# Set Theory Relationship Mapping (STRM)

**Reference Document :** **Secure Controls Framework (SCF) version 2024.4**
**Focal Document:** **NIST SP 800-218 Secure Software Development Framework (SSDF) Version 1.1**
**Focal Document Source:** **https://csrc.nist.gov/pubs/sp/800/218/final**
**STRM URL:** **https://securecontrolsframework.com/content/strm/scf-strm-nist-800-218.pdf**

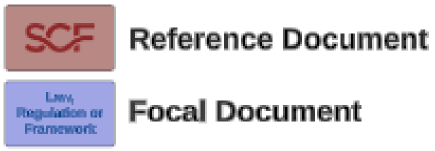
**Set Theory Relationship Mapping (STRM)** is well-suited for mapping between sets of elements that exist in two distinct concepts that are mostly the same as each other (e.g., cybersecurity & data privacy requirements). STRM also allows the strength of the mapping to be captured.

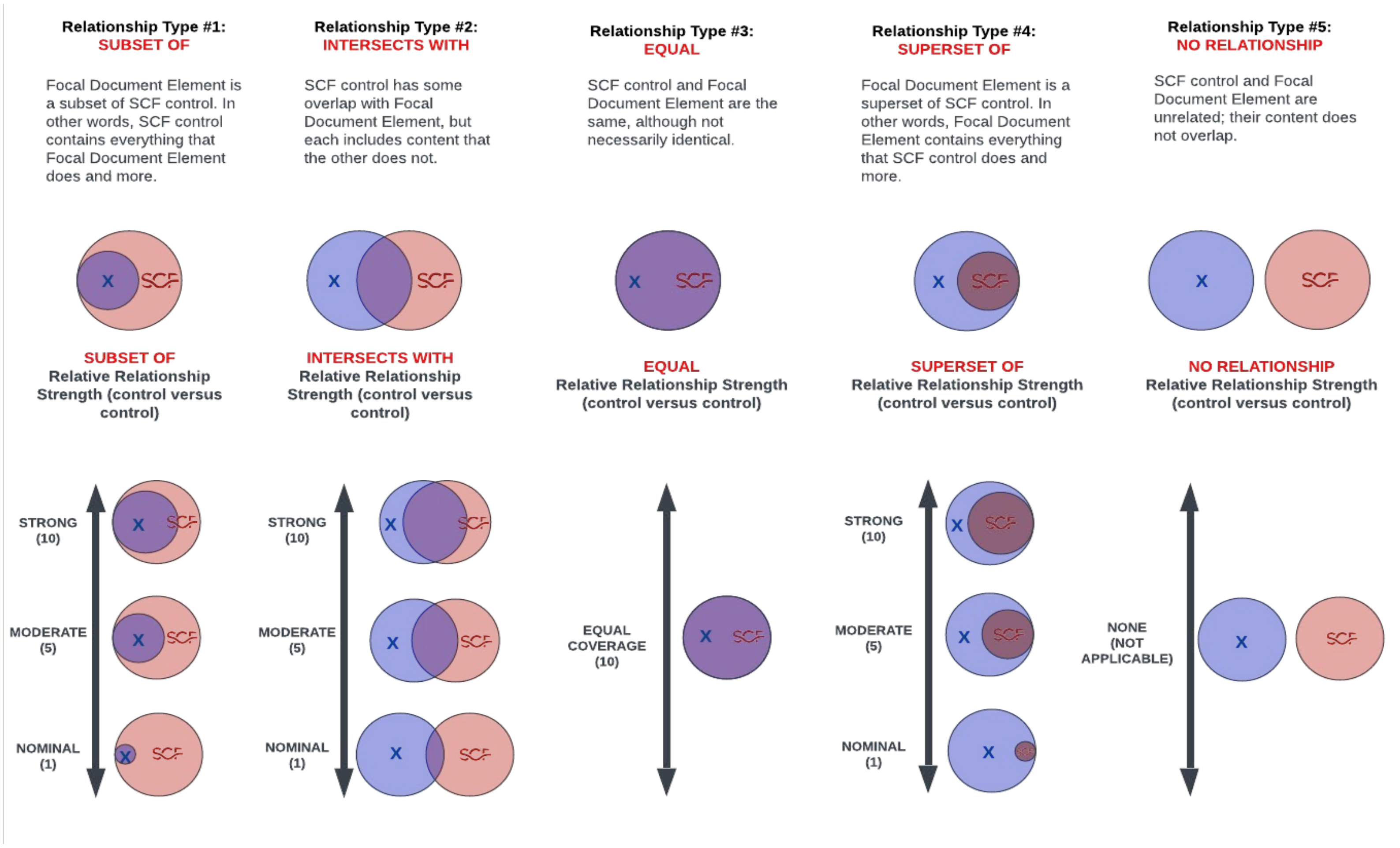
STRM relies on a justification for the relationship claim. There are three (3) options for the rationale, which is a high-level context within which the two concepts are related:

1. **Syntactic**: How similar is the wording that expresses the two concepts? This is a word-for-word analysis of the relationship, not an interpretation of the language.
2. **Semantic**: How similar are the meanings of the two concepts? This involves some interpretation of each concept's language.
3. **Functional**: How similar are the results of executing the two concepts? This involves understanding what will happen if the two concepts are implemented, performed, or otherwise executed.

