Secure Code Alliance Body of Knowledge (SCA-BoK)

Developing Security & Privacy by Design

Version 2025.2

Table of Contents

EXECUTIVE SUMMARY

The Secure Code Alliance (SCA) is focused on technical competence. The Developing Security & Privacy by Design (DSPD) initiative is a conformity assessment methodology designed to issue individual-level certifications, specific to Secure Software Development Practices (SSDP).

Application developers (developers) are vital to the success of any organization, regardless of the industry. Developers create the tools that we all use, from operating systems to mobile apps, backend programming, firmware, front-end interface design and other forms of applications. Based on the new realities of an interconnected world that we live in, developers need to implement SSDP in order to protect their code from malicious attacks. By following SSDP, developers serve a crucial role in helping ensure security and safety, not just within an organization, but across the supply chain and society, as a whole.

Developers are in a unique position where they often have access to a wealth of sensitive information. As such, it is vital that developers create code with both security and privacy in mind, since improper coding practices can lead to exploitable vulnerabilities that enable hostile actors to affect the Confidentiality, Integrity, Availability and Safety (CIAS) of those affected systems, applications and/or services.

- *CONFIDENTIALITY* addresses preserving authorized restrictions on access and disclosure to authorized users and services, including means for protecting personal privacy and proprietary information.
- *INTEGRITY* addresses guarding against improper modification or destruction, including ensuring non-repudiation and authenticity.
- *AVAILABILITY* addresses timely, reliable access to data, systems and services for authorized users
- *SAFETY* addresses reducing risk associated with technologies that could fail or be manipulated by nefarious actors to cause death, injury, illness, damage to or loss of equipment.

This reality requires developers to adopt a "secure development mindset" that influences how their code will affect not only the secure functionality of the application, service or process, but how it affects the privacy and security of individuals who are not necessarily users, but anyone who is potentially affected by the application under development. Security and privacy must be considered during development and appropriately validated before it is released "into the wild."

The SCA supports the strategic cyber resiliency design principles that are established by NIST SP 800-160, Vol 2, Rev 1:[1]

- Focus on common critical assets;
- Support agility and architect for adaptability;
- Reduce attack surfaces;
- Assume compromised resources; and
- Expect adversaries to evolve.

Cyber resiliency is the ability to anticipate, withstand, recover from and adapt to adverse conditions, stresses, attacks or compromises on systems that use or are enabled by cyber resources. From a risk management perspective, cyber resiliency is intended to reduce the mission, business, organizational, or sector risk of depending on cyber resources.

The SCA references numerous leading industry frameworks for SSDP in an effort to provide "industry-recognized secure practices" references.

[1] *NIST SP 800-160 Vol 2 Rev 1 - https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-160v2r1.pdf*

About The Secure Code Alliance (SCA)

The Secure Code Alliance (SCA) was formed to address the need that organizations have to ensure its developers are aware of and implement Secure Software Development Practices (SSDP) in order to minimize the threat posed by malicious actors against the organization's Applications, Services and Processes (ASP).

The SCA's conformity assessment is the Developing Security & Privacy by Design (DSPD) initiative. The DSPD is an effort to promote transdisciplinary competency for developers to deliver trustworthy ASP. This concept of competency is focused on a practitioner's or architect's ability to: [2]

- Work with stakeholders to ensure that security objectives, protection needs/concerns, security requirements and associated validation methods are defined;
- Define security and privacy requirements, including associated verification methods;
- Develop security views and viewpoints of the system architecture and design;
- Identify and assess susceptibilities and vulnerabilities to lifecycle hazards and adversities;
- Design proactive and reactive features and functions encompassed within a balanced strategy to control asset loss and associated loss consequences;
- Provide security considerations to inform systems engineering efforts with the objective to reduce errors, flaws and weaknesses that may constitute a security vulnerability;
- Perform system security analyses and interprets the results of system security-relevant analyses in support of decision-making for engineering trades and risk management;
- Identify, quantify and evaluate the costs and benefits of security features and functions and considerations to inform assessments of alternative solutions, engineering trade-offs and risk treatment decisions;
- Demonstrate through evidence-based reasoning that security and trustworthiness claims for the system have been satisfied; and
- Leverage multiple security and other specialties to address all feasible solutions.

SCA Body of Knowledge (SCA-BoK)

For reference materials, the SCA's intent is to leverage freely-available content that are available at no cost to the public. In the realm of SSDP, there are certain voluntary consensus standards that are important to consider as industry-recognized practices and those primarily include, but are not limited to:

- NIST SP 800-218 [3]
- NIST SP 800-160 Vol 1 [4]
- NIST SP 800-160 Vol 2 [5]
- OWASP Top Ten [6]
- ISO/IEC/IEEE 15288 [7] (as referenced by NIST SP 800-160 vol 1)
- Microsoft Security Development Lifecycle [8]

Secure Software Development Framework (SSDF)

Few Software Development Life Cycle (SDLC) models explicitly address software security in detail, so it is necessary to add Secure Software Development Practices (SSDP) to each SDLC model to ensure that the application has adequate security and data protections "baked-in" during the development process. The SCA recognizes the Secure Software Development Framework (SSDF) [9] as a core set of high-level SSDP that can be integrated into a SDLC methodology. The SSDF offers multiple benefits that includes:

- Reducing the number of vulnerabilities in released software;
- Reducing the potential impact of the exploitation of undetected or unaddressed vulnerabilities;
- Addressing the root causes of vulnerabilities to prevent future recurrences;
- Providing a common vocabulary for SSDP; and
- Reduce miscommunications or assumption with parties in acquisition processes and other management activities.

---

[2] *NIST SP 800-160 Vol 2 Rev 1*
[3] *NIST SP 800-218 v1.1 - https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf*
[4] *NIST SP 800-160 Vol 1 Rev 1 - https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-160v1r1.pdf*
[5] *NIST SP 800-160 Vol 2 Rev 1 - https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-160v2r1.pdf*
[6] *OWASP Top 10 - https://owasp.org/www-project-top-ten/*
[7] *ISO/IEC/IEEE 15288 - https://www.iso.org/standard/63711.html*
[8] *Microsoft SDL - https://www.microsoft.com/en-us/securityengineering/sdl*
[9] *NIST SP 800-218 v1.1 - https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf*

SSDP are applicable for the following types of technology assets:

- Operating Systems (OS);
- Firmware;
- Mobile device apps;
- General-purpose or multi-use systems (e.g., Enterprise Information Technology (EIT));
- Dedicated or special-purpose systems (e.g., security-dedicated/purposed systems), such as Operational Technology (OT) devices that used in industrial/manufacturing systems that includes:
    - Industrial Control Systems (ICS);
    - Supervisory Control and Data Acquisition (SCADA) systems;
    - Programmable Logic Controllers (PLCs);
    - Computerized Numerical Control (CNC) devices;
    - Cyber-Physical Systems (CPS);
    - Machine controllers;
    - Fabricators;
    - Assemblers; and
    - Machining technologies; and
- Internet of Things (IoT) / Industrial Internet of Things (IIoT) that are interconnected devices having physical or virtual representation in the digital world, sensing/actuation capability and programmability features that includes:
    - Wearable technologies;
    - Security systems;
    - Lighting;
    - Heating
    - Air conditioning; and
    - Fire / smoke detectors.

MISSION

The SCA's mission is to improve the awareness and adherence to Secure Software Development Practices (SSDP) by application developers and architects through operating a conformity assessment methodology that:

- Spans the design, development and maintenance of Applications, Services and Processes (ASP);
- Educates applicants through reinforcing reasonably-expected security and privacy principles, based on voluntary consensus standards that considered developer-specific industry-recognized practices; and
- Leverages an online platform to test applicants on subject matter expertise that awards the applicant with a Certificate of Conformity (CoC) upon receiving a successful score.

VISION

The SCA's vision is that organizations from all industries ensure that the development of Applications, Services and Processes (ASP) employ adequate security and privacy measures throughout the Software Development Life Cycle (SDLC) to ensure security and privacy-related risks are identified and remediated appropriately.

The SCA's conformity assessment methodology is designed with these concepts in mind:

- Identify the discipline basics for Secure Software Development Practices (SSDP) in terms of its principles, concepts and activities; and
- Foster a common mindset to deliver secure Applications, Services and Processes (ASP), regardless of its purpose, type, scope, size, complexity, or stage of the SDLC.

STRATEGY

The SCA's strategy is to:

- Operate a cost-effective and meaningful conformity assessment methodology, the Developing Security & Privacy by Design (DSPD) initiative; and
- Certify individuals for competency among application developers and architects. There are two certifications available:
    (1) Certified SCA Practitioner (CSCAP) (*designed for developer roles)*
    (2) Certified SCA Architect (CSCAA) *(designed for application architecture roles)*

The DSPD initiative is focused on developing a conformity assessment methodology that addresses:

- "Practitioner-level competency" among developers; and
- "Expert-level competency" among architects.

Developing Security & Privacy by Design (DSPD) Conformity Assessment

As a personnel certification body, the SCA determines if an applicant fulfils certification requirements.[10] Each applicant's subject matter expertise on selected voluntary consensus standards is tested to determine if an acceptable level of competency is met.

Per ISO/IEC 17024 guidelines, certifications:

- Are meant to be a public statement or declaration that an individual has passed an examination and otherwise met specified criteria demonstrating that the individual has the competencies necessary to successfully perform the role and responsibilities that comprise a specific occupation;
- Are granted for a limited period of time;
- Must be renewed to ensure that individuals continue to possess the competencies required to perform the job; and
- May require ongoing education and/or assessment and/or experience for renewal.

The Developing Security & Privacy by Design (DSPD) initiative's conformity assessment leverages an online platform to test applicants on subject matter expertise through a one hundred (100) question set of multiple-choice problems. The DSPD leverages the three (3) general types of test questions and principle areas of focus that are used when constructing test questions:

(1) Recall;
(2) Application; and
(3) Analysis.

Recall

Recall questions are designed to assess:

- Basic facts;
- Definitions;
- Concepts;
- Principles;
- Processes and procedures; and
- Generalizations

Recall questions typically test the recognition or memory of isolated information from study materials and subject matter. Performance is not dependent on the acquisition and/or practice of skills, where it can be achieved through lectures and/or reading.

Recall questions:

- Are well-suited for beginning topics to bolster student confidence;
- Should be less utilized as courses progress into more complex topics; and
- May also be used in addition to advanced items as "fillers."

Application

Application items/questions are more complex than simple recall questions and are designed to assess the acquisition and/or practice of skills:

- Additional understanding of concepts and skills may be required to distinguish between plausible distractors and the one correct key answer;
- Sets of variables may be provided within the stem or on a table or chart; and
- Scenarios may be provided initially with several multiple-choice items following which relate back to the data provided within them.

Application questions:

- May include hierarchical progressions and sequencing of steps in application-type questions;

[10] *The SCA is not accredited under the ISO/IEC 17024, but does leverage the framework as a guideline for the DSPD.*

- Avoid overly wordy stems that teach or preach and add non-relevant data; and
- Due to the higher complexity, ensure that the question being asked is clear and that only one correct (key) choice can be made.

Analysis

Analysis items/questions represent the highest level of complexity in test preparation and performance:

- The student/examinee must demonstrate the ability to synthesize multiple variables beyond simple recall of basic facts; and
- Application of ordered steps demonstrates conceptual thinking

Analysis questions include:

- Analysis-type items/questions;
- The evaluation of data;
- Complex problem solving;
- Making judgments about best, most appropriate or effective course of action in a given situation or scenario; and
- Integrating aspects of various areas of curricula to formulate a complete picture of the data represented in the stem or given scenario

COMPETENCY EXPECTATIONS

The Secure Code Alliance (SCA) is focused on technical competence and it expects developers to invest the requisite time and effort necessary to familiarize themselves with referenced materials, since these voluntary consensus standards form the basis of the SCA Body of Knowledge (SCA-BoK) that is leveraged in the conformity assessment.

For practical purposes, individuals who earn a *Certified SCA Practitioner (CSCAP)* or *Certified SCA Architect (CSCAA)* Certificate of Conformity (CoC) have demonstrated a level of competence necessary to ensure that the security of an organization's Applications, Services and Processes (ASP) are assessed throughout their operational life to reduce risks to the organization and its clients. These certifications are valid for a period of three (3) years from the date of issue of the CoC, at which point the certification expires and will need to be renewed through a re-examination.

Practitioner Role - Certified SCA Practitioner (CSCAP)

Application developers (practitioners) are expected to use Secure Development Lifecycle (SDL) processes for new systems, system upgrades, or systems that are being repurposed. These processes can be employed at any stage of the system lifecycle and can take advantage of any system or software development methodology, including agile, spiral, or waterfall.

CSCAPs are expected to:

- Apply lifecycle processes recursively, iteratively, concurrently, sequentially, or in parallel and to any system regardless of its size, complexity, purpose, scope, environment of operation, or special nature.
- Understand and operationalize the organization's security architecture that must be followed for application development processes for development, testing, staging and production environments.
- Incorporate the organization's risk management practices throughout application development processes across the entire SDLC.
- Develop software applications in accordance with industry-recognized secure coding practices.
- Incorporate security and privacy measures throughout the SDLC.
- Control changes to ASP across the SDLC using formal change control procedures.
- Review custom code through a formal change management and approval process prior to release to production.
- Remove custom application accounts, user IDs and passwords before applications become active or are released to customers.
- Confidently review SBOM documentation for security and privacy-related implications.
- Perform software conformity assessments.

Architect Role - Certified SCA Architect (CSCAA)

Architects are expected to employ cyber resiliency constructs (e.g., goals, objectives, techniques, approaches and design principles), as well as the analytic and lifecycle processes, to tailor them to the technical, operational and threat environments for which the architect's systems need to be engineered.

CSCAAs are expected to:

- Define the security architecture(s) the organization will follow for application development processes.
- Define application development considerations for the organization's risk management practices across the entire Software Development Life Cycle (SDLC).
- Publish rules for the organization's application development processes for development, testing, staging and production environments.
- Develop conformity assessment practices for the organization to follow in order to demonstrate alignment with stated Secure Software Development Practices.
- Ensure that information security and privacy principles are an integral part of Secure Software Development Practices (SSDP) across the entire SDLC.
- Ensure security & privacy-related measures are included in the requirements for new systems or enhancements to existing systems.
- Ensure application development practices (internal and external) adhere to industry-recognized secure coding practices.
- Develop Software Bill of Materials (SBOM) documentation for application development projects.
- Oversee changes to ASP across the SDLC using formal change control procedures.
- Oversee application security testing practices.
- implement the SSDP concepts and techniques for all High Value Assets (HVA):
    - New Systems;
    - Dedicated or Special-Purpose Systems;
    - System of Systems;
    - System Modifications;
    - System Evolution; and
    - System Retirement.

SECURE CODE ALLIANCE (SCA) ECOSYSTEM OVERVIEW

The Secure Code Alliance (SCA) is focused on technical competence of both individuals and organizations that develop Applications, Services and Processes (ASP). With evolving statutory and regulatory requirements mandating Secure Development Practices (SDP), the SCA is uniquely situated to assist with evidence to demonstrate familiarity with and a commitment to SDP at the:

- Individual-level; and
- Organization-level.