Based on NIST IR 8477, STRM supports five (5) five relationship types to describe the logical similarity between two distinct concepts:

1. **Subset Of**
2. **Intersects With**
3. **Equal**
4. **Superset Of**
5. **No Relationship**





Focal Document Element is a subset of SCF control. In other words, SCF control contains everything that Focal Document Element does and more.

SCF control has some overlap with Focal Document Element, but each includes content that the other does not.

SCF control and Focal Document Element are the same, although not necessarily identical.

Focal Document Element is a superset of SCF control. In other words, Focal Document Element contains everything that SCF control does and more.

SCF control and Focal Document Element are unrelated; their content does not overlap.

| FDE # | FDE Name | Focal Document Element (FDE) Description | STRM Rationale | STRM Relationship | SCF Control | SCF # | Secure Controls Framework (SCF) Control Description | Strength of Relationship (optional) | Notes (optional) |
|---|---|---|---|---|---|---|---|---|---|
| PO.1 | Define Security Requirements for Software Development | 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). | Functional | Intersects With | Statutory, Regulatory & Contractual Compliance | CPL-01 | Mechanisms exist to facilitate the identification and implementation of relevant statutory, regulatory and contractual controls. | | Example 1: Define policies for securing software development infrastructures and their components, including development endpoints, throughout the SDLC and maintaining that security.<br>Example 2: Define policies for securing software development |
| | | | Functional | Intersects With | Compliance Scope | CPL-01.2 | Mechanisms exist to document and validate the scope of cybersecurity & data privacy controls that are determined to meet statutory, regulatory and/or contractual | | |
| | | | Functional | Intersects With | Data Privacy Requirements for Contractors & Service Providers | PRI-07.1 | Mechanisms exist to include data privacy requirements in contracts and other acquisition-related documents that establish data privacy roles and responsibilities for | | |
| | | | Functional | Intersects With | Cybersecurity & Data Privacy Requirements Definition | PRM-05 | Mechanisms exist to identify critical system components and functions by performing a criticality analysis for critical systems, system components or services at pre-defined | | |
| | | | Functional | Intersects With | Secure Development Life Cycle (SDLC) Management | PRM-07 | Mechanisms exist to ensure changes to systems within the Secure Development Life Cycle (SDLC) are controlled through formal change control procedures. | | |
| | | | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | |
| | | | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| | | | Functional | Subset Of | Technology Development & Acquisition | TDA-01 | Mechanisms exist to facilitate the implementation of tailored development and acquisition strategies, contract tools and procurement methods to meet unique | | |
| | | | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| | | | Functional | Intersects With | Third-Party Contract Requirements | TPM-05 | Mechanisms exist to require contractual requirements for cybersecurity & data privacy requirements with third-parties, reflecting the organization's needs to protect its | | |
| PO.1.1 | N/A | Identify and document all security requirements for the organization's software development infrastructures and processes, and maintain the requirements over time. | Functional | Intersects With | Cybersecurity & Data Privacy Requirements Definition | PRM-05 | Mechanisms exist to identify critical system components and functions by performing a criticality analysis for critical systems, system components or services at pre-defined | | Example 1: Define policies that specify risk-based software architecture and design requirements, such as making code modular to facilitate code reuse and updates; isolating security components from other components during execution; avoiding |
| | | | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | |
| | | | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| PO.1.2 | N/A | Identify and document all security requirements for organization-developed software to meet, and maintain the requirements over time. | Functional | Subset Of | Statutory, Regulatory & Contractual Compliance | CPL-01 | Mechanisms exist to facilitate the identification and implementation of relevant statutory, regulatory and contractual controls. | | Example 1: Define a core set of security requirements for software components, and include it in acquisition documents, software contracts, and other agreements with third parties.<br>Example 2: Define security-related criteria for selecting |
| | | | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | |
| | | | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| PO.2 | Implement Roles and Responsibilities | 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. | Functional | Intersects With | Defined Roles & Responsibilities | HRS-03 | Mechanisms exist to define cybersecurity roles & responsibilities for all personnel. | | |
| | | | Functional | Intersects With | Competency Requirements for Security-Related Positions | HRS-03.2 | Mechanisms exist to ensure that all security-related positions are staffed by qualified individuals who have the necessary skill set. | | |
| PO.2.1 | N/A | 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. | Functional | Subset Of | Human Resources Security Management | HRS-01 | Mechanisms exist to facilitate the implementation of personnel security controls. | | Example 1: Define SDLC-related roles and responsibilities for all members of the software development team.<br>Example 2: Integrate the security roles into the software development team. |
| | | | Functional | Intersects With | Defined Roles & Responsibilities | HRS-03 | Mechanisms exist to define cybersecurity roles & responsibilities for all personnel. | | |
| PO.2.2 | N/A | 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. | Functional | Equal | Role-Based Cybersecurity & Data Privacy Training | SAT-03 | Mechanisms exist to provide role-based cybersecurity & data privacy-related training:<br>(1) Before authorizing access to the system or performing assigned duties; | | Example 1: Document the desired outcomes of training for each role.<br>Example 2: Define the type of training or curriculum required to achieve the desired outcome for each role. |
| | | | Functional | Intersects With | Sensitive Information Storage, Handling & Processing | SAT-03.3 | Mechanisms exist to ensure that every user accessing a system processing, storing or transmitting sensitive information is formally trained in data handling requirements. | | |
| | | | Functional | Intersects With | Privileged Users | SAT-03.5 | Mechanisms exist to provide specific training for privileged users to ensure privileged users understand their unique roles and responsibilities | | |
| | | | Functional | Intersects With | Cyber Threat Environment | SAT-03.6 | Mechanisms exist to provide role-based cybersecurity & data privacy awareness training that is current and relevant to the cyber threats that users might encounter in | | |
| PO.2.3 | N/A | Obtain upper management or authorizing official commitment to secure development, and convey that commitment to all with development-related roles and responsibilities. | Functional | Intersects With | Assigned Cybersecurity & Data Protection Responsibilities | GOV-04 | Mechanisms exist to assign one or more qualified individuals with the mission and resources to centrally-manage, coordinate, develop, implement and maintain an | | Example 1: Appoint a single leader or leadership team to be responsible for the entire secure software development process, including being accountable for releasing software to production and delegating responsibilities as appropriate. |
| | | | Functional | Intersects With | Stakeholder Accountability Structure | GOV-04.1 | Mechanisms exist to enforce an accountability structure so that appropriate teams and individuals are empowered, responsible and trained for mapping, | | |
| PO.3 | Implement Supporting Toolchains | 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. | Functional | Subset Of | Technology Development & Acquisition | TDA-01 | Mechanisms exist to facilitate the implementation of tailored development and acquisition strategies, contract tools and procurement methods to meet unique | | |
| | | | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | |
| | | | Functional | Equal | Supporting Toolchain | TDA-06.4 | Automated mechanisms exist to improve the accuracy, consistency and comprehensiveness of secure practices throughout the asset's lifecycle. | | |

| FDE # | FDE Name | Focal Document Element (FDE) Description | STRM Rationale | STRM Relationship | SCF Control | SCF # | Secure Controls Framework (SCF) Control Description | Strength of Relationship (optional) | Notes (optional) |
|---|---|---|---|---|---|---|---|---|---|
| PO.3.1 | N/A | 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. | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | Example 1: Define categories of toolchains, and specify the mandatory tools or tool types to be used for each category.<br>Example 2: Identify security tools to integrate into the developer toolchain. |
| | | | Functional | Intersects With | Supporting Toolchain | TDA-06.4 | Automated mechanisms exist to improve the accuracy, consistency and comprehensiveness of secure practices throughout the asset's lifecycle. | | |
| PO.3.2 | N/A | Follow recommended security practices to deploy, operate, and maintain tools and toolchains. | Functional | Intersects With | Technology Development & Acquisition | TDA-01 | Mechanisms exist to facilitate the implementation of tailored development and acquisition strategies, contract tools and procurement methods to meet unique | | Example 1: Evaluate, select, and acquire tools, and assess the security of each tool.<br>Example 2: Integrate tools with other tools and existing software development processes and workflows. |
| | | | Functional | Intersects With | Standardized Operating Procedures (SOP) | OPS-01.1 | Mechanisms exist to identify and document Standardized Operating Procedures (SOP), or similar documentation, to enable the proper execution of day-to-day / assigned tasks. | | |
| | | | Functional | Intersects With | Service Delivery (Business Process Support) | OPS-03 | Mechanisms exist to define supporting business processes and implement appropriate governance and service management to ensure appropriate planning, | | |
| | | | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | |
| | | | Functional | Intersects With | Supporting Toolchain | TDA-06.4 | Automated mechanisms exist to improve the accuracy, consistency and comprehensiveness of secure practices throughout the asset's lifecycle. | | |
| PO.3.3 | N/A | Configure tools to generate artifacts of their support of secure software development practices as defined by the organization. | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | Example 1: Use existing tooling (e.g., workflow tracking, issue tracking, value stream mapping) to create an audit trail of the secure development-related actions that are performed for continuous improvement purposes. |
| | | | Functional | Intersects With | Identification & Justification of Ports, Protocols & Services | TDA-02.5 | Mechanisms exist to require process owners to identify, document and justify the business need for the ports, protocols and other services necessary to operate their | 3 | |
| | | | Functional | Intersects With | Documentation Requirements | TDA-04 | Mechanisms exist to obtain, protect and distribute administrator documentation for systems that describe:<br>(1) Secure configuration, installation and | | |
| | | | Functional | Intersects With | Functional Properties | TDA-04.1 | Mechanisms exist to require software developers to provide information describing the functional properties of the security controls to be utilized within systems, system | 3 | |
| PO.4 | Define and Use Criteria for Software Security Checks | 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. | Functional | Intersects With | Software Design Review | TDA-06.5 | Mechanisms exist to have an independent review of the software design to confirm that all cybersecurity & data privacy requirements are met and that any identified risks are | 5 | |
| | | | Functional | Intersects With | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | 5 | |
| | | | Functional | Intersects With | Static Code Analysis | TDA-09.2 | Mechanisms exist to require the developers of systems, system components or services to employ static code analysis tools to identify and remediate common flaws and | 3 | |
| | | | Functional | Intersects With | Dynamic Code Analysis | TDA-09.3 | Mechanisms exist to require the developers of systems, system components or services to employ dynamic code analysis tools to identify and remediate common flaws and | 3 | |
| PO.4.1 | N/A | Define criteria for software security checks and track throughout the SDLC. | Functional | Intersects With | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | Example 1: Ensure that the criteria adequately indicate how effectively security risk is being managed.<br>Example 2: Define key performance indicators (KPIs), key risk indicators (KRIs), vulnerability severity scores, and other |
| PO.4.2 | N/A | Implement processes, mechanisms, etc. to gather and safeguard the necessary information in support of the criteria. | Functional | Intersects With | Standardized Operating Procedures (SOP) | OPS-01.1 | Mechanisms exist to identify and document Standardized Operating Procedures (SOP), or similar documentation, to enable the proper execution of day-to-day / assigned tasks. | | Example 1: Use the toolchain to automatically gather information that informs security decision-making.<br>Example 2: Deploy additional tools if needed to support the generation and collection of information supporting the criteria. |
| | | | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| | | | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | |
| PO.5 | Implement and Maintain Secure Environments for Software Development | 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. | Functional | Intersects With | Development & Test Environment Configurations | CFG-02.4 | Mechanisms exist to manage baseline configurations for development and test environments separately from operational baseline configurations to minimize the risk | | |
| | | | Functional | Subset Of | Secure Development Environments | TDA-07 | Mechanisms exist to maintain a segmented development network to ensure a secure development environment. | | |
| | | | Functional | Intersects With | Separation of Development, Testing and Operational Environments | TDA-08 | Mechanisms exist to manage separate development, testing and operational environments to reduce the risks of unauthorized access or changes to the | | |
| | | | Functional | Intersects With | Secure Migration Practices | TDA-08.1 | Mechanisms exist to ensure secure migration practices purge systems, applications and services of test/development/staging data and accounts before it is migrated into a | | |
| PO.5.1 | N/A | Separate and protect each environment involved in software development. | Functional | Subset Of | Secure Development Environments | TDA-07 | Mechanisms exist to maintain a segmented development network to ensure a secure development environment. | | Example 1: Use multi-factor, risk-based authentication and conditional access for each environment.<br>Example 2: Use network segmentation and access controls to separate the environments from each other and from |
| | | | Functional | Intersects With | Separation of Development, Testing and Operational Environments | TDA-08 | Mechanisms exist to manage separate development, testing and operational environments to reduce the risks of unauthorized access or changes to the | | |
| PO.5.2 | N/A | Secure and harden development endpoints (i.e., endpoints for software designers, developers, testers, builders, etc.) to perform development-related tasks using a risk-based approach. | Functional | Subset Of | System Hardening Through Baseline Configurations | CFG-02 | Mechanisms exist to develop, document and maintain secure baseline configurations for technology platforms that are consistent with industry-accepted system hardening | | Example 1: Configure each development endpoint based on approved hardening guides, checklists, etc.; for example, enable FIPS-compliant encryption of all sensitive data at rest and in transit. |
| | | | Functional | Intersects With | Development & Test Environment Configurations | CFG-02.4 | Mechanisms exist to manage baseline configurations for development and test environments separately from operational baseline configurations to minimize the risk | | |
| | | | Functional | Intersects With | Configure Systems, Components or Services for High-Risk Areas | CFG-02.5 | Mechanisms exist to configure systems utilized in high-risk areas with more restrictive baseline configurations. | | |
| PS.1 | Protect All Forms of Code from Unauthorized Access and Tampering | 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. | Functional | Intersects With | Access Restriction For Change | CHG-04 | Mechanisms exist to enforce configuration restrictions in an effort to restrict the ability of users to conduct unauthorized changes. | | |
| | | | Functional | Intersects With | Library Privileges | CHG-04.5 | Mechanisms exist to restrict software library privileges to those individuals with a pertinent business need for access. | | |