INDIVIDUAL-LEVEL

The SCA expects developers to invest the requisite time and effort necessary to familiarize themselves with referenced materials, since these voluntary consensus standards form the basis of the SCA Body of Knowledge (SCA-BoK) that is leveraged in the conformity assessment.[11]

The SCF Assessor and Instructor Certification Organization (SAICO) is authorized by the SCA to conduct individual-level certification-related services for the:

1. Certified SCA Architect (CSCAA) - "practitioner-level competency" among developers; and
2. Certified SCA Architect (CSCAA) – "expert-level competency" among architects.

Practitioner Role - Certified SCA Practitioner (CSCAP)

Application developers (practitioners) are expected to use Software Development Life Cycle (SDLC) processes for new systems, system upgrades, or systems that are being repurposed. These processes can be employed at any stage of the system lifecycle and can take advantage of any system or software development methodology, including agile, spiral, or waterfall.

CSCAPs are expected to:

- Apply lifecycle processes recursively, iteratively, concurrently, sequentially, or in parallel and to any system regardless of its size, complexity, purpose, scope, environment of operation, or special nature.
- Understand and operationalize the organization's security architecture that must be followed for application development processes for development, testing, staging and production environments.
- Incorporate the organization's risk management practices throughout application development processes across the entire SDLC.
- Develop software applications in accordance with industry-recognized secure coding practices.
- Incorporate security and privacy measures throughout the SDLC.
- Control changes to ASP across the SDLC using formal change control procedures.
- Review custom code through a formal change management and approval process prior to release to production.
- Remove custom application accounts, user IDs and passwords before applications become active or are released to customers.
- Confidently review SBOM documentation for security and privacy-related implications.
- Perform software conformity assessments.

Architect Role - Certified SCA Architect (CSCAA)

Architects are expected to employ cyber resiliency constructs (e.g., goals, objectives, techniques, approaches and design principles), as well as the analytic and lifecycle processes, to tailor them to the technical, operational and threat environments for which the architect's systems need to be engineered.

CSCAAs are expected to:

- Define the security architecture(s) the organization will follow for application development processes.
- Define application development considerations for the organization's risk management practices across the entire SDLC.
- Publish rules for the organization's application development processes for development, testing, staging and production environments.
- Develop conformity assessment practices for the organization to follow in order to demonstrate alignment with stated Secure Software Development Practices.

[11] *SCA BoK - https://securecodealliance.com/sca-bok/*

- Ensure that information security and privacy principles are an integral part of Secure Software Development Practices (SSDP) across the entire SDLC.
- Ensure security & privacy-related measures are included in the requirements for new systems or enhancements to existing systems.
- Ensure application development practices (internal and external) adhere to industry-recognized secure coding practices.
- Develop Software Bill of Materials (SBOM) documentation for application development projects.
- Oversee changes to ASP across the SDLC using formal change control procedures.
- Oversee application security testing practices.
- Implement the SSDP concepts and techniques for all High Value Assets (HVA):
    - New Systems;
    - Dedicated or Special-Purpose Systems;
    - System of Systems;
    - System Modifications;
    - System Evolution; and
    - System Retirement.

ORGANIZATION-LEVEL

There are two (2) types of organization-level offerings:

1. Secure Development Organization (SDO) designation; and
2. Certified Organization for Development Excellence (CODE) certification.

The Cyber AB governs organization-level SCA designations and certifications.[12]

Secure Development Organization (SDO)

The SDO designation was developed as a way for organizations to clearly identify a commitment to SDP, through:

1. Adherence to the respective requirements and constructs of the SCA framework; and
2. Employing SCA-certified individuals to operationalize SDP.

There are three (3) levels of SDO:

1. **SDO**
    a. Organization is registered with the Cyber AB as a SDO; and
    b. Employs at least one (1) CSCAP.
2. **SDO Advanced**
    a. Organization is registered with the Cyber AB as a SDO; and
    b. Employs at least three (3) CSCAP.
3. **SDO Elite**
    a. Organization is registered with the Cyber AB as a SDO; and
    b. Employees at least:
        i. Three (3) CSCAP; and
        ii. One (1) CSCAA.

Certified Organization for Development Excellence (CODE)

The concept of the Certified Organization for Development Excellence (CODE) certification is to utilize a third-party conformity assessment of SDP. CODE certification is exclusive of a SDO designation, where an organization does not have to be a SDO to seek/obtain CODE certification.

For CODE, the SCA:

- Appointed The Cyber AB to serve as the Accreditation Body (AB) for the SCA's organization-level certification scheme; and
- Leverages the Secure Controls Framework Conformity Assessment Program (SCF CAP) for the methodology and infrastructure to conduct the conformity assessment.[13]

[12] *The Cyber AB - https://cyberab.org/*
[13] *SCF CAP - https://securecontrolsframework.com/certification/scf-conformity-assessment-program-cap/*

As part of the SCF CAP, an Organization Seeking Assessment (OSA) hires a SCF Third-Party Assessment Organization (SCF 3PAO) to perform Third-Party Assessment, Attestation & Certification (3PAAC) services.

There are three (3) CODE levels that are designed to be progressive, building upon the previous CODE level:

1. **SCF Certified - SCA CODE 1**
    a. Focus: Organization-level attestation of SDP.
    b. Control Set: The CISA Secure Software Development Attestation Form (SSDAF) is used as the basis for CODE 1 certification.[14] CISA derived the SSDAF directly from the requirements outlined in Executive Order (EO) 14028.[15]
    c. Prerequisite(s): None
2. **SCF Certified - SCA CODE 2**
    a. Focus: Commercial Off The Shelf (COTS) Applications, Services and Processes (ASP)
    b. Control Set: NIST SP 800-218, Secure Software Development Framework (SSDF).[16]
    c. Prerequisite(s): SCA CODE 1 (may be conducted in conjunction with SCA CODE 1)
3. **SCF Certified - SCA CODE 3**
    a. Focus: Custom ASP
    b. Control Set: Tailored SCF control set (bespoke for the specific use case).
    c. Prerequisite(s): SCA CODE 2

[14] *CISA Secure Software Development Attestation Form - https://www.cisa.gov/secure-software-attestation-form*
[15] *EO 14028 - https://www.federalregister.gov/documents/2021/05/17/2021-10460/improving-the-nations-cybersecurity*
[16] *NIST SP 800-218 - https://csrc.nist.gov/pubs/sp/800/218/final*

PRACTITIONER-LEVEL: COMMON REQUIREMENTS FOR SDP

From a day-to-day perspective of requirements for Secure Development Practices (SDP), there are certain frameworks that impact nearly every organization, regardless of the industry it serves.

This section identifies the most common SDP-related controls from leading cybersecurity and data protection laws, regulations and frameworks.

EXECUTIVE ORDER (EO) 14028

The following SDP-related requirements are from EO 14028, Enhancing Software Supply Chain Security, include:

- Sec 4(e)(i) secure software development environments, including such actions as:
  - using administratively separate build environments;
  - auditing trust relationships;
  - establishing multi-factor, risk-based authentication and conditional access across the enterprise;
  - documenting and minimizing dependencies on enterprise products that are part of the environments used to develop, build, and edit software;
  - employing encryption for data; and
  - monitoring operations and alerts and responding to attempted and actual cyber incidents;
- Sec 4(e) (ii) generating and, when requested by a purchaser, providing artifacts that demonstrate conformance to the processes set forth in subsection (e)(i) of this section;
- Sec 4(e) (iii) employing automated tools, or comparable processes, to maintain trusted source code supply chains, thereby ensuring the integrity of the code;
- Sec 4(e) (iv) employing automated tools, or comparable processes, that check for known and potential vulnerabilities and remediate them, which shall operate regularly, or at a minimum prior to product, version, or update release;
- Sec 4(e) (v) providing, when requested by a purchaser, artifacts of the execution of the tools and processes described in subsection (e)(iii) and (iv) of this section, and making publicly available summary information on completion of these actions, to include a summary description of the risks assessed and mitigated;
- Sec 4(e) (vi) maintaining accurate and up-to-date data, provenance (i.e., origin) of software code or components, and controls on internal and third-party software components, tools, and services present in software development processes, and performing audits and enforcement of these controls on a recurring basis;
- Sec 4(e) (vii) providing a purchaser a Software Bill of Materials (SBOM) for each product directly or by publishing it on a public website;
- Sec 4(e) (viii) participating in a vulnerability disclosure program that includes a reporting and disclosure process;
- Sec 4(e) (ix) attesting to conformity with secure software development practices; and
- Sec 4(e) (x) ensuring and attesting, to the extent practicable, to the integrity and provenance of open source software used within any portion of a product.

NIST SP 800-171 R2 & R3 / CYBERSECURITY MATURITY MODEL CERTIFICATION (CMMC)

The following SDP-related requirements are from NIST SP 800-171 and Cybersecurity Maturity Model Certification (CMMC), include:

NIST SP 800-171 R2

- 3.4.1 - Establish and maintain baseline configurations and inventories of organizational systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles.
- 3.12.2 - Develop and implement plans of action designed to correct deficiencies and reduce or eliminate vulnerabilities in organizational systems.
- 3.13.2 - Employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational systems.
- 3.13.11 - Employ FIPS-validated cryptography when used to protect the confidentiality of CUI.
- 3.13.13 - Control and monitor the use of mobile code.

NIST SP 800-171 R3

- 03.04.01.b - Review and update the baseline configuration of the system [Assignment: organization-defined frequency] and when system components are installed or modified.
- 03.05.07.d - Store passwords in a cryptographically protected form.

- 03.11.01.a - Assess the risk (including supply chain risk) of unauthorized disclosure resulting from the processing, storage, or transmission of CUI.
- 03.13.08 - Implement cryptographic mechanisms to prevent the unauthorized disclosure of CUI during transmission and while in storage.
- 03.13.13.a - Define acceptable mobile code and mobile code technologies.
- 03.13.13.b - Authorize, monitor, and control the use of mobile code.
- 03.13.15 - Protect the authenticity of communications sessions.
- 03.14.01.a - Identify, report, and correct system flaws.
- 03.14.01.b - Install security-relevant software and firmware updates within [Assignment: organization-defined time period] of the release of the updates.
- 03.17.01.a - Develop a plan for managing supply chain risks associated with the research and development, design, manufacturing, acquisition, delivery, integration, operations, maintenance, and disposal of the system, system components, or system services.
- 03.17.02 - Develop and implement acquisition strategies, contract tools, and procurement methods to identify, protect against, and mitigate supply chain risks.
- 03.17.03.b - Enforce the following security requirements to protect against supply chain risks to the system, system components, or system services and to limit the harm or consequences from supply chain-related events: [Assignment: organization-defined security requirements].

NIST SP 800-53 R5

The following SDP-related requirements are from NIST SP 800-53 R5, include:

- SA-3 - System Development Life Cycle
- SA-3(2) - Use of Live or Operational Data
- SA-4 - Acquisition Process
- SA-4(3) - Development Methods, Techniques and Practices
- SA-4(9) - Functions, Ports, Protocols and Services In Use
- SA-8 - Security and Privacy Engineering Principles
- SA-8(28) - Acceptable Security
- SA-8(32) - Sufficient Documentation
- SA-8(33) - Minimization
- SA-10 - Developer Configuration Management
- SA-11 - Developer Security Testing and Evaluation
- SA-11(1) - Static Code Analysis
- SA-11(2) - Threat Modeling and Vulnerability Analysis
- SA-11(3) - Independent Verification of Assessment Plans and Evidence
- SA-11(4) - Manual Code Reviews
- SA-11(5) - Penetration Testing
- SA-11(6) - Attack Surface Reviews
- SA-11(7) - Verify Scoping of Testing and Evaluation
- SA-11(8) - Dynamic Code Analysis
- SA-15 - Development Process, Standards and Tools
- SA-15(1) - Quality Metrics
- SA-15(2) - Security and Privacy Tracking Tools
- SA-15(3) - Criticality Analysis
- SA-15(5) - Attack Surface Reduction
- SA-15(6) - Continuous Improvement
- SA-15(7) - Automated Vulnerability Analysis
- SA-15(8) - Reuse of Threat and Vulnerability Information
- SA-15(10) - Incident Response Plan
- SA-15(11) - Archive System or Component
- SA-15(12) - Minimize Personally Identifiable Information
- SA-16 - Developer-Provided Training
- SA-17 - Developer Security and Privacy Architecture and Design
- SA-17(1) - Formal Policy Model
- SA-17(2) - Security-Relevant Components

- SA-17(5) - Conceptually Simple Design
- SA-17(6) - Structure for Testing
- SA-17(7) - Structure for Least Privilege
- SA-21 - Developer Screening
- SA-22 - Unsupported System Components
- SA-23 – Specialization

PAYMENT CARD INDUSTRY DATA SECURITY STANDARD (PCI DSS)

The following SDP-related requirements are from PCI DSS v4 include:

- 6.2.1- Bespoke and custom software are developed securely, as follows:
  - Based on industry standards and/or best practices for secure development.
  - In accordance with PCI DSS (for example, secure authentication and logging).
  - Incorporating consideration of information security issues during each stage of the software development lifecycle.
- 6.2.2- Software development personnel working on bespoke and custom software are trained at least once every 12 months as follows:
  - On software security relevant to their job function and development languages.
  - Including secure software design and secure coding techniques.
  - Including, if security testing tools are used, how to use the tools for detecting vulnerabilities in software.
- 6.2.3- Bespoke and custom software is reviewed prior to being released into production or to customers, to identify and correct potential coding vulnerabilities, as follows:
  - Code reviews ensure code is developed according to secure coding guidelines.
  - Code reviews look for both existing and emerging software vulnerabilities.
  - Appropriate corrections are implemented prior to release.
- 6.2.3.1- If manual code reviews are performed for bespoke and custom software prior to release to production, code changes are:
  - Reviewed by individuals other than the originating code author, and who are knowledgeable about code-review techniques and secure coding practices.
  - Reviewed and approved by management prior to release.
- 6.2.4- Software engineering techniques or other methods are defined and in use by software development personnel to prevent or mitigate common software attacks and related vulnerabilities in bespoke and custom software, including but not limited to the following:
  - Injection attacks, including SQL, LDAP, XPath, or other command, parameter, object, fault, or injection-type flaws.
  - Attacks on data and data structures, including attempts to manipulate buffers, pointers, input data, or shared data.
  - Attacks on cryptography usage, including attempts to exploit weak, insecure, or inappropriate cryptographic implementations, algorithms, cipher suites, or modes of operation.
  - Attacks on business logic, including attempts to abuse or bypass application features and functionalities through the manipulation of APIs, communication protocols and channels, client-side functionality, or other system/application functions and resources. This includes cross-site scripting (XSS) and cross-site request forgery (CSRF).
  - Attacks on access control mechanisms, including attempts to bypass or abuse identification, authentication, or authorization mechanisms, or attempts to exploit weaknesses in the implementation of such mechanisms.
  - Attacks via any “high-risk” vulnerabilities identified in the vulnerability identification process, as defined in Requirement 6.3.1.
- 6.3.1- Security vulnerabilities are identified and managed as follows:
  - New security vulnerabilities are identified using industry-recognized sources for security vulnerability information, including alerts from international and national computer emergency response teams (CERTs).
  - Vulnerabilities are assigned a risk ranking based on industry best practices and consideration of potential impact.
  - Risk rankings identify, at a minimum, all vulnerabilities considered to be a high-risk or critical to the environment.
  - Vulnerabilities for bespoke and custom, and third-party software (for example operating systems and databases) are covered.
- 6.3.2 - An inventory of bespoke and custom software, and third-party software components incorporated into bespoke and custom software is maintained to facilitate vulnerability and patch management.
- 6.3.3 - All system components are protected from known vulnerabilities by installing applicable security patches/updates as follows:
  - Critical or high-security patches/updates (identified according to the risk ranking process at Requirement 6.3.1) are installed within one month of release.