| FDE # | FDE Name | Focal Document Element (FDE) Description | STRM Rationale | STRM Relationship | SCF Control | SCF # | Secure Controls Framework (SCF) Control Description | Strength of Relationship (optional) | Notes (optional) |
|---|---|---|---|---|---|---|---|---|---|
| PS.1.1 | N/A | 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. | Functional | Equal | Access to Program Source Code | TDA-20 | Mechanisms exist to limit privileges to change software resident within software libraries. | | |
| PS.2 | Provide a Mechanism for Verifying Software Release Integrity | Help software acquirers ensure that the software they acquire is legitimate and has not been tampered with. | Functional | Equal | Software Release Integrity Verification | TDA-20.1 | Mechanisms exist to publish integrity verification information for software releases. | | |
| PS.2.1 | N/A | Make software integrity verification information available to software acquirers. | Functional | Equal | Software Release Integrity Verification | TDA-20.1 | Mechanisms exist to publish integrity verification information for software releases. | | Example 1: Post cryptographic hashes for release files on a well-secured website.<br>Example 2: Use an established certificate authority for code signing so that consumers' operating systems or other tools and |
| PS.3 | Archive and Protect Each Software Release | Preserve software releases in order to help identify, analyze, and eliminate vulnerabilities discovered in the software after release. | Functional | Equal | Archiving Software Releases | TDA-20.2 | Mechanisms exist to archive software releases and all of their components (e.g., code, package files, third-party libraries, documentation) to maintain integrity | | |
| PS.3.1 | N/A | Securely archive the necessary files and supporting data (e.g., integrity verification information, provenance data) to be retained for each software release. | Functional | Equal | Archiving Software Releases | TDA-20.2 | Mechanisms exist to archive software releases and all of their components (e.g., code, package files, third-party libraries, documentation) to maintain integrity | | Example 1: Store the release files, associated images, etc. in repositories following the organization's established policy. Allow read-only access to them by necessary personnel and no access by anyone else. |
| | | | Functional | Intersects With | Software Escrow | TDA-20.3 | Mechanisms exist to escrow source code and supporting documentation to ensure software availability in the event the software provider goes out of business or is unable to | | |
| PS.3.2 | N/A | Collect, safeguard, maintain, and share provenance data for all components of each software release (e.g., in a software bill of materials [SBOM]). | Functional | Intersects With | Documentation Requirements | TDA-04 | Mechanisms exist to obtain, protect and distribute administrator documentation for systems that describe:<br>(1) Secure configuration, installation and | | |
| | | | Functional | Intersects With | Software Bill of Materials (SBOM) | TDA-04.2 | Mechanisms exist to generate, or obtain, a Software Bill of Materials (SBOM) for systems, applications and services that lists software packages in use, including versions and | | |
| PW.1 | Design Software to Meet Security Requirements and Mitigate Security Risks | 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 risks during software design (secure by design) is key for improving software security and also helps improve development efficiency. | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| | | | Functional | Intersects With | Criticality Analysis | TDA-06.1 | Mechanisms exist to require the developer of the system, system component or service to perform a criticality analysis at organization-defined decision points in the Secure | | |
| | | | Functional | Intersects With | Threat Modeling | TDA-06.2 | Mechanisms exist to perform threat modelling and other secure design techniques, to ensure that threats to software and solutions are identified and | | |
| | | | Functional | Intersects With | Software Assurance Maturity Model (SAMM) | TDA-06.3 | Mechanisms exist to utilize a Software Assurance Maturity Model (SAMM) to govern a secure development lifecycle for the development of systems, applications and | | |
| PW.1.1 | N/A | Use forms of risk modeling – such as threat modeling, attack modeling, or attack surface mapping – to help assess the security risk for the software. | Functional | Intersects With | Threat Modeling | TDA-06.2 | Mechanisms exist to perform threat modelling and other secure design techniques, to ensure that threats to software and solutions are identified and | | Example 1: Train the development team (security champions, in particular) or collaborate with a risk modeling expert to create models and analyze how to use a risk-based approach to communicate the risks and determine how to address them, |
| PW.1.2 | N/A | Track and maintain the software's security requirements, risks, and design decisions. | Functional | Subset Of | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | Example 1: Record the response to each risk, including how mitigations are to be achieved and what the rationales are for any approved exceptions to the security requirements. Add any mitigations to the software's security requirements. |
| | | | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | |
| PW.1.3 | N/A | 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. [Formerly PW.4.3] | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | Example 1: Maintain one or more software repositories of modules for supporting standardized security features and services.<br>Example 2: Determine secure configurations for modules for |
| | | | Functional | Intersects With | Secure Settings By Default | TDA-09.6 | Mechanisms exist to implement secure configuration settings by default to reduce the likelihood of software being deployed with weak security settings that would put | | |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| PW.2 | Review the Software Design to Verify Compliance with Security Requirements and Risk Information | Help ensure that the software will meet the security requirements and satisfactorily address the identified risk information. | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | |
| | | | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | |
| | | | Functional | Intersects With | Insecure Ports, Protocols & Services | TDA-02.6 | Mechanisms exist to mitigate the risk associated with the use of insecure ports, protocols and services necessary to operate technology solutions. | | |
| | | | Functional | Intersects With | Cybersecurity & Data Privacy Representatives For Product Changes | TDA-02.7 | Mechanisms exist to include appropriate cybersecurity & data privacy representatives in the product feature and/or functionality change control review process. | | |
| | | | Functional | Intersects With | Software Assurance Maturity Model (SAMM) | TDA-06.3 | Mechanisms exist to utilize a Software Assurance Maturity Model (SAMM) to govern a secure development lifecycle for the development of systems, applications and | | |
| | | | Functional | Intersects With | Software Design Review | TDA-06.5 | Mechanisms exist to have an independent review of the software design to confirm that all cybersecurity & data privacy requirements are met and that any identified risks are | | |
| PW.2.1 | N/A | 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. | Functional | Equal | Software Design Review | TDA-06.5 | Mechanisms exist to have an independent review of the software design to confirm that all cybersecurity & data privacy requirements are met and that any identified risks are | | Example 1: Review the software design to confirm that it addresses applicable security requirements.<br>Example 2: Review the risk models created during software design to determine if they appear to adequately identify the |
| PW.4 | Reuse Existing, Well-Secured Software When Feasible Instead of Duplicating Functionality | 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. | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| | | | Functional | Intersects With | Pre-Established Secure Configurations | TDA-02.4 | Mechanisms exist to ensure vendors / manufacturers:<br>(1) Deliver the system, component, or service with a pre-established, secure | | |
| | | | Functional | Intersects With | Commercial Off-The-Shelf (COTS) Security Solutions | TDA-03 | Mechanisms exist to utilize only Commercial Off-the-Shelf (COTS) security products. | | |
| PW.4.1 | N/A | 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. | Functional | Intersects With | Commercial Off-The-Shelf (COTS) Security Solutions | TDA-03 | Mechanisms exist to utilize only Commercial Off-the-Shelf (COTS) security products. | | Example 1: Review and evaluate third-party software components in the context of their expected use. If a component is to be used in a substantially different way in the future, perform the review and evaluation again with that new |
| | | | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | Example 1: Follow organization-established security practices for secure software development when creating and maintaining the components.<br>Example 2: Determine secure configurations for software |