- All other applicable security patches/updates are installed within an appropriate time frame as determined by the entity (for example, within three months of release).
- 6.5.6 - Test data and test accounts are removed from system components before the system goes into production.

CENTER FOR INTERNET SECURITY CRITICAL SECURITY CONTROLS (CIS CSC)

The following SDP-related requirements are from CIS v8 include:

- 16.1 - Establish and Maintain a Secure Application Development Process
- 16.2 - Establish and Maintain a Process to Accept and Address Software Vulnerabilities
- 16.7 - Use Standard Hardening Configuration Templates for Application Infrastructure
- 16.9 - Train Developers in Application Security Concepts and Secure Coding
- 16.10 - Apply Secure Design Principles in Application Architectures
- 16.11 - Leverage Vetted Modules or Services for Application Security Components
- 16.12 - Implement Code-Level Security Checks
- 16.13 - Conduct Application Penetration Testing
- 16.14 - Conduct Threat Modeling

ISO 27002:2022

The following SDP-related requirements are from ISO 27002:2022 include:

- 8.24 - Use of cryptography
- 8.25 - Secure development life cycle
- 8.26 - Application security requirements
- 8.27 - Secure system architecture and engineering principles
- 8.28 - Secure coding
- 8.29 - Security testing in development and acceptance
- 8.30 - Outsourced development
- 8.31 - Separation of development, test and production environments
- 8.32 - Change Management
- 8.33 - Test information
- 8.34 - Protection of information systems during audit testing

DIGITAL MILLENNIUM COPYRIGHT ACT (DMCA)

The following SDP-related requirements are from DCMA include:[17]

- Section 512. Limitations on liability relating to material online.
  - Service providers must comply with the requirements of the notice-and-takedown system in order to qualify for the safe harbors.
- Section 1201. Circumvention of copyright protection systems.
  - Prohibits circumventing Technological Protection Measures (or TPMs) used by copyright owners to control access to their works.
  - Prohibits manufacturing, importing, offering to the public, providing, or otherwise trafficking in certain circumvention technologies, products, services, devices, or components.
- Section 1202. Integrity of copyright management information.
  - Prohibits providing or distributing false copyright management information (CMI) with the intent to induce or conceal infringement.

[17] *DCMA - https://www.congress.gov/105/plaws/publ304/PLAW-105publ304.pdf*

PRACTITIONER-LEVEL: UNDERSTANDING THE ROLE OF SECURITY MECHANISMS

One important facet of cybersecurity and data protection that the Developing Security & Privacy by Design (DSPD) initiative focuses on is educating applicants on the holistic concepts of how broader business planning and analysis ultimately leads to actionable cybersecurity and privacy requirements. Understanding this hierarchical nature of requirements is a fundamental construct of governance processes.

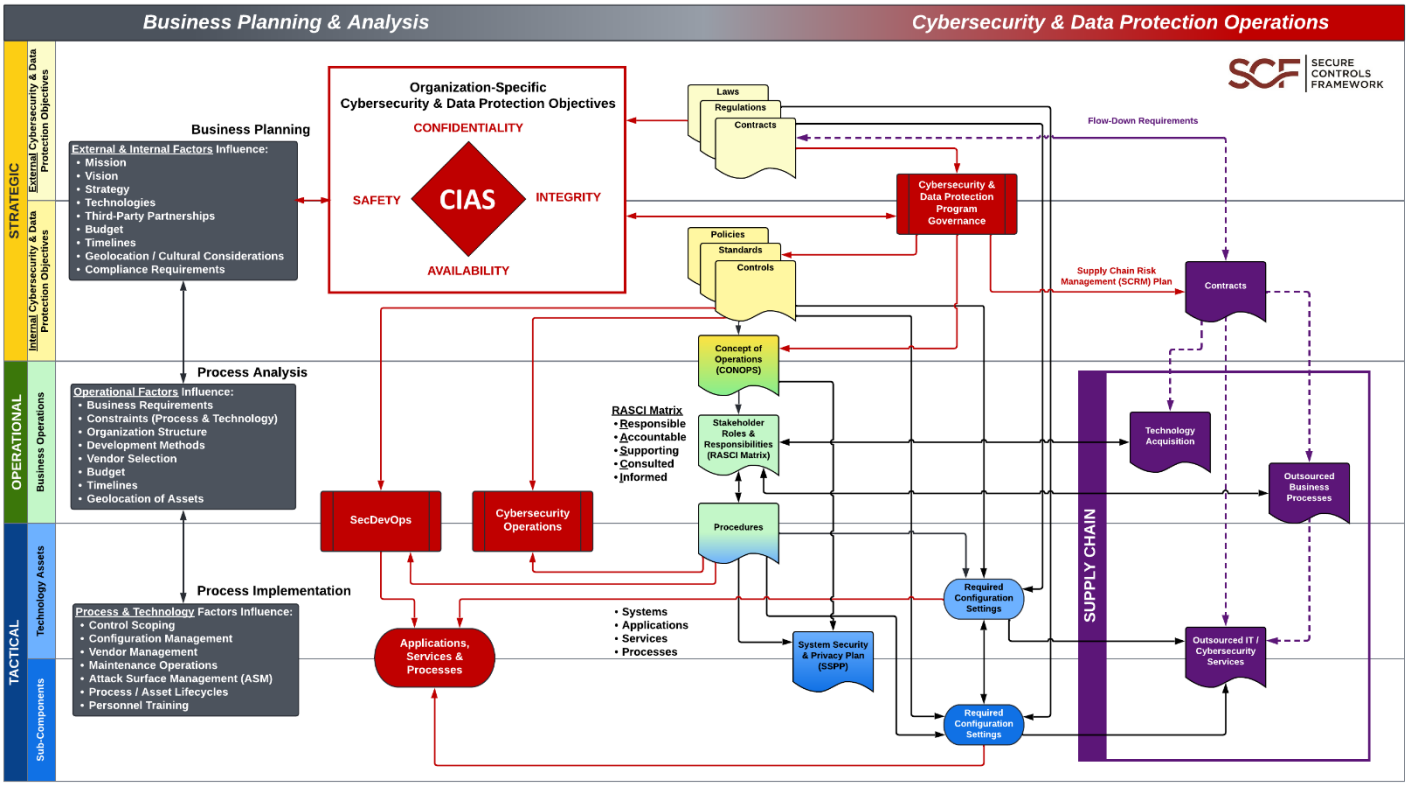
ADEQUATE SECURITY

No technology can provide absolute security due to the limits of human certainty, the uncertainty that exists in the life cycle of every system and the constraints of cost, schedule, performance, feasibility and practicality. Therefore, trade-offs are expected to be routinely made across contradictory, competing and conflicting needs and constraints. However, these trade-offs must be optimized to achieve "adequate security" which reflects a risk-based decision made by stakeholders. [18]

An organization publishes policies to eliminate potential gaps in that desired governed behavior in an attempt to achieve "adequate security" for the organization based on what a reasonable individual would be expected to do in a similar situation. The rules associated with this "governed behavior" must be accurate, consistent, compatible and complete with respect to the executive leadership's objectives to successfully accomplish the organization's mission and overall strategy.

An organization's policies ultimately define the behavior of Individual Contributors (IC) (e.g., developers) in performing their roles and associated responsibilities, as well as for the development of processes and procedures. This eventually leads to the configuration of technology assets (e.g., systems, applications, services and processes), where a discrete set of restrictions and properties must exist to specify how that asset enforces or contributes to the enforcement of the organizational security policies. This concept is depicted in the graphic shown below: [19]

Hierarchical Methodology To Determine "Adequate Security" To Build Secure, Resilient & Compliant Systems and Processes



[18] *NIST SP 800-160 Vol 1 Rev 1 Appendix C*
[19] *SCF Adequate Security Determination Process - https://securecontrolsframework.com/content/adequate-cybersecurity-methodology.pdf*

The required configuration settings for technology assets must be inclusive of technical and business requirements, which ultimately fall under organizational cybersecurity and privacy policies. Requirements can be categorized as: [20]

- Stakeholder requirements that address the need to be satisfied in a design-independent manner; and
- System requirements that express the specific solution that will be delivered (design-dependent manner).

Secure Systems

A "secure system" is a system that ensures that only the authorized intended behaviors and outcomes occur, thereby providing freedom from those conditions, both intentionally/with malice and unintentionally/without malice, that can cause a loss of information assets with unacceptable consequences.[21] This definition expresses an ideal that captures three essential aspects of what it means to achieve security:

1. Enable the delivery of the required system capability despite intentional and unintentional forms of adversity;
2. Enforce constraints to ensure that only the desired behaviors and outcomes associated with the required system capability are realized while satisfying the first aspect; and
3. Enforce constraints based on a set of rules to ensure that only authorized human-to-machine and machine-to-machine interactions and operations are allowed to occur while satisfying the second aspect.

For a system, adequate security is an evidence-based determination that achieves and optimizes security performance against all other performance objectives and constraints. Judgments of adequate security are driven by the stakeholder objectives, needs and concerns associated with the system. Adequate security has two elements:

- Achieve the minimum acceptable threshold of security performance; and
- Maximize security performance to the extent that any additional increase in security performance results in a degradation of some other aspect of system performance or requires an unacceptable operational commitment.

Stakeholder Security Requirements

Stakeholder security requirements are those stakeholder requirements that are security-relevant. Stakeholder security requirements specify:

- The protection needed for the mission or business, data, information, processes, functions, humans and system assets;
- The roles, responsibilities and security-relevant actions of individuals who perform and support the mission or business processes;
- The interactions between the security-relevant solution elements; and
- The assurance that is to be obtained in the security solution.

System Security Requirements

System requirements specify the technical view of a system or solution that meets the specified stakeholder needs. The system requirements are a transformation of the validated stakeholder requirements. System requirements specify what the system or solution must do to satisfy the stakeholder requirements. System security requirements are those system requirements that are security relevant. These requirements define:

- The protection capabilities provided by the security solution;
- The performance and behavioral characteristics exhibited by the security solution;
- Assurance processes, procedures and techniques;
- Constraints on the system and the processes, methods and tools used to realize the system; and
- The evidence required to determine the system security requirements have been satisfied.

System of Systems Mindset

A system is *"an arrangement of parts or elements that together exhibit a behavior or meaning that the individual constituents do not."*[22] Since developers do not design, code and maintain Applications, Services and Processes (ASP) in a vacuum, developers need to embrace a "system of systems" mindset toward system interaction, since there are legitimate security and privacy concerns with untrustworthy dependencies. A system of systems is *"set of systems or system elements that interact to provide a unique capability that none of the constituent systems can accomplish on its own.*[23] A system of systems consists of a number

[20] *NIST SP 800-160 Vol 1 Rev 1 Appendix C*
[21] *NIST SP 800-160 Vol 1 Rev 1 – Section 3.2*
[22] *NIST SP 800-160 Vol 1 Rev 1 – Section 2.1.1*
[23] *NIST SP 800-160 Vol 1 Rev 1 – Appendix B. Glossary*

of constituent systems plus any inter-system infrastructure, facilities and processes necessary to enable the constituent systems to integrate or interoperate.

This concept includes "interfacing systems" that have an interface for exchanging data or information, energy, or other resources. Interfacing systems have two specific subsets:

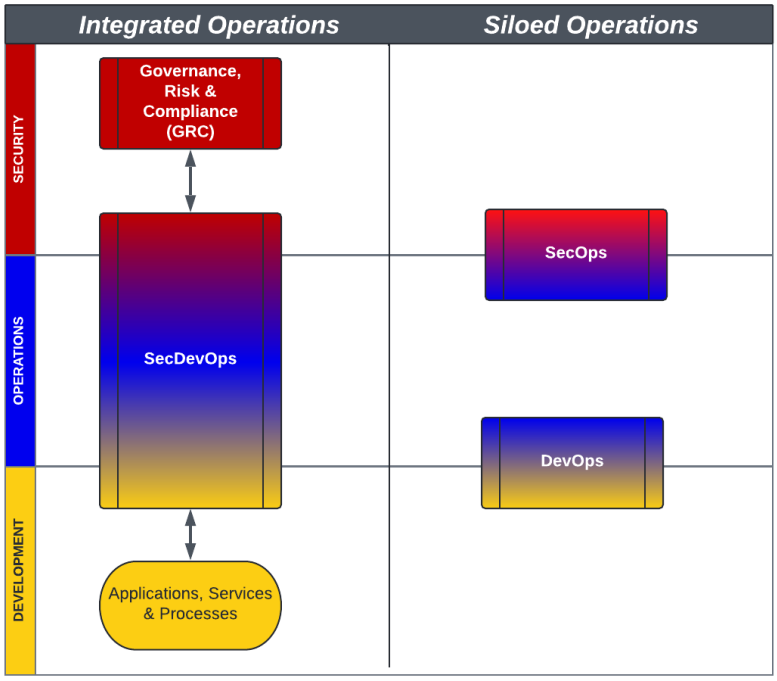
- Enabling Systems. These provide essential services required to create and sustain the system. Examples of enabling systems include:
  - Software development environments;
  - Production systems;
  - Training systems; and
  - Maintenance systems; and
- Interoperating Systems. These interact with systems for the purpose of jointly performing a function during the utilization and sustainment stages of the system life cycle. Interoperating systems often form a system of systems.

SecDevOps

Security, Development & Operations (SecDevOps), also commonly called DevSecOps, is a relationship model that integrates security in development and operational practices throughout the SDLC. SecDevOps is a "three legs of the stool" concept where each leg is important and the stool falls over if one or more leg is missing. There is considerable material that exists on the subject of SecDevOps, so the SCA-BoK will focus only on a few of its concepts.[24]

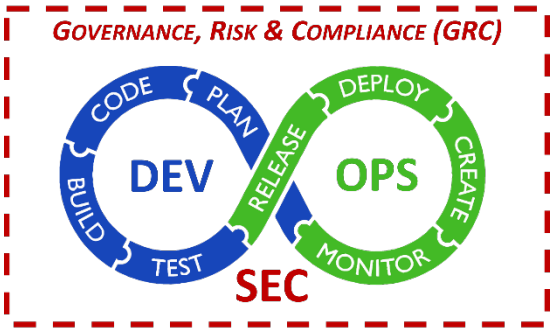
Avoiding Siloes

SecDevOps merges security, development and operations capabilities so these specialized functions can work in unison to achieve a common goal, which is rapid and secure development. This process requires organizational leadership to properly resource integrations and decision making capabilities, but its benefits outweigh the drawbacks that exist with siloed operations.