| FDE # | FDE Name | Focal Document Element (FDE) Description | STRM Rationale | STRM Relationship | SCF Control | SCF # | Secure Controls Framework (SCF) Control Description | Strength of Relationship (optional) | Notes (optional) |
|---|---|---|---|---|---|---|---|---|---|
| PW.4.2 | N/A | 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. | Functional | Intersects With | Developer Architecture & Design | TDA-05 | Mechanisms exist to require the developers of systems, system components or services to produce a design specification and security architecture that: | | |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| | | | Functional | Intersects With | Software Assurance Maturity Model (SAMM) | TDA-06.3 | Mechanisms exist to utilize a Software Assurance Maturity Model (SAMM) to govern a secure development lifecycle for the development of systems, applications and | | |
| PW.4.4 | N/A | 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. | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | Example 1: Regularly check whether there are publicly known vulnerabilities in the software modules and services that vendors have not yet fixed.<br>Example 2: Build into the toolchain automatic detection of |
| | | | Functional | Intersects With | Ports, Protocols & Services In Use | TDA-02.1 | Mechanisms exist to require the developers of systems, system components or services to identify early in the Secure Development Life Cycle (SDLC), the functions, ports, | | |
| | | | Functional | Intersects With | Identification & Justification of Ports, Protocols & Services | TDA-02.5 | Mechanisms exist to require process owners to identify, document and justify the business need for the ports, protocols and other services necessary to operate their | | |
| | | | Functional | Intersects With | Insecure Ports, Protocols & Services | TDA-02.6 | Mechanisms exist to mitigate the risk associated with the use of insecure ports, protocols and services necessary to operate technology solutions. | | |
| | | | Functional | Intersects With | Software Bill of Materials (SBOM) | TDA-04.2 | Mechanisms exist to generate a Software Bill of Materials (SBOM) for systems, applications and services that lists software packages in use, including versions and applicable | | |
| PW.5 | Create Source Code by Adhering to Secure Coding Practices | 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. | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| | | | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| PW.5.1 | N/A | Follow all secure coding practices that are appropriate to the development languages and environment to meet the organization's requirements. | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | Example 1: Validate all inputs, and validate and properly encode all outputs.<br>Example 2: Avoid using unsafe functions and calls.<br>Example 3: Detect errors, and handle them gracefully. |
| | | | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). | | |
| | | | Functional | Intersects With | Pre-Established Secure Configurations | TDA-02.4 | Mechanisms exist to ensure vendors / manufacturers:<br>(1) Deliver the system, component, or service with a pre-established, secure | | |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| | | | Functional | Intersects With | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | |
| | | | Functional | Intersects With | Secure Settings By Default | TDA-09.6 | Mechanisms exist to implement secure configuration settings by default to reduce the likelihood of software being deployed with weak security settings that would put | | |
| PW.6 | Configure the Compilation, Interpreter, and Build Processes to Improve Executable Security | Decrease the number of security vulnerabilities in the software and reduce costs by eliminating vulnerabilities before testing occurs. | Functional | Intersects With | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | |
| | | | Functional | Intersects With | Secure Settings By Default | TDA-09.6 | Mechanisms exist to implement secure configuration settings by default to reduce the likelihood of software being deployed with weak security settings that would put | | |
| PW.6.1 | N/A | Use compiler, interpreter, and build tools that offer features to improve executable security. | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure | | Example 1: Use up-to-date versions of compiler, interpreter, and build tools.<br>Example 2: Follow change management processes when deploying or updating compiler, interpreter, and build tools, and |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | Example 1: Use up-to-date versions of compiler, interpreter, and build tools.<br>Example 2: Follow change management processes when deploying or updating compiler, interpreter, and build tools, and |
| | | | Functional | Intersects With | Supporting Toolchain | TDA-06.4 | Automated mechanisms exist to improve the accuracy, consistency and comprehensiveness of secure practices throughout the asset's lifecycle. | | |
| PW.6.2 | N/A | Determine which compiler, interpreter, and build tool features should be used and how each should be configured, then implement and use the approved configurations. | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | Example 1: Enable compiler features that produce warnings for poorly secured code during the compilation process.<br>Example 2: Implement the "clean build" concept, where all compiler warnings are treated as errors and eliminated except |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |
| | | | Functional | Intersects With | Supporting Toolchain | TDA-06.4 | Automated mechanisms exist to improve the accuracy, consistency and comprehensiveness of secure practices throughout the asset's lifecycle. | | |
| PW.7 | Review and/or Analyze Human-Readable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements | 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. | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | |
| | | | Functional | Intersects With | Static Code Analysis | TDA-09.2 | Mechanisms exist to require the developers of systems, system components or services to employ static code analysis tools to identify and remediate common flaws and | | |
| | | | Functional | Intersects With | Dynamic Code Analysis | TDA-09.3 | Mechanisms exist to require the developers of systems, system components or services to employ dynamic code analysis tools to identify and remediate common flaws and | | |
| | | | Functional | Intersects With | Malformed Input Testing | TDA-09.4 | Mechanisms exist to utilize testing methods to ensure systems, services and products continue to operate as intended when subject to invalid or unexpected inputs on its | | |
| | | | Functional | Intersects With | Application Penetration Testing | TDA-09.5 | Mechanisms exist to perform application-level penetration testing of custom-made applications and services. | | |