[24] DevOps.com - https://devops.com/category/blogs/devsecops/

GRC Frames SecDevOps Controls



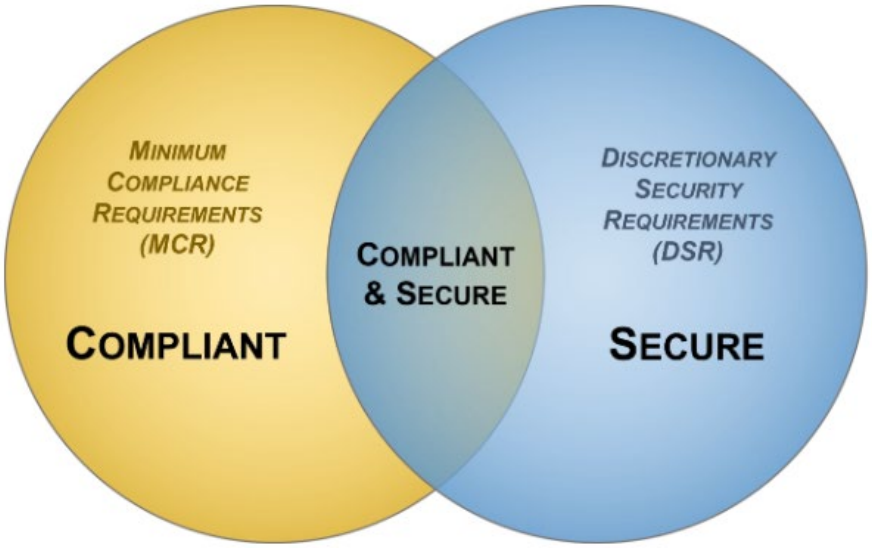
Just as the concept SecDevOps is to integrate functions to eliminate siloed operations, it is important to understand that SecDevOps does not exist within a vacuum. The "security" component of SecDevOps is constrained by the organization's overall Governance, Risk & Compliance (GRC) function. GRC exists to align people, processes and technology with the organization's business objectives, while managing risk and meeting statutory, regulatory and contractual compliance requirements. The security feed of SecDevOps is constrained by the controls GRC deems appropriate.

Compliant vs Secure

Both practitioners and architects need to understand the difference between "compliant" versus "secure" since that is necessary to have coherent risk management discussions. When evaluating "what looks right" for security controls that are applicable to an application, service or process, this involves not only the GRC team, but the business stakeholders to properly identify "must have" vs "nice to have" security and privacy requirements:[25]

- **Minimum Compliance Requirements (MCR)**
  - These are the absolute minimum requirements that must be addressed to comply with applicable laws, regulations and contracts.
  - MCR are primarily externally-influenced, based on industry, government, state and local regulations.
  - MCR should never imply adequacy for secure practices and data protection, since they are merely compliance-related.
- **Discretionary Security Requirements (DSR)**
  - These are tied to the organization's risk appetite since DSR are "above and beyond" MCR, where the organization self-identifies additional cybersecurity and data protection controls to address voluntary industry practices or internal requirements, such as findings from internal audits or risk assessments.
  - DSR are primarily internally-influenced, based on the organization's respective industry and risk tolerance.
  - While MCR establish the foundational floor that must be adhered to, DSR are where organizations often achieve improved efficiency, automation and enhanced security.

The combination of MCR and DSR identify Minimum Viable Product (MVP) security and privacy requirements. This can also be considered Minimum Security Requirements (MSR) for the application, service or process throughout its lifecycle.



Developers and architects should strive for a set of security and privacy controls that equates to "secure and compliant" instead of just "compliant" since meeting minimum compliance requirements rarely means an application, service or process will be secure.

[25] *Integrated Controls Model (ICM) - https://complianceforge.com/content/pdf/complianceforge-integrated-controls-management.pdf*

PRACTITIONER-LEVEL: SECURE SOFTWARE DEVELOPMENT PRACTICES (SSDP)

The SCA promotes a principle-based approach to Secure Software Development Practices. The most important concept to understand is to avoid assumptions anywhere along the SDLC. As a developer, it is your obligation to eliminate assumptions to ensure Secure Software Development Practices (SSDP) are properly implemented.

The SSDP are based on the Secure Software Development Framework (SSDF) that provides developers with fundamental, sound and secure recommended practices based on established secure software development practice documents. The practices are organized into four (4) domains:

1. Prepare the Organization (PO);
2. Protect the Software (PS);
3. Produce Well-Secured Software (PW); and
4. Respond to Vulnerabilities (RV).

DOMAIN 1: PREPARE THE ORGANIZATION (PO)

Organizations should ensure that their people, processes and technology are prepared to perform secure software development at the organization level. Many organizations will find some PO practices to also be applicable to subsets of their software development, like individual development groups or projects.

Practice PO.1: Define Security Requirements for Software Development

Practice: Ensure that security requirements for software development are known at all times so that they can be taken into account throughout the SDLC and duplication of effort can be minimized because the requirements information can be collected once and shared. This includes requirements from internal sources (e.g., the organization's policies, business objectives and risk management strategy) and external sources (e.g., applicable laws and regulations). [26]

SSDP Task PO.1.1

Task: Identify and document all security requirements for the organization's software development infrastructures and processes and maintain the requirements over time. [27]

SSDP Task PO.1.2

Task: Identify and document all security requirements for organization-developed software to meet and maintain the requirements over time. [28]

SSDP Task PO.1.3

Task: Communicate requirements to all third parties who will provide commercial software components to the organization for reuse by the organization's own software. [29]

Practice PO.2: Implement Roles and Responsibilities

Practice: Ensure that everyone inside and outside of the organization involved in the SDLC is prepared to perform their SDLC-related roles and responsibilities throughout the SDLC. [30]

SSDP Task PO.2.1

Task: Create new roles and alter responsibilities for existing roles as needed to encompass all parts of the SDLC. Periodically review and maintain the defined roles and responsibilities, updating them as needed. [31]

[26] *NIST SP 800-218. Table 1. Practice PO.1*
[27] *NIST SP 800-218. Table 1. Task PO.1.1 | OWASP A04:2021*
[28] *NIST SP 800-218. Table 1. Task PO.1.2 | OWASP A04:2021*
[29] *NIST SP 800-218. Table 1. Task PO.1.3 | OWASP A04:2021*
[30] *NIST SP 800-218. Table 1. Practice PO.2*
[31] *NIST SP 800-218. Table 1. Task PO.2.1 | OWASP A04:2021*

SSDP Task PO.2.2
Task: Provide role-based training for all personnel with responsibilities that contribute to secure development. Periodically review personnel proficiency and role-based training and update the training as needed. [32]

SSDP Task PO.2.3
Task: Obtain upper management or authorizing official commitment to secure development and convey that commitment to all with development-related roles and responsibilities. [33]

Practice PO.3: Implement Supporting Toolchains
Practice: Use automation to reduce human effort and improve the accuracy, reproducibility, usability and comprehensiveness of security practices throughout the SDLC, as well as provide a way to document and demonstrate the use of these practices. Toolchains and tools may be used at different levels of the organization, such as organization-wide or project-specific and may address a particular part of the SDLC, like a build pipeline. [34]

SSDP Task PO.3.1
Task: Specify which tools or tool types must or should be included in each toolchain to mitigate identified risks, as well as how the toolchain components are to be integrated with each other. [35]

SSDP Task PO.3.2
Task: Follow recommended security practices to deploy, operate and maintain tools and toolchains. [36]

SSDP Task PO.3.3
Task: Configure tools to generate artifacts of their support of secure software development practices as defined by the organization. [37]

Practice PO.4: Define and Use Criteria for Software Security Checks
Practice: Help ensure that the software resulting from the SDLC meets the organization's expectations by defining and using criteria for checking the software's security during development. [38]

SSDP Task PO.4.1
Task: Define criteria for software security checks and track throughout the SDLC. [39]

SSDP Task PO.4.2
Task: Implement processes, mechanisms, etc. to gather and safeguard the necessary information in support of the criteria. [40]

Practice PO.5: Implement and Maintain Secure Environments for Software Development
Practice: Ensure that all components of the environments for software development are strongly protected from internal and external threats to prevent compromises of the environments or the software being developed or maintained within them. Examples of environments for software development include development, build, test and distribution environments. [41]

SSDP Task PO.5.1
Task: Separate and protect each environment involved in software development. [42]

---

[32] *NIST SP 800-218. Table 1. Task PO.2.2 | OWASP A04:2021*
[33] *NIST SP 800-218. Table 1. Task PO.2.3 | OWASP A04:2021*
[34] *NIST SP 800-218. Table 1. Practice PO.3*
[35] *NIST SP 800-218. Table 1. Task PO.3.1 | OWASP A06:2021 & A08:2021*
[36] *NIST SP 800-218. Table 1. Task PO.3.2 | OWASP A06:2021 & A08:2021*
[37] *NIST SP 800-218. Table 1. Task PO.3.3 | OWASP A06:2021 & A08:2021*
[38] *NIST SP 800-218. Table 1. Practice PO.4*
[39] *NIST SP 800-218. Table 1. Task PO.4.1 | OWASP A04:2021*
[40] *NIST SP 800-218. Table 1. Task PO.4.2 | OWASP A04:2021*
[41] *NIST SP 800-218. Table 1. Practice PO.5*
[42] *NIST SP 800-218. Table 1. Task PO.5.1 | OWASP A04:2021*

SSDP Task PO.5.2

Task: Secure and harden development endpoints (e.g., endpoints for software designers, developers, testers, builders, etc.) to perform development-related tasks using a risk-based approach. [43]

DOMAIN 2: PROTECT SOFTWARE (PS)

Organizations should protect all components of their software from tampering and unauthorized access.

Practice PS.1: Protect All Forms of Code from Unauthorized Access and Tampering

Practice: Help prevent unauthorized changes to code, both inadvertent and intentional, which could circumvent or negate the intended security characteristics of the software. For code that is not intended to be publicly accessible, this helps prevent theft of the software and may make it more difficult or time-consuming for attackers to find vulnerabilities in the software. [44]

SSDP Task PS.1.1

Task: Store all forms of code – including source code, executable code and configuration-as-code – based on the principle of least privilege so that only authorized personnel, tools, services, etc. have access. [45]

Practice PS.2: Provide a Mechanism for Verifying Software Release Integrity

Practice: Help software acquirers ensure that the software they acquire is legitimate and has not been tampered with. [46]

SSDP Task PS.2.1

Task: Make software integrity verification information available to software acquirers. [47]

Practice PS.3: Archive and Protect Each Software Release

Practice: Preserve software releases in order to help identify, analyze and eliminate vulnerabilities discovered in the software after release. [48]

SSDP Task PS.3.1

Task: Securely archive the necessary files and supporting data (e.g., integrity verification information, provenance data) to be retained for each software release. [49]

SSDP Task PS.3.2

Task: Collect, safeguard, maintain and share provenance data for all components of each software release (e.g., in a Software Bill of Materials (SBOM). [50]

DOMAIN 3: PRODUCE WELL-SECURED SOFTWARE (PW)

Organizations should produce well-secured software with minimal security vulnerabilities in its releases.

Practice PW.1: Design Software to Meet Security Requirements and Mitigate Security Risks

Practice: Identify and evaluate the security requirements for the software; determine what security risks the software is likely to face during operation and how the software's design and architecture should mitigate those risks; and justify any cases where risk-based analysis indicates that security requirements should be relaxed or waived. Addressing security requirements and

[43] *NIST SP 800-218. Table 1. Task PO.5.2 | OWASP A04:2021*
[44] *NIST SP 800-218. Table 1. Practice PS.1*
[45] *NIST SP 800-218. Table 1. Task PS.1.1 | OWASP A01:2021, A02:2021, A04:2021, A08:2021*
[46] *NIST SP 800-218. Table 1. Practice PS.2*
[47] *NIST SP 800-218. Table 1. Task PS.2.1*
[48] *NIST SP 800-218. Table 1. Practice PS.3*
[49] *NIST SP 800-218. Table 1. Task PS.3.1*
[50] *NIST SP 800-218. Table 1. Task PS.3.2*

risks during software design (secure by design) is key for improving software security and also helps improve development efficiency. [51]

SSDP Task PW.1.1

Task: Use forms of risk modelling – such as threat modelling, attack modelling, or attack surface mapping – to help assess the security risk for the software. [52]

SSDP Task PW.1.2

Task: Track and maintain the software's security requirements, risks and design decisions. [53]

SSDP Task PW.1.3

Task: Where appropriate, build in support for using standardized security features and services (e.g., enabling software to integrate with existing log management, identity management, access control and vulnerability management systems) instead of creating proprietary implementations of security features and services. [54]

Practice PW.2: Review the Software Design to Verify Compliance with Security Requirements and Risk Information

Practice: Help ensure that the software will meet the security requirements and satisfactorily address the identified risk information. [55]

SSDP Task PW.2.1

Task: Have 1) a qualified person (or people) who were not involved with the design; and/or 2) automated processes instantiated in the toolchain review the software design to confirm and enforce that it meets all of the security requirements and satisfactorily addresses the identified risk information. [56]

Practice PW.3: Verify Third-Party Software Complies with Security Requirements

Moved to PW.4 *[per NIST SP 800-218]*

- PW.3.1: Moved to PO.1.3
- PW.3.2: Moved to PW.4.4

Practice PW.4: Reuse Existing, Well-Secured Software When Feasible Instead of Duplicating Functionality

Practice: Lower the costs of software development, expedite software development and decrease the likelihood of introducing additional security vulnerabilities into the software by reusing software modules and services that have already had their security posture checked. This is particularly important for software that implements security functionality, such as cryptographic modules and protocols. [57]

SSDP Task PW.4.1

Task: Acquire and maintain well-secured software components (e.g., software libraries, modules, middleware, frameworks) from commercial, open-source and other third-party developers for use by the organization's software. [58]

SSDP Task PW.4.2

Task: Create and maintain well-secured software components in-house following SDLC processes to meet common internal software development needs that cannot be better met by third-party software components. [59]

---

[51] *NIST SP 800-218. Table 1. Practice PW.1*
[52] *NIST SP 800-218. Table 1. Task PW.1.1*
[53] *NIST SP 800-218. Table 1. Task PW.1.2| OWASP A04:2021*
[54] *NIST SP 800-218. Table 1. Task PW.1.3 | OWASP A01:2021 & A04:2021*
[55] *NIST SP 800-218. Table 1. Practice PW.2*
[56] *NIST SP 800-218. Table 1. Task PW.2.1*
[57] *NIST SP 800-218. Table 1. Practice PW.4*
[58] *NIST SP 800-218. Table 1. Task PW.4.1 | OWASP A02:2021, A06:2021, A07:2021, A08:2021 A09:2021 & A10:2021*
[59] *NIST SP 800-218. Table 1. Task PW.4.2 | OWASP A02:2021, A06:2021, A07:2021, A08:2021 A09:2021 & A10:2021*

SSDP Task PW.4.3
Moved to PW.1.3 *[per NIST SP 800-218]*

SSDP Task PW.4.4
Task: Verify that acquired commercial, open-source and all other third-party software components comply with the requirements, as defined by the organization, throughout their life cycles. [60]