| FDE # | FDE Name | Focal Document Element (FDE) Description | STRM Rationale | STRM Relationship | SCF Control | SCF # | Secure Controls Framework (SCF) Control Description | Strength of Relationship (optional) | Notes (optional) |
|---|---|---|---|---|---|---|---|---|---|
| PW.7.1 | N/A | 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. | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing |  | Example 1: Follow the organization's policies or guidelines for when code review should be performed and how it should be conducted. This may include third-party code and reusable code modules written in-house. |
| PW.7.2 | N/A | 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. | Functional | Intersects With | Static Code Analysis | TDA-09.2 | Mechanisms exist to require the developers of systems, system components or services to employ static code analysis tools to identify and remediate common flaws and |  | Example 1: Perform peer review of code, and review any existing code review, analysis, or testing results as part of the peer review.<br>Example 2: Use expert reviewers to check code for backdoors |
|  |  |  | Functional | Intersects With | Dynamic Code Analysis | TDA-09.3 | Mechanisms exist to require the developers of systems, system components or services to employ dynamic code analysis tools to identify and remediate common flaws and |  |  |
| PW.8 | Test Executable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements | 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. | Functional | Intersects With | Malformed Input Testing | TDA-09.4 | Mechanisms exist to utilize testing methods to ensure systems, services and products continue to operate as intended when subject to invalid or unexpected inputs on its |  |  |
|  |  |  | Functional | Intersects With | Application Penetration Testing | TDA-09.5 | Mechanisms exist to perform application-level penetration testing of custom-made applications and services. |  |  |
| PW.8.1 | N/A | 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. | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct |  | Example 1: Follow the organization's policies or guidelines for when code testing should be performed and how it should be conducted (e.g., within a sandboxed environment). This may include third-party executable code and reusable executable |
|  |  |  | Functional | Intersects With | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing |  |  |
| PW.8.2 | N/A | 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. | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing |  | Example 1: Perform robust functional testing of security features.<br>Example 2: Integrate dynamic vulnerability testing into the project's automated test suite. |
| PW.9 | Configure Software to Have Secure Settings by Default | 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. | Functional | Equal | Secure Settings By Default | TDA-09.6 | Mechanisms exist to implement secure configuration settings by default to reduce the likelihood of software being deployed with weak security settings that would put |  |  |
| PW.9.1 | N/A | 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. | Functional | Equal | System Hardening Through Baseline Configurations | CFG-02 | Mechanisms exist to develop, document and maintain secure baseline configurations for technology platforms that are consistent with industry-accepted system hardening |  | Example 1: Conduct testing to ensure that the settings, including the default settings, are working as expected and are not inadvertently causing any security weaknesses, operational issues, or other problems. |
|  |  |  | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). |  |  |
|  |  |  | Functional | Intersects With | Pre-Established Secure Configurations | TDA-02.4 | Mechanisms exist to ensure vendors / manufacturers:<br>(1) Deliver the system, component, or service with a pre-established, secure |  |  |
|  |  |  | Functional | Intersects With | Secure Settings By Default | TDA-09.6 | Mechanisms exist to implement secure configuration settings by default to reduce the likelihood of software being deployed with weak security settings that would put |  |  |
| PW.9.2 | N/A | Implement the default settings (or groups of default settings, if applicable), and document each setting for software administrators. | Functional | Intersects With | Minimum Viable Product (MVP) Security Requirements | TDA-02 | Mechanisms exist to ensure risk-based technical and functional specifications are established to define a Minimum Viable Product (MVP). |  | Example 1: Verify that the approved configuration is in place for the software.<br>Example 2: Document each setting's purpose, options, default value, security relevance, potential operational impact, and |