SSDP Task PW.4.5
Moved to Tasks PW.4.1 and PW.4.4 *[per NIST SP 800-218]*

Practice PW.5: Create Source Code by Adhering to Secure Coding Practices

Practice: Decrease the number of security vulnerabilities in the software and reduce costs by minimizing vulnerabilities introduced during source code creation that meet or exceed organization-defined vulnerability severity criteria. [61]

SSDP Task PW.5.1
Task: Follow all secure coding practices that are appropriate to the development languages and environment to meet the organization's requirements. [62]

SSDP Task PW.5.2
Moved to PW.5.1 *[per NIST SP 800-218]*

Practice PW.6: Configure the Compilation, Interpreter and Build Processes to Improve Executable Security

Practice: Decrease the number of security vulnerabilities in the software and reduce costs by eliminating vulnerabilities before testing occurs. [63]

SSDP Task PW.6.1
Task: Use compiler, interpreter and build tools that offer features to improve executable security. [64]

SSDP Task PW.6.2
Task: Determine which compiler, interpreter and build tool features should be used and how each should be configured, then implement and use the approved configurations. [65]

Practice PW.7: Review and/or Analyze Human-Readable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements

Practice: Help identify vulnerabilities so that they can be corrected before the software is released to prevent exploitation. Using automated methods lowers the effort and resources needed to detect vulnerabilities. Human-readable code includes source code, scripts and any other form of code that an organization deems human-readable. [66]

SSDP Task PW.7.1
Task: Determine whether code review (a person looks directly at the code to find issues) and/or code analysis (tools are used to find issues in code, either in a fully automated way or in conjunction with a person) should be used, as defined by the organization. [67]

SSDP Task PW.7.2

[60] *NIST SP 800-218. Table 1. Task PW.4.4 | OWASP A02:2021, A06:2021, A07:2021, A08:2021 A09:2021 & A10:2021*
[61] *NIST SP 800-218. Table 1. Practice PW.5*
[62] *NIST SP 800-218. Table 1. Task PW.5.1*
[63] *NIST SP 800-218. Table 1. Practice PW.6*
[64] *NIST SP 800-218. Table 1. Task PW.6.1*
[65] *NIST SP 800-218. Table 1. Task PW.6.2*
[66] *NIST SP 800-218. Table 1. Practice PW.7*
[67] *NIST SP 800-218. Table 1. Task PW.7.1 | OWASP A03:2021*

Task: Perform the code review and/or code analysis based on the organization's secure coding standards and record and triage all discovered issues and recommended remediations in the development team's workflow or issue tracking system. [68]

Practice PW.8: Test Executable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements

Practice: Help identify vulnerabilities so that they can be corrected before the software is released in order to prevent exploitation. Using automated methods lowers the effort and resources needed to detect vulnerabilities and improves traceability and repeatability. Executable code includes binaries, directly executed bytecode and source code and any other form of code that an organization deems executable. [69]

SSDP Task PW.8.1

Task: Determine whether executable code testing should be performed to find vulnerabilities not identified by previous reviews, analysis, or testing and, if so, which types of testing should be used. [70]

SSDP Task PW.8.2

Task: Scope the testing, design the tests, perform the testing and document the results, including recording and triaging all discovered issues and recommended remediations in the development team's workflow or issue tracking system. [71]

Practice PW.9: Configure Software to Have Secure Settings by Default

Practice: Help improve the security of the software at the time of installation to reduce the likelihood of the software being deployed with weak security settings, putting it at greater risk of compromise. [72]

SSDP Task PW.9.1

Task: Define a secure baseline by determining how to configure each setting that has an effect on security or a security-related setting so that the default settings are secure and do not weaken the security functions provided by the platform, network infrastructure, or services. [73]

SSDP Task PW.9.2

Task: Implement the default settings (or groups of default settings, if applicable) and document each setting for software administrators. [74]

Domain 4: Respond to Vulnerabilities (RV)

Organizations should identify residual vulnerabilities in their software releases and respond appropriately to address those vulnerabilities and prevent similar ones from occurring in the future.

Practice RV.1: Identify and Confirm Vulnerabilities on an Ongoing Basis

Practice: Help ensure that vulnerabilities are identified more quickly so that they can be remediated more quickly in accordance with risk, reducing the window of opportunity for attackers. [75]

SSDP Task RV.1.1

Task: Gather information from software acquirers, users and public sources on potential vulnerabilities in the software and third-party components that the software uses and investigate all credible reports. [76]

---

[68] *NIST SP 800-218. Table 1. Task PW.7.2 | OWASP A03:2021*
[69] *NIST SP 800-218. Table 1. Practice PW.8*
[70] *NIST SP 800-218. Table 1. Task PW.8.1 | OWASP A03:2021*
[71] *NIST SP 800-218. Table 1. Task PW.8.2 | OWASP A03:2021*
[72] *NIST SP 800-218. Table 1. Practice PW.9*
[73] *NIST SP 800-218. Table 1. Task PW.9.1 | OWASP A04:2021 & A05:2021*
[74] *NIST SP 800-218. Table 1. Task PW.9.2 | OWASP A04:2021 & A05:2021*
[75] *NIST SP 800-218. Table 1. Practice RV.1*
[76] *NIST SP 800-218. Table 1. Task RV.1.1*

SSDP Task RV.1.2
Task: Review, analyze, and/or test the software's code to identify or confirm the presence of previously undetected vulnerabilities. [77]

SSDP Task RV.1.3
Task: Have a policy that addresses vulnerability disclosure and remediation and implement the roles, responsibilities and processes needed to support that policy. [78]

Practice RV.2: Assess, Prioritize and Remediate Vulnerabilities
Practice: Help ensure that vulnerabilities are remediated in accordance with risk to reduce the window of opportunity for attackers. [79]

SSDP Task RV.2.1
Task: Analyze each vulnerability to gather sufficient information about risk to plan its remediation or other risk response. [80]

SSDP Task RV.2.2
Task: Plan and implement risk responses for vulnerabilities. [81]

Practice RV.3: Analyze Vulnerabilities to Identify Their Root Causes
Practice: Help reduce the frequency of vulnerabilities in the future. [82]

SSDP Task RV.3.1
Task: Analyze identified vulnerabilities to determine their root causes. [83]

SSDP Task RV.3.2
Task: Analyze the root causes over time to identify patterns, such as a particular secure coding practice not being followed consistently. [84]

SSDP Task RV.3.3
Task: Review the software for similar vulnerabilities to eradicate a class of vulnerabilities and proactively fix them rather than waiting for external reports. [85]

SSDP Task RV.3.4
Task: Review the SDLC process and update it if appropriate to prevent (or reduce the likelihood of) the root cause recurring in updates to the software or in new software that is created. [86]

[77] *NIST SP 800-218. Table 1. Task RV.1.2 | OWASP A05:2021, A06:2021*
[78] *NIST SP 800-218. Table 1. Task RV.1.3*
[79] *NIST SP 800-218. Table 1. Practice RV.2*
[80] *NIST SP 800-218. Table 1. Task RV.2.1*
[81] *NIST SP 800-218. Table 1. Task RV.2.2*
[82] *NIST SP 800-218. Table 1. Practice RV.3*
[83] *NIST SP 800-218. Table 1. Task RV.3.1*
[84] *NIST SP 800-218. Table 1. Task RV.3.2*
[85] *NIST SP 800-218. Table 1. Task RV.3.3*
[86] *NIST SP 800-218. Table 1. Task RV.3.4*

PRACTITIONER-LEVEL: TRUSTWORTHY SECURE DESIGN PRINCIPLES & CONCEPTS

Key Security Objective: An important objective for security is the reduction of uncertainty regarding the occurrence and effects of adverse events. Reducing the uncertainty of adverse events is achieved by eliminating hazards, susceptibility and vulnerability to the extent possible. Where elimination cannot occur, their effects should be controlled to the extent possible. Applying the design principles for trustworthy secure systems is a means of achieving the elimination and control of the hazards, susceptibility and vulnerability that lead to adverse events. [87]

The SCA promotes a principle-based approach to Secure Software Development Practices. The most important concept to understand is to avoid assumptions anywhere along the SDLC. As a developer, it is your obligation to eliminate assumptions to ensure Secure Software Development Practices (SSDP) are properly implemented.

NIST SP 800-160, Vol 1, Rev 1 [88] contains a set of principles that serve as the foundation for engineering trustworthy secure systems. The principles for trustworthy secure design are applied to control the adversity that might occur as a direct or indirect result of the system delivering a specified capability at a specified level of performance. The principles represent research, development and application experience starting with the early incorporation of security mechanisms for trusted operating systems to today's fully networked, distributed, mobile and virtual computing components, environments and systems. The principles are intended to be universally applicable across this broad range of systems, as well as new systems as they emerge and mature.

The principles for trustworthy secure design provide a basis for reasoning about a system. As reasoning tools, the inherent suitability of the principles in a particular situation will depend on the judgment of the practitioner. Engineering judgment must be exercised in the application of the principles for trustworthy secure systems. The principles should not be applied as "rules" to be complied with, nor should they be prioritized, sequenced, or ordered for prescriptive application, or used individually or in groups as a basis for making judgments of conformance. Principles are subject to various priorities and constraints that may restrict or preclude their application. At times, these principles may be in conflict with other principles and must be deconflicted. In practice, the principles can be satisfied or implemented in various and perhaps equally effective ways. Within the system life cycle, the applicability of a particular principle may change due to evolving requirements, protection needs, priorities, or constraints; architecture and design decisions and trade-offs; or changes in the risk acceptance threshold.

APPLICATION OF DESIGN PRINCIPLES TO COMMERCIAL PRODUCTS

For commercial products to be trustworthy commensurate with their criticality, security design principles should be selected and applied appropriately throughout the products' system life cycle. Each design principle must be assessed for its relevance, applicability and validity.

Many commercial products have been designed, developed and evaluated against specifications from those standards and guidelines up to and including the highest levels of assurance (e.g., TCSEC Class A1 and Class B3). These products represent use cases of trustworthy components and systems that have been verified to be highly resistant to penetration from determined adversaries. To merit the trust of consumers, commercial products must demonstrate – in a manner that can be independently verified – that the security design principles articulated in this section have been applied to produce components and systems that are both sound and logically coherent with respect to security.

TRUSTWORTHINESS DESIGN PRINCIPLES

Trustworthiness design principles are based on the historical meaning of trustworthiness and trust and their use as the basis for the design of secure systems. The terms trustworthiness and trust as follows: [89]

- Trustworthiness: The demonstrated worthiness of an entity to be trusted based on evidence that supports a claim or judgment of being trustworthy.
- Trust: A belief that an entity can be trusted. (Implies that trust may be granted to an entity whether the entity is trustworthy or not).

Trustworthiness is a cross-cutting objective in the design of systems due to the consequences of the failure of systems to behave and produce outcomes only as authorized and intended. The terms trust and trusted are used to mean "the decision is

[87] *NIST SP 800-160 Vol 1 Rev 1 – Appendix E*
[88] *NIST SP 800-160 Vol 1 Rev 1*
[89] *NIST SP 800-160 Vol 1 Rev 1 – Section 2.3*

made to trust because the required trustworthiness is demonstrated." Trustworthiness is associated with one of the essential design criteria and the reference monitor concept.[90] A protection mechanism or feature must be evaluable (e.g., the mechanism must be sufficiently small and simple enough to be assessed to produce adequate confidence in the protection provided, the constraint or control objective enforced and the correct implementation of the mechanism).

Trustworthiness design principles are fundamental to managing complexity and otherwise aid in understanding the engineered system. The principles are necessary to achieve loss control objectives given the complexity in understanding loss in context (based on how the system is intended to be utilized and sustained). Complexity increases analysis workloads and reduces confidence in that analysis. Complexity also increases the costs and difficulty of performing systems analyses for loss. That is, systems may be too complex to be analyzed for adequate assurance.

There are thirty (30) Trustworthy Secure Design (TSD) principles identified in NIST SP 800-160 Vol 1 Rev 1: [91]

TSD-1: Anomaly Detection

Principle: Any salient anomaly in the system or its environment is detected in a timely manner that enables effective response action.[92]

TSD-2: Clear Abstractions

Principle: The abstractions used to characterize the system are simple, well-defined, accurate, precise, necessary, and sufficient.[93]

TSD-3: Commensurate Protection

Principle: The strength and type of protection provided to a system element are commensurate with the most significant adverse effect that results from a failure of that element. [94]

TSD-4: Commensurate Response

Principle: The system design matches the aggressiveness of an engineered response action's effect to the needed immediacy to control the effects of each loss scenario. [95]

TSD-5: Commensurate Rigor

Principle: The rigor associated with the conduct of an engineering activity provides the confidence required to address the most significant adverse effect that can occur. [96]

TSD-6: Commensurate Trustworthiness

Principle: A system element is trustworthy to a level commensurate with the most significant adverse effect that results from a failure of that element. [97]

TSD-7: Compositional Trustworthiness

Principle: The system design is trustworthy for each aggregate composition of interacting system elements.[98]

TSD-8: Continuous Protection

Principle: The protection provided for a system element must be effective and uninterrupted during the time that the protection is required.[99]

---

[90] *NIST SP 800-160 Vol 1 Rev 1 – Section D.4.2*
[91] *NIST SP 800-160 Vol 1 Rev 1 Appendix E*
[92] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.1*
[93] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.2*
[94] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.3*
[95] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.4*
[96] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.5*
[97] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.6*
[98] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.7*
[99] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.8*

TSD-9: Defense In Depth
Principle: Loss is prevented or minimized by employing multiple coordinated mechanisms. [100]

TSD-10: Distributed Privilege
Principle: Multiple authorized entities act in a coordinated manner before an operation on the system is allowed to occur. [101]

TSD-11: Diversity (Dynamicity)
Principle: The system design delivers the required capability through structural, behavioral, or data or control flow variation. [102]

TSD-12: Domain Separation
Principle: Domains with distinctly different protection needs are physically or logically separated. [103]

TSD-13: Hierarchical Protection
Principle: A system element need not be protected from more trustworthy elements. [104]

TSD-14: Least Functionality
Principle: Each system element has the capability to accomplish its required functions but no more. [105]

TSD-15: Least Persistence
Principle: System elements and other resources are available, accessible, and able to fulfill their design intent only for the time for which they are needed. [106]

TSD-16: Least Privilege
Principle: Each system element is allocated privileges that are necessary to accomplish its specified functions but no more. [107]

TSD-17: Least Sharing
Principle: System resources are shared among system elements only when necessary and among as few elements as possible. [108]

TSD-18: Loss Margins
Principle: The system is designed to operate in a state space sufficiently distanced from the threshold at which loss occurs. [109]

TSD-19: Mediated Access
Principle: All access to and operations on system elements are mediated. [110]

TSD-20: Minimal Trusted Elements
Principle: A system has as few trusted system elements as practicable. [111]

TSD-21: Minimize Detectability
Principle: The design of the system minimizes the detectability of the system as much as practicable. [112]

---

[100] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.9*
[101] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.10*
[102] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.11*
[103] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.12*
[104] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.13*
[105] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.14*
[106] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.15*
[107] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.16*
[108] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.17*
[109] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.18*
[110] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.19*
[111] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.20*
[112] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.21 | OWASP A01:2021*

TSD-22: Protective Defaults
Principle: The default configuration of the system provides maximum protection effectiveness. [113]

TSD-23: Protective Failure
Principle: A failure of a system element neither results in an unacceptable loss nor invokes another loss scenario. [114]

TSD-24: Protective Recovery
Principle: The recovery of a system element does not result in nor lead to unacceptable loss. [115]

TSD-25: Reduced Complexity
Principle: The system design is as simple as practicable. [116]

TSD-26: Redundancy
Principle: The system design delivers the required capability by replication of system functions or elements. [117]

TSD-27: Self-Reliant Trustworthiness
Principle: The trustworthiness of a system element is achieved with minimal dependence on other elements. [118]

TSD-28: Structured Decomposition and Composition
Principle: System complexity is managed through the structured decomposition of the system and the structured composition of the constituent elements to deliver the required capability. [119]

TSD-29: Substantiated Trustworthiness
Principle: System trustworthiness judgments are based on evidence that the criteria for trustworthiness have been satisfied. [120]

TSD-30: Trustworthy System Control
Principle: The design for system control functions conforms to the properties of the generalized reference monitor. [121]

[113] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.22 | OWASP A01:2021*
[114] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.23 | OWASP A01:2021*
[115] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.24*
[116] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.25 | OWASP A01:2021*
[117] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.26*
[118] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.27*
[119] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.28*
[120] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.29*
[121] *NIST SP 800-160 Vol 1 Rev 1 Appendix E.30*

PRACTITIONER-LEVEL: COMPLIANCE OBLIGATIONS FOR SOFTWARE SUPPLY CHAIN SECURITY (SSCS)

Executive Order (EO) 14028, *Improving the Nation's Cybersecurity*, directs the National Institute of Standards and Technology (NIST) to publish guidance on practices for software supply chain security. [122] EO 14028 Section 4e contains ten (10) subsections, each of which specifies actions or outcomes for software producers, such as Commercial-Off-The-Shelf (COTS) product vendors, Government-Off-The-Shelf (GOTS) software developers, contractors and other custom software developers.