|  |  |  | Functional | Intersects With | Pre-Established Secure Configurations | TDA-02.4 | Mechanisms exist to ensure vendors / manufacturers:<br>(1) Deliver the system, component, or service with a pre-established, secure |  |  |
|  |  |  | Functional | Intersects With | Secure Settings By Default | TDA-09.6 | Mechanisms exist to implement secure configuration settings by default to reduce the likelihood of software being deployed with weak security settings that would put |  |  |
| RV.1 | Identify and Confirm Vulnerabilities on an Ongoing Basis | 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. | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure |  |  |
|  |  |  | Functional | Intersects With | Cybersecurity & Data Privacy Representatives For Product Changes | TDA-02.7 | Mechanisms exist to include appropriate cybersecurity & data privacy representatives in the product feature and/or functionality change control review process. |  |  |
|  |  |  | Functional | Intersects With | Software Design Review | TDA-06.5 | Mechanisms exist to have an independent review of the software design to confirm that all cybersecurity & data privacy requirements are met and that any identified risks are |  |  |
|  |  |  | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing |  |  |
|  |  |  | Functional | Intersects With | Continuous Monitoring Plan | TDA-09.1 | Mechanisms exist to require the developers of systems, system components or services to produce a plan for the continuous monitoring of cybersecurity & data privacy |  |  |
| RV.1.1 | N/A | 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. | Functional | Intersects With | Documentation Requirements | TDA-04 | Mechanisms exist to obtain, protect and distribute administrator documentation for systems that describe:<br>(1) Secure configuration, installation and |  | Example 1: Monitor vulnerability databases , security mailing lists, and other sources of vulnerability reports through manual or automated means.<br>Example 2: Use threat intelligence sources to better understand |
|  |  |  | Functional | Intersects With | Functional Properties | TDA-04.1 | Mechanisms exist to require software developers to provide information describing the functional properties of the security controls to be utilized within systems, system |  |  |
|  |  |  | Functional | Intersects With | Software Bill of Materials (SBOM) | TDA-04.2 | Mechanisms exist to generate a Software Bill of Materials (SBOM) for systems, applications and services that lists software packages in use, including versions and applicable |  |  |
|  |  |  | Functional | Intersects With | Developer Architecture & Design | TDA-05 | Mechanisms exist to require the developers of systems, system components or services to produce a design specification and security architecture that: |  |  |
| RV.1.2 | N/A | Review, analyze, and/or test the software's code to identify or confirm the presence of previously undetected vulnerabilities. | Functional | Subset Of | Software Design Review | TDA-06.5 | Mechanisms exist to have an independent review of the software design to confirm that all cybersecurity & data privacy requirements are met and that any identified risks are |  | Example 1: Configure the toolchain to perform automated code analysis and testing on a regular or continuous basis for all supported releases.<br>Example 2: See PW.7 and PW.8. |
| RV.1.3 | N/A | Have a policy that addresses vulnerability disclosure and remediation, and implement the roles, responsibilities, and processes needed to support that policy. | Functional | Equal | Vulnerability Disclosure Program (VDP) | THR-06 | Mechanisms exist to establish a Vulnerability Disclosure Program (VDP) to assist with the secure development and maintenance of products and services that receives |  | Example 1: Establish a vulnerability disclosure program, and make it easy for security researchers to learn about your program and report possible vulnerabilities.<br>Example 2: Have a Product Security Incident Response Team |
| RV.2 | Assess, Prioritize, and Remediate Vulnerabilities | Help ensure that vulnerabilities are remediated in accordance with risk to reduce the window of opportunity for attackers. | Functional | Intersects With | Development Methods, Techniques & Processes | TDA-02.3 | Mechanisms exist to require software developers to ensure that their software development processes employ industry-recognized secure practices for secure |  |  |
|  |  |  | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing |  |  |
| RV.2.1 | N/A | Analyze each vulnerability to gather sufficient information about risk to plan its remediation or other risk response. | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing |  | Example 1: Use existing issue tracking software to record each vulnerability.<br>Example 2: Perform risk calculations for each vulnerability based on estimates of its exploitability, the potential impact if |

| FDE # | FDE Name | Focal Document Element (FDE) Description | STRM Rationale | STRM Relationship | SCF Control | SCF # | Secure Controls Framework (SCF) Control Description | Strength of Relationship (optional) | Notes (optional) |
|---|---|---|---|---|---|---|---|---|---|
| RV.2.2 | N/A | Plan and implement risk responses for vulnerabilities. | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | Example 1: Make a risk-based decision as to whether each vulnerability will be remediated or if the risk will be addressed through other means (e.g., risk acceptance, risk transference), and prioritize any actions to be taken. |
| | | | Functional | Intersects With | Threat Modeling | TDA-06.2 | Mechanisms exist to perform threat modelling and other secure design techniques, to ensure that threats to software and solutions are identified and | | |
| | | | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | |
| | | | Functional | Intersects With | Vulnerability Remediation Process | VPM-02 | Mechanisms exist to ensure that vulnerabilities are properly identified, tracked and remediated. | | |
| RV.3 | Analyze Vulnerabilities to Identify Their Root Cause | Help reduce the frequency of vulnerabilities in the future. | Functional | Subset Of | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| | | | Functional | Intersects With | Root Cause Analysis (RCA) & Lessons Learned | IRO-13 | Mechanisms exist to incorporate lessons learned from analyzing and resolving cybersecurity & data privacy incidents to reduce the likelihood or impact of future | | |
| RV.3.1 | N/A | Analyze identified vulnerabilities to determine their root causes. | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | Example 1: Record the root cause of discovered issues.<br>Example 2: Record lessons learned through root cause analysis in a wiki that developers can access and search. |
| RV.3.2 | N/A | Analyze the root causes over time to identify patterns, such as a particular secure coding practice not being followed consistently. | Functional | Subset Of | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | Example 1: Record lessons learned through root cause analysis in a wiki that developers can access and search.<br>Example 2: Add mechanisms to the toolchain to automatically detect future instances of the root cause. |
| RV.3.3 | N/A | Review the software for similar vulnerabilities to eradicate a class of vulnerabilities, and proactively fix them rather than waiting for external reports. | Functional | Subset Of | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | Example 1: See PW.7 and PW.8. |
| | | | Functional | Intersects With | Cybersecurity & Data Privacy Testing Throughout Development | TDA-09 | Mechanisms exist to require system developers/integrators consult with cybersecurity & data privacy personnel to:<br>(1) Create and implement a Security Testing | | |
| RV.3.4 | N