EXECUTIVE ORDER (EO) 14028

EO 14028 emphasizes that "*the security of software used by the Federal Government is vital to the Federal Government's ability to perform its critical functions*," and "*there is a pressing need to implement more rigorous and predictable mechanisms for ensuring that products function securely and as intended*." Accordingly, Secure Software Development Practices (SSDP) should be integrated throughout software life cycles for three (3) reasons:

1. To reduce the number of vulnerabilities in released software;
2. To reduce the potential impact of the exploitation of undetected or unaddressed vulnerabilities; and
3. To address the root causes of vulnerabilities to prevent recurrences.

NIST SP 800-218, *Secure Software Development Framework (SSDF)*, defines outcome-based SSDP for software producers to follow. [123] The following chart contains a mapping for each Section 4e item to the SSDF practices:

SSDF Practices Corresponding to EO 14028 Subsections

EO 14028 Subsection	Subsection Summary	SSDF Practice and Task Reference Numbers
4e(i)	Have secure software development environments, including:	[See rows below]
4e(i)(A)	administratively separate build environments;	PO.5.1
4e(i)(B)	trust relationship auditing;	PO.5.1
4e(i)(C)	multi-factor, risk-based authentication and conditional access;	PO.5.1, PO.5.2
4e(i)(D)	minimized dependencies on enterprise products in development environments;	PO.5.1
4e(i)(E)	data encryption; and	PO.5.2
4e(i)(F)	operational monitoring and incident detection and response.	PO.3.2, PO.3.3, PO.5.1, PO.5.2
4e(ii)	Provide artifacts from 4e(i) upon request.	PO.3.2, PO.3.3, PO.5.1, PO.5.2
4e(iii)	Maintain trusted source code supply chains.	PO.3.1, PO.3.2, PO.5.1, PO.5.2, PS.1.1, PS.2.1, PS.3.1, PW.4.1, PW.4.4
4e(iv)	Check software for vulnerabilities and remediate them.	PO.4.1, PO.4.2, PS.1.1, PW.2.1, PW.4.4, PW.5.1, PW.6.1, PW.6.2, PW.7.1, PW.7.2, PW.8.2, PW.9.1, PW.9.2, RV.1.1, RV.1.2, RV.2.1, RV.2.2, RV.3.3
4e(v)	Provide artifacts from 4e(iii) and 4e(iv) upon request and make a summary description of risks assessed and mitigated publicly available.	PO.3.2, PO.3.3, PO.4.1, PO.4.2, PO.5.1, PO.5.2, PW.1.2, PW.2.1, PW.7.2, PW.8.2, RV.2.2
4e(vi)	Maintain provenance data for internal and 3rd party components.	PO.1.3, PO.3.2, PO.5.1, PO.5.2, PS.3.1, PS.3.2, PW.4.1, PW.4.4, RV.1.1, RV.1.2
4e(vii)	Provide a software bill of materials (SBOM) for each product.	PS.3.2
4e(viii)	Participate in a vulnerability disclosure program.	RV.1.1, RV.1.2, RV.1.3, RV.2.1, RV.2.2, RV.3.3
4e(ix)	Attest to conformity with secure software development practices.	All practices and tasks consistent with a risk-based approach
4e(x)	Attest to the integrity and provenance of open- source software components.	PS.2.1, PS.3.1, PS.3.2, PW.4.1, PW.4.4

[122] *Executive Order (EO) 14028, Improving the Nation's Cybersecurity* - https://www.federalregister.gov/documents/2021/05/17/2021-10460/improving-the-nations-cybersecurity
[123] *NIST SP 800-218 v1.1* - https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf

Software Producer Obligations

NIST SP 800-218 addresses EO 14028 Section 4e from a software producer viewpoint. The software producers are the ones who implement SSDF practices. EO 14028 Section 4k explains that federal agencies will need to comply with NIST guidelines. In this context, federal agencies are software purchasers, not software producers, so additional guidance from the US Government is needed to address EO 14028 Section 4e from a software purchaser viewpoint. However, when a federal agency (purchaser) acquires software or a product containing software, the agency is required to receive attestation from the software producer that the software's development complies with US Government-specified secure software development practices.

It is expected that Federal agencies will request artifacts from the software producer that support its attestation of conformity with the SSDP described in EO 14028 Section 4e subsections (i), (iii) and (iv), which are listed here:

(i) secure software development environments, including such actions as:
(A) using administratively separate build environments;
(B) auditing trust relationships;
(C) establishing multi-factor, risk-based authentication and conditional access across the enterprise;
(D) documenting and minimizing dependencies on enterprise products that are part of the environments used to develop, build and edit software;
(E) employing encryption for data; and
(F) monitoring operations and alerts and responding to attempted and actual cyber incidents;

(iii) employing automated tools, or comparable processes, to maintain trusted source code supply chains, thereby ensuring the integrity of the code;

(iv) employing automated tools, or comparable processes, that check for known and potential vulnerabilities and remediate them, which shall operate regularly, or at a minimum prior to product, version, or update release.

Software Conformity Assessment

EO 14028 Section 4e uses several terms, including "conformity," "attestation," and "artifacts" so it is important to understand the meaning of those terms according to definitions from existing standards and guidance:

- Conformity assessment is a "*demonstration that specified requirements are fulfilled*." In the context of EO 14028 Section 4e, the requirements are SSDP, so a conformity assessment is a demonstration that the software producer has followed SSDP for its software. [124]
- Attestation is the "*issue of a statement, based on a decision, that fulfilment of specified requirements has been demonstrated*." [125]
    - If the software producer self-attests that it conforms to SSDP through a Supplier's Declaration of Conformity (SDoC):
    - If the software purchaser attests to the software producer's conformity with secure software development practices, this is known as second-party attestation.
    - If an independent third-party attests to the software producer's conformity with secure software development practices, this is known as third-party attestation or certification.
- Artifacts are "*a piece of evidence*." [adapted from NISTIR 7692] Evidence is "*grounds for belief or disbelief; data on which to base proof or to establish truth or falsehood.*" [NIST SP 800-160 Vol. 1] Artifacts provide records of secure software development practices:
    - Low-level artifacts are generated manually, or by automated means, during software development and are maintained by the software producer. Low-level artifacts include, but are not limited to:
        - Threat models;
        - Log entries;
        - Source code files;
        - Source code vulnerability scan reports;
        - Testing results;
        - Telemetry; and
        - Risk-based mitigation decisions for a particular piece of software.

[124] *ISO/IEC 17000:2020 - https://www.iso.org/standard/73029.html*
[125] *ISO/IEC 17000:2020 - https://www.iso.org/standard/73029.html*

- High-level artifacts are generated by summarizing SSDP derived from the low-level artifacts. An example of a high-level artifact is a publicly-accessible document describing the methodology, procedures and processes a software producer uses its SSDP.

The following subsections of EO 14028 Section 4e use these terms:

> *(ii) generating and, when requested by a purchaser, providing artifacts that demonstrate conformance to the processes set forth in subsection (e)(i) of this section;*
>
> *(v) providing, when requested by a purchaser, artifacts of the execution of the tools and processes described in subsection (e)(iii) and (iv) of this section and making publicly available summary information on completion of these actions, to include a summary description of the risks assessed and mitigated;*
>
> *(ix) attesting to conformity with secure software development practices.*

In summary, when a Federal agency (purchaser) acquires software or a product containing software, the agency will likely require attestation from the software producer that the software's development complies with government-specified SSDP. The Federal agency might also request artifacts from the software producer that support its attestation of conformity with the SSDP described in EO 14028 Section 4e subsections (i), (iii) and (iv).

Attesting to Conformity with Secure Software Development Practices (SSDP)

NIST defined the following minimum recommendations for Federal agencies as they acquire software or a product containing software. These recommendations are intended to assist Federal agencies and software producers in communicating clearly with each other regarding secure software development artifacts, attestation and conformity:

(1) **Use the SSDF's terminology and structure to organize communications about secure software development requirements.** This enables all software producers to use the same lexicon when attesting to conformity for federal agencies. Software producers can map their secure development methodology to the SSDF's secure software development practices or associated informative references.

(2) **Require attestation to cover secure software development practices performed as part of processes and procedures throughout the software life cycle.** With the highly dynamic nature of software today, attesting to how things were and are done on an ongoing basis (processes and procedures) is typically more valuable than attesting to how things were done for a specific software release generated by one instance of those processes. This is especially true for post-release practices such as vulnerability disclosure and response, where processes might not yet have been performed for the latest release.

(3) **Accept first-party attestation of conformity with SSDF practices unless a risk-based approach determines that second or third-party attestation is required.** First-party attestation is recommended for meeting the EO 14028 requirements. This is consistent with the guidance in NIST SP 800-161 Rev. 1, which states in Section 3.1.2: *"There are a variety of acceptable validation and revalidation methods, such as requisite certifications, site visits, third-party assessment, or self-attestation. The type and rigor of the required methods should be commensurate to the criticality of the service or product being acquired and the corresponding assurance requirements."*

(4) **When requesting artifacts of conformance, request high-level artifacts.** The software producer should be able to trace the practices summarized in the high-level artifacts to the corresponding low-level artifacts that are generated by those practices. Asking for low-level artifacts for a particular software release is not recommended for meeting the requirements of EO 14028, but may be needed to meet other agency requirements. Reasons for avoiding low-level artifacts include the following:

   a. Low-level artifacts provide a snapshot in time of only a small aspect of secure software development, whereas high-level artifacts can provide a big-picture view of how secure software development is performed.
   b. Artifacts should address the needs of the audience receiving them, thus providing the necessary information in a usable format for that audience. Understanding low-level artifacts requires the agency to expend considerable technical resources and expertise in analyzing them and determining how to consider them within the context of the broader secure software development practices.
   c. Low-level artifacts often contain intellectual property or other proprietary information, or details that attackers could use for hostile purposes, so accepting low-level artifacts gives the agency additional sensitive information to protect. These minimum recommendations apply to attestation for all software within the scope of this guidance procured by federal agencies and they should be part of each agency's processes. The

recommendations are not intended to replace more stringent requirements for secure software development that federal agencies may have.

Note: These minimum practices will not be sufficient in some cases. For example, a Federal agency may need greater visibility into the practices for a particular product so that it can better understand how the product would affect the agency's cybersecurity risk.

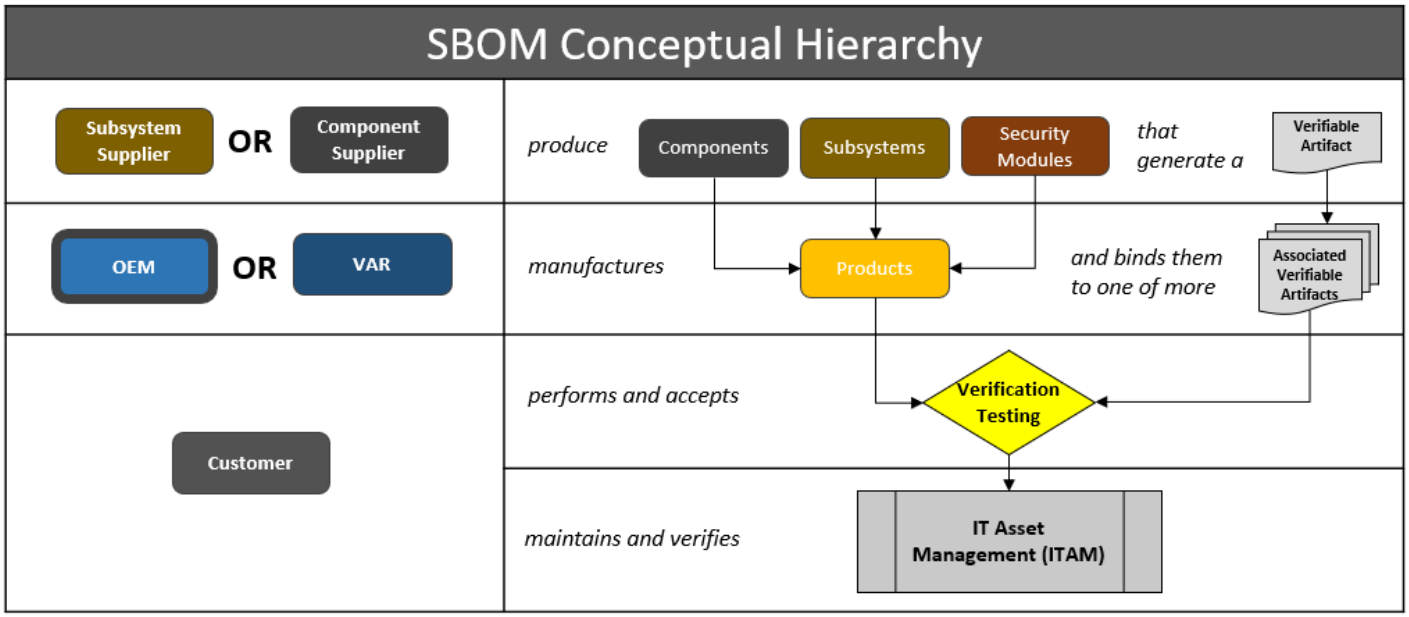
PRACTITIONER-LEVEL: SOFTWARE BILL OF MATERIALS (SBOM)

A Software Bill of Materials (SBOM) is quite simply a "nested inventory" - all the software, firmware and other components that come together to make a solution. It is a comprehensive inventory of software, subassemblies and other components, needed to create a software product.

The reason SBOMs are important is every dependency (e.g., external component) added to a software product introduces a new vector of security vulnerabilities. For example, you are developing a software product that leverages open-source components (e.g., OpenSSL). If the version of OpenSSL has known exploits, then the software product is vulnerable to those underlying OpenSSL exploits.

HIERARCHICAL NATURE OF SBOMS

The concept of hierarchy is important when visualizing a SBOM. There can be "nested SBOMs" where each layer of developers is ingesting SBOMs from its supply chain to output another SBOM. The end result is the SBOM is able to address the various components, subsystems, modules and other products that make up an end solution.



At a high-level, a SBOM is expected to help a SCA Practitioner in the following ways:

- Assist in the building and maintaining software; and
- Identify software and component dependencies to quickly implement either compensating controls or software fixes.

At a high-level, a SBOM is expected to help a SCA Architect in the following ways:

- Assist in design of software, including components that will be utilized to make it work;
- Plan implementation strategies;
- Manage licensing and compliance for software components; and
- Provide vulnerability management and asset management functions with visibility into the software that affects other stakeholders.

While there are competing standards for SBOM development, from a conceptual perspective, a SBOM is expected to have coverage for:

- An application's open-source libraries;
- Plugins, extensions and other add-ons;
- Custom source code (e.g., written by in-house or contracted developers); and
- Component versions.

SBOM LEADING PRACTICES

There are many tools available (both commercial and open-source) to help automate the process of developing and managing a SBOM more accurate and efficient.

The Cybersecurity and Infrastructure Security Agency (CISA) *Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM)* document contains guidance on SBOM components.[126]

From a tool / solution perspective:

- An SBOM system that follows the guidance and framing proposed in this document must support these Baseline Attributes.
- An SBOM system or format may support supplemental Attributes. Attributes that are unavailable, not applicable, or do not materially contribute to Component identification are discussed in Section 2.3.

SBOM Tool Requirements

CISA's model leverages three (3) attribute maturity levels. These levels describe the evolving content provided in Attribute entries as well as possible approaches to achieve increased maturity. If there are no maturity levels for the Attribute, the instructions presented are the minimum expected.

The data maturity levels and cybersecurity tooling automation support for each Attribute are intended to communicate the following guidance:

(1) Minimum Expected - This maturity level describes the minimum data elements for documenting a Primary Component and its Included Components for SBOMs globally.
(2) Recommended Practice - This maturity level describes the addition of Attribute data that supplements Component identification as well as practices for creating SBOMs.
(3) Aspirational Goal - This maturity level describes areas that creators of SBOMs can consider for documenting dynamic and/or remote Dependencies (see Appendix B for descriptions) that can be uniquely and unambiguously identified in an SBOM.

[126] *CISA Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM) - https://www.cisa.gov/sites/default/files/2024-10/SBOM%20Framing%20Software%20Component%20Transparency%202024.pdf*

ARCHITECT-LEVEL: DESIGN FOR CYBER RESILIENCY

The SCA supports the strategic cyber resiliency design principles that established by NIST SP 800-160, Vol 2, Rev 1: [127]

- Focus on common critical assets;
- Support agility and architect for adaptability;
- Reduce attack surfaces;
- Assume compromised resources; and
- Expect adversaries to evolve.

NIST SP 800-160, Vol 2, Rev 1 [128], presents a "cyber resiliency engineering framework" to aid in understanding and applying cyber resiliency in the system life cycle. Cyber resiliency engineering intends to architect, design, develop, maintain and sustain the trustworthiness of systems with the capability to anticipate, withstand, recover from and adapt to adverse conditions, stresses, attacks, or compromises that use or are enabled by cyber resources. From a risk management perspective, cyber resiliency is intended to reduce the mission, business, organizational, or sector risk of depending on cyber resources.

NIST illustrates the types of situations in which an adversary can maintain a long-term presence or persistence in a system without attacking the system via cyberspace:

- Compromising the pre-execution environment of a system through a hardware or software implant (e.g., compromise of the firmware or microcode of a system element, such as a network switch or a router that activates before initialization in the system's environment of operation). This is extremely difficult to detect and can result in compromise of the entire environment;
- Compromising the software development toolchain (e.g., compilers, linkers, interpreters, continuous integration tools, code repositories). This allows malicious code to be inserted by the adversary without modifying the source code or without the knowledge of the software developers; and
- Compromising a semiconductor product or process (e.g., maliciously altering the Hardware Description Language (HDL) of a microprocessor, a Field-Programmable Gate Array (FPGA), a Digital Signal Processor (DSP) or an Application-Specific Integrated Circuit (ASIC)).

Cyber resiliency constructs, including goals, objectives, techniques, implementation approaches and design principles, enable systems engineers to express cyber resiliency concepts and the relationships among them. The cyber resiliency constructs also relate to risk management. That relationship leads systems engineers to analyze cyber resiliency solutions in terms of potential effects on risk and on specific threat events or types of malicious cyber activities. The selection and relative priority of these cyber resiliency constructs is determined by the organization's strategy for managing the risks of depending on systems, which include cyber resources—in particular, by the organization's risk framing.

The relative priority of the cyber resiliency goals and objectives and relevance of the cyber resiliency design principles are determined by the risk management strategy of the organization that takes into consideration the concerns of, constraints on and equities of all stakeholders (including those who are not part of the organization). The risk management strategy for the organization is translated into specific interpretations and prioritizations of cyber resiliency goals and objectives, which guide and inform trade-offs among different forms of risk mitigation.

CYBER RESILIENCY CONSTRUCTS

Cyber resiliency is framed according to:

- Goals;
- Objectives; and
- Strategic Design Principles.

Goal

A high-level statement supporting (or focusing on) one aspect (e.g., anticipate, withstand, recover, adapt) in the definition of cyber resiliency.

[127] *NIST SP 800-160 Vol 2 Rev 1*
[128] *NIST SP 800-160 Vol 2 Rev 1*

Objective

A high-level statement (designed to be restated in system-specific and stakeholder-specific terms) of what a system must achieve in its operational environment and throughout its life cycle to meet stakeholder needs for mission assurance and resilient security. The objectives are more specific than goals and more relatable to threats.

Strategic Design Principles

A high-level statement that reflects an aspect of the risk management strategy that informs systems security engineering practices for an organization, mission, or system.

Strategic design principles are driven by an organization's risk management strategy and, in particular, by its risk framing. Risk framing may include assumptions about the threats the organization should be prepared for, the constraints on risk management decision-making (including which risk response alternatives are irrelevant) and organizational priorities and trade-offs.

Strategic design principles:

- Focus on common critical assets;
- Support agility and architect for adaptability;
- Reduce attack surfaces;
- Assume compromised resource; and
- Expect adversaries to evolve.

Cyber Resiliency Goals

The goals associated with cyber resiliency are:

- Anticipate;
- Withstand;
- Recover;
- Adapt;
- Understand;
- Transform; and
- Re-Architect.

These goals can be further decomposed into sub-objectives and capabilities.

Cyber Resiliency Objectives

Cyber resiliency objectives are specific statements of what a system is intended to achieve in its operational environment and throughout its life cycle to meet stakeholder needs for mission assurance and resilient security.[129] Cyber resiliency objectives:

- Support interpretation;
- Facilitate prioritization and assessment; and
- Enable development of questions such as:
    - What does each cyber resiliency objective mean in the context of the organization and the mission or business process that the system is intended to support?
    - Which cyber resiliency objectives are most important to a given stakeholder?
    - To what degree can each cyber resiliency objective be achieved?
    - How quickly and cost-effectively can each cyber resiliency objective be achieved?
    - With what degree of confidence or trust can each cyber resiliency objective be achieved?

The five (5) objectives of cyber resiliency are: [130]

(1) **Prevent or Avoid**. Preclude the successful execution of an attack or the realization of adverse conditions;
(2) **Prepare**. Maintain a set of realistic courses of action that address predicted or anticipated adversity;
(3) **Continue**. Maximize the duration and viability of essential mission or business functions during adversity;
(4) **Constrain**. Limit damage from adversity; and
(5) **Reconstitute**. Restore as much mission or business functionality as possible after adversity.

[129] *NIST SP 800-160 Vol 2 Rev 1 Appendix D*
[130] *NIST SP 800-160 Vol 2 Rev 1 Table 3*

Resilient & Secure Development Lifecycle (RSDL) Stages

The six (6) phases of the Resilient & Secure Development Lifecycle (RSDL) include:

(1) Concept;
(2) Development;
(3) Production;
(4) Utilization;
(5) Support; and
(6) Retirement.

Concept

- Prioritize and tailor objectives.
- Prioritize design principles and align with other disciplines.
- Limit the set of techniques and approaches to use in solutions.

Development

- Apply design principles to analyze and shape architecture and design.
- Use techniques and approaches to define alternative solutions.
- Develop capabilities to achieve the Prevent/Avoid, Continue, Constrain, Reconstitute and Understand objectives.

Production

- Implement and evaluate the effectiveness of cyber resiliency solutions.
- Provide resources (or ensure that resources will be provided) to achieve the Prepare objective.

Utilization

- Monitor the effectiveness of cyber resiliency solutions using capabilities to achieve Understand and Prepare objectives.
- Reprioritize and tailor objectives as needed and adapt mission, business, and/or security processes to address environmental changes (Transform objective).

Support

- Revisit the prioritization and tailoring of objectives; use the results of monitoring to identify new or modified requirements.
- Revisit constraints on techniques and approaches.
- Modify or upgrade capabilities consistent with changes as noted (Re- Architect objective).

Retirement

- Prioritize and tailor objectives for the environment of operation.
- Ensure that disposal processes enable those objectives to be achieved, modifying or upgrading capabilities of other systems as necessary (Re- Architect objective).

ARCHITECT-LEVEL: TRUSTWORTHY SECURE DESIGN (TSD)

Trustworthy Secure Design (TSD) is a means to optimally satisfy the requirements that form the basis for achieving system security objectives across competing and conflicting stakeholder capability needs, concerns and constraints. [131]

DESIGN APPROACH FOR TRUSTWORTHY SYSTEMS [132]

The design approach for engineering trustworthy secure systems is intended to establish and maintain the ability to deliver system capabilities at an acceptable level of performance while minimizing the occurrence and extent of loss. The approach provides a system structure for optimal employment of the tactical engineered features and devices.

The system design must provide the intended behaviors and outcomes, avoid the unintended behaviors and outcomes, prevent loss and limit loss when it occurs. A trustworthy secure design includes a margin and a situational awareness capability to account for the unknowns and uncertainty inherent in the system and its operational environment, as well as related adversity.

A trustworthy secure design includes a margin and a situational awareness capability to account for the unknowns and uncertainty inherent in the system and its operational environment, as well as related adversity. The design approach includes the following elements:

- Define the intended behaviors and outcomes for the system;
- Identify the system states and conditions that reflect the intended behaviors and outcomes;
- Identify the system states and conditions that potentially lead to loss in the system; and
- Engineer to prevent loss to the extent practicable (preferred) and limit the loss that does occur (where, when and to the extent necessary and practicable).

DESIGN FOR BEHAVIORS & OUTCOMES [133]

A system is to deliver the required capability at a specified level of performance. The system capability is reflected in its behaviors and outcomes. The design goal is to provide capabilities that are authorized and intended. However, the system can also deliver a capability that is not authorized or intended. This possibility exists due to the concept of emergence.

Emergence refers to the behaviors and outcomes that result from how individual system elements compose to form the system as a whole. That is, the behavior and outcomes produced by the system are not those of the individual system elements that comprise the system. Rather, the emergent system behavior and outcomes, or properties, result from the composition of multiple system elements

SECURITY DESIGN ORDER OF PRECEDENCE (SECDOP) [134]

The security design order of precedence (SecDOP) is part of a design approach that uses passive architectural features to provide the structure for the employment of engineered features and devices. SecDOP reflects a design goal to eliminate the design basis for loss potential.

Using a principled and assured engineering approach, the SecDOP eliminates susceptibility, hazard and vulnerability to the extent practicable, thereby eliminating the associated risk. For those cases in which susceptibility, hazard, or vulnerability cannot be eliminated, the SecDOP reduces the loss potential (e.g., occurrence, impact) to the lowest acceptable level within the constraints of cost, schedule and performance. The SecDOP identifies the design options and lists those options in order of decreasing effectiveness, thus enabling a maximized return on investment.

The SecDOP acts as follows:

1. Eliminate the potential for loss through design selection.
2. Reduce the potential for loss through design alteration.
3. Incorporate engineered features or devices to control the potential for loss.
    a. Mandatory security features and devices
    b. Function-specific features and devices
4. Provide visibility and feedback to external entities.
5. Incorporate signage, procedures, training and proper equipment.

[131] *NIST SP 800-160 Vol 1 Rev 1 Appendix D*
[132] *NIST SP 800-160 Vol 1 Rev 1 Appendix D.1*
[133] *NIST SP 800-160 Vol 1 Rev 1 Appendix D.2*
[134] *NIST SP 800-160 Vol 1 Rev 1 Appendix D.3*

Functional Design Considerations[135]

Protection control functions may be characterized and analyzed by using the following designations:

- **Protection Control Decision Functions**. These functions make authorization decisions or take other actions for protection control enforcement functions. For example, a protection control decision function is a function that decides to grant or deny access to a resource based on a request, possibly from a protection control enforcement function.
- **Protection Control Enforcement Functions**. These functions enforce a constraint to ensure that the system or system element exhibits only authorized and intended behaviors or outcomes. For example, a protection control enforcement function enforces a decision to grant or deny access to a resource.
- **Protection Control Infrastructure Functions**. These functions support and help protection control enforcement and control decision functions fulfil their purposes. The functions also provide data or services or perform operations upon which protection control enforcement and decision functions depend. For example, a protection control infrastructure function includes secure storage, secure communication and anomaly detection mechanisms.

Mechanism Design Criteria

To effectively achieve the objectives of trustworthy secure design, mechanisms must satisfy four essential design criteria.

- **Non-Bypassable**. The mechanism must not be circumventable.
- **Evaluatable**. The mechanism must be sufficiently small and simple enough to be assessed to produce adequate confidence in the protection provided, the constraint (or control objective) enforced and the correct implementation of the mechanism. The assessment includes the analysis and testing needed.
- **Always Invoked**. The protection provided by a mechanism or feature that is not always invoked is not continuous and therefore, a loss may occur while the mechanism or feature is suspended or turned off.
- **Tamper Proof**. The mechanism or feature and the data that the mechanism or feature depends on cannot be modified in an unauthorized manner.

Protective Failure

The failure of a security function is of special concern, given the need for security functions to always be invoked and operating correctly. Consequently, failure analyses must be performed during system design to determine the impacts of function failure on the system capabilities, including the protection capability relative to the resulting consequences of such failure and the needed assurance of the protection capability.

Failure analyses consider the assets that may be impacted by security function failure and the associated loss consequences. Failure analyses also consider the function allocation to system elements and the way the system function and element combination interacts with other system function and element combinations, independent of specific events and conditions that might lead to the failure. The outcomes of the security function failure analyses also drive assurance levels and objectives, as well as the fidelity and rigor of architecture, design and implementation methods employed to achieve those objectives.

[135] *NIST SP 800-160 Vol 1 Rev 1 Appendix D.4*

ARCHITECT-LEVEL: SECURE DEVELOPMENT LIFECYCLE (SDL)

There is a multitude of considerations and contributions that build upon traditional System/Software Development Lifecycle (SDLC) processes to produce the behaviors and outcomes that are necessary to achieve trustworthy secure systems.[136]

Per ISO 15288 and NIST SP 800-160 Vol 1 Rev 1, each system life cycle process description has the following sections:

- Life Cycle Purpose: Describes the goals of performing the process [*per ISO 15288*].
- Security Purpose: Establishes what the process achieves from the security standpoint.
- Security Outcomes: Expresses the security-related observable results expected from the successful performance of the process and the data generated by the process.
- Security Activities: Provides a set of cohesive security-related tasks that support achievement of the security outcomes for the process. The tasks are accomplished cooperatively within and across various roles of the organization, inclusive of systems security engineering. While this publication focuses on the scope and responsibility of systems security engineering, it is not the case that all aspects of every task are fulfilled by systems security engineering.

SDL PROCESSES

These thirty (30) NIST SP 800-160 Vol1 Rev 1 Secure Development Lifecycle (SDL) processes are organized into four (4) groups:

Technical Processes

- Business or Mission Analysis (BA) [137]
- Stakeholder Needs and Requirements Definition (SN) [138]
- System Requirements Definition (SR) [139]
- System Architecture Definition (AR) [140]
- Design Definition (DE) [141]
- System Analysis (SA) [142]
- Implementation (IP) [143]
- Integration (IN) [144]
- Verification (VE) [145]
- Transition (TR) [146]
- Validation (VA) [147]
- Operation (OP) [148]
- Maintenance (MA) [149]
- Disposal (DS) [150]

Technical Management Processes

- Project Planning (PL) [151]
- Project Assessment and Control (PA) [152]
- Decision Management (DM) [153]
- Risk Management (RM) [154]

---

[136] *NIST SP 800-160 Vol 1 Rev 1*
[137] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.1*
[138] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.2*
[139] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.3*
[140] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.4*
[141] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.5*
[142] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.6*
[143] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.7*
[144] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.8*
[145] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.9*
[146] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.10*
[147] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.11*
[148] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.12*
[149] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.13*
[150] *NIST SP 800-160 Vol 1 Rev 1 – Appendix H.14*
[151] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.1*
[152] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.2*
[153] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.3*
[154] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.4*

- Configuration Management (CM) [155]
- Information Management (IM) [156]
- Measurement (MS) [157]
- Quality Assurance (QA) [158]

Organizational Project Enabling Processes

- Life Cycle Model Management (LM) [159]
- Infrastructure Management (IF) [160]
- Portfolio Management (PM) [161]
- Human Resource Management (HR) [162]
- Quality Management (QM) [163]
- Knowledge Management (KM) [164]

Agreement Process

- Acquisition (AQ) [165]
- Supply (SP) [166]

Microsoft Operational Security Practices (OSP)

An alternative to ISO 15288's approach to SDL, Microsoft's Operational Security Practices (OSP) incorporates the knowledge gained through capabilities that are unique to Microsoft, including the Microsoft Security Development Lifecycle (SDL), the Microsoft Security Response Center (MSRC) program, a deep awareness of the cybersecurity threat landscape and data from industry standard tools. OSP combines this knowledge with the experience of running millions of servers in data centers globally that deliver massive-scale online services to customers and enterprises.

Microsoft's OSP contains the following eleven (11) practices: [167]

OSP Practice 1: Provide Training

Security is everyone's job. Ensuring everyone understands the attacker's perspective, their goals and the art of the possible will help capture the attention of everyone and raise the collective knowledge bar. Developers, service engineers and product managers must understand security basics and know how to build security into software and services to make products more secure while still addressing business needs and delivering user value.

Effective training will complement and re-enforce security policies, OSP Practices, standards and security requirements and be guided by insights derived through data or newly available technical capabilities.

OSP Practice 2: Use Multi-Factor Authentication

Passwords can be stolen and identities compromised. Requiring a second factor in addition to a password immediately improves security. Further, authenticating the identity of a user or administrator and verifying their authorization to perform an action are foundational controls that other security controls are built upon. It's beneficial to standardize on an approach to both authentication and authorization.

---

[155] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.5*
[156] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.6*
[157] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.7*
[158] *NIST SP 800-160 Vol 1 Rev 1 – Appendix I.8*
[159] *NIST SP 800-160 Vol 1 Rev 1 – Appendix J.1*
[160] *NIST SP 800-160 Vol 1 Rev 1 – Appendix J.2*
[161] *NIST SP 800-160 Vol 1 Rev 1 – Appendix J.3*
[162] *NIST SP 800-160 Vol 1 Rev 1 – Appendix J.4*
[163] *NIST SP 800-160 Vol 1 Rev 1 – Appendix J.5*
[164] *NIST SP 800-160 Vol 1 Rev 1 – Appendix J.6*
[165] *NIST SP 800-160 Vol 1 Rev 1 – Appendix K.1*
[166] *NIST SP 800-160 Vol 1 Rev 1 – Appendix K.2*
[167] *Microsoft OSA - https://www.microsoft.com/en-us/securityengineering/osa/practices*

OSP Practice 3: Enforce Least Privilege

It's important to restrict and minimize the number of people in privileged roles who have access to secured information or resources. This reduces the chance of a malicious user getting that access, or an authorized user inadvertently compromising a sensitive resource. However, users still need to carry out privileged operations on a service and there is a need to understand what those operations are and to separate those roles such that there's no easy opportunity for privilege escalation. The principle of "just enough administration" should be adopted to constrain the elevated privilege only to those functions the administrator requires to complete the task at hand and only on a "just-in-time" (JIT) basis and only for the minimum practical period.

The use of Privileged Access Workstations (PAWs) also helps protect privileged users from internet attacks and threat vectors by providing a dedicated machine for sensitive tacks and separating these sensitive tasks and accounts from the daily use workstations.

OSP Practice 4: Protect Secrets

Encrypt and store application secrets and eliminate the need to include secrets and other sensitive configuration information in code or configuration files of the code. Never store passwords or other sensitive data in source code or configuration files or in plaintext files (documents, spreadsheets) stored in unprotected locations. Production secrets should not be used for development or testing.

OSP Practice 5: Minimize Attack Surface

Minimize the number of features that can be attacked by a malicious party. A defense-in-depth approach should be adopted and the attack surface should be minimized at every level of the stack, including limiting and locking down the network ports available, implementing baseline server role configurations and restricting the applications a server is allowed to run.

OSP Practice 6: Encrypt Data in Transit and at Rest

With the rise of mobile and cloud computing, it's critically important to ensure all data—including security-sensitive information and management and control data—is protected from unintended disclosure or alteration when it's being transmitted or stored. Encryption is typically used to achieve this. In the operational world, only use industry-vetted encryption libraries and only use strong versions of the encryption protocol. Also, be sure you understand the protections an encryption solution provides, especially when encrypting stored data.

OSP Practice 7: Implement Security Monitoring

It is critically important to be able to detect, respond to and recover from attacks. Well-designed application, system and security log files are the fundamental data sources that inform automated security information and event management (SIEM) systems alerting and that support forensic analysis in the event of an incident.

OSP Practice 8: Implement A Security Update Strategy

Attackers often exploit previously discovered vulnerabilities for which updates have been published, before the systems they affect are patched. To help address this, all systems must be continuously monitored and updated with the latest security updates. For operating system and software packages, only use currently supported software versions and ideally the latest versions. In addition, to help detect and prevent malware infections, servers should be required to run anti-malware software which will block and remediate potential infections before they can cause damage.

OSP Practice 9: Protect Against DDOS Attacks

Distributed Denial of Service (DDoS) attacks are some of the largest availability and security concerns facing cloud applications, because any endpoint that's publicly reachable over the internet can be targeted. To address this, at a minimum traffic must be continually monitored and real-time mitigations must be provided for common network-level attacks. However, as DDoS attacks become more sophisticated and targeted, it may also be necessary to provide DDoS mitigations to protocol and application layer attacks.

OSP Practice 10: Validate the Configuration of Web Applications and Sites

Website and application scanning is a critical part of maintaining a highly secure operations environment for online services. Regularly validate that websites and web applications are configured optimally to prevent common web attacks and to use secure versions of transport protocols and have opted into security-relevant options. Scans using authenticated credentials will typically produce more valuable results and any issues found should be remediated immediately.

OSP Practice 11: Perform Penetration Testing

The objective of the penetration test is to uncover potential vulnerabilities resulting from coding errors, system configuration faults, or other operational deployment weaknesses. It is performed by a dedicated “red team” of security experts who simulate real-world attacks at the network, platform and application layers—challenging the ability of cloud services “blue team”, a dedicated team of security responders, to detect, protect against and recover from security breaches. Every Red Team breach is followed by full disclosure between the Red Team and Blue Team to identify gaps, address findings and significantly improve breach response.

GLOSSARY: ACRONYMS & DEFINITIONS

ACRONYMS

Application-Specific Integrated Circuit (ASIC)
Applications, Services, and Processes (ASP)
Body of Knowledge (BoK)
Certificate of Competence (CoC)
Certified SCA Architect (CSCAA)
Certified SCA Practitioner (CSCAP)
Commercial-Off-The-Shelf (COTS)
Common Weakness Enumeration (CWE)
Computerized Numerical Control (CNC)
Confidentiality, Integrity, Availability & Safety (CIAS)
Cyber-Physical Systems (CPS)
Defense Acquisition Regulations System (DFARS)
Developing Security & Privacy by Design (DSPD)
Development & Operations (DevOps)
Digital Signal Processor (DSP)
Discretionary Security Requirements (DSR)
Distributed Denial of Service (DDoS)
Enterprise Information Technology (EIT)
Executive Order (EO)
Federal Acquisition Regulation (FAR)
Field-Programmable Gate Array (FPGA)
Governance, Risk & Compliance (GRC)
Government-Off-The-Shelf (GOTS)
High Value Asset (HVA)
High Value Target (HVT)
Individual Contributors (IC)
Industrial Control Systems (ICS)
Integrated Controls Management (ICM)
International Organization for Standardization (ISO)
Just In Time (JIT)
Microsoft Security Response Center (MSRC)
Minimum Compliance Criteria (MCR)
Minimum Security Requirements (MSR)
Minimum Viable Product (MVP)
National Institute of Standards and Technology (NIST)
Open Web Application Security Project (OWASP)
Operating Systems (OS)
Operational Security Assurance (OSA)
Operational Technology (OT)
Privileged Access Workstations (PAW)
Programmable Logic Controllers (PLCs)
Secure Code Alliance (SCA)
Secure Development Lifecycle (SDL)
Secure Software Development Practices (SSDP)
Security, Development & Operations (SecDevOps)
Software Bill of Materials (SBOM)
Software/System Development Life Cycle (SDLC)
Supervisory Control and Data Acquisition (SCADA)
Supplier's Declaration of Conformity (SDoC)
Trustworthy Secure Design (TSD)

DEFINITIONS

The SCA-BoK recognizes two (2) sources for authoritative reference documents to define common cybersecurity and data protection terms:

- The National Institute of Standards and Technology (NIST) IR 7298, *Glossary of Key Cybersecurity Terms*, is the approved reference document used to define common digital security terms;[168] and
- NIST Glossary.[169]

Security Requirements and Controls

The term control can be applied to a variety of contexts and can serve multiple purposes. When used in the security context, a security control can be a mechanism (e.g., a safeguard or countermeasure) designed to address protection needs that are specified by a set of security requirements.

- Controls are defined as the power to make decisions about how something is managed or how something is done; the ability to direct the actions of someone or something; an action, method or law that limits; or a device or mechanism used to regulate or guide the operation of a machine, apparatus or system.

[168] *NIST IR 7298 - https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.7298r3.pdf*
[169] *NIST Glossary - https://csrc.nist.gov/glossary*

NORMATIVE REFERENCES

- Building Security In Maturity Model (BSIMM)[170]
- Cloud Security Alliance (CSA) – Top Threats to Cloud Computing: Egregious Eleven[171]
- Common Weakness Enumeration (CWE): Software Development[172]
- Integrated Controls Management (ICM)[173]
- Dark Reading – NIST Misses Opportunity With New 'Minimum Standard' for Software Security Testing[174]
- DevOps.com[175]
- Executive Order 14028– Improving the Nation's Cybersecurity [176]
- Executive Order 14028, Improving the Nation's Cybersecurity: NIST's Responsibilities under the Executive Order[177]
- Minimum Viable Secure Product (MVSP)[178]
- Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework (SSDF)[179]
- NIST SP 800-160 Vol. 1 Rev 1 – Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems[180]
- NIST SP 800-160 Vol. 2 Rev 1 – Developing Cyber-Resilient Systems: A Systems Security Engineering Approach[181]
- NIST SP 800-218 – Secure Software Development Framework (SSDF) Version 1:1 Recommendations for Mitigating the Risk if Software Vulnerabilities[182]
- NIST SP 800-218A – Secure Software Development Practices for Generative AI and Dual-Use Foundation Models: An SSDF Community Profile[183]
- Open Web Application Security Project (OWASP) - Security Assurance Maturity Model (SAMM)[184]
- State of Alabama – Information Technology Guideline 661G2-00: Security Engineering Principles[185]
- Secure Controls Framework (SCF)[186]
- Supply Chain Risk Management (SCRM)[187]

[170] https://www.bsimm.com
[171] https://cloudsecurityalliance.org/artifacts/top-threats-to-cloud-computing-egregious-eleven
[172] https://cwe.mitre.org/data/definitions/699.html
[173] https://www.complianceforge.com/reasons-to-buy/integrated-controls-management
[174] https://www.darkreading.com/edge-articles/nist-misses-opportunity-with-new-minimum-standard-for-software-security-testing
[175] https://devops.com/category/blogs/devsecops
[176] https://www.federalregister.gov/documents/2021/05/17/2021-10460/improving-the-nations-cybersecurity
[177] https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity
[178] https://mvsp.dev/mvsp.en/index.html
[179] https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04232020.pdf
[180] https://csrc.nist.gov/pubs/sp/800/161/r1/upd1/final
[181] https://csrc.nist.gov/pubs/sp/800/160/v2/r1/final
[182] https://csrc.nist.gov/publications/detail/sp/800-218/final
[183] https://csrc.nist.gov/pubs/sp/800/218/a/final
[184] https://www.opensamm.org
[185] https://oit.alabama.gov/wp-content/uploads/2017/09/Guideline_661G2_Security_Engineering_Principles.pdf
[186] https://www.securecontrolsframework.com
[187] https://www.metascrm.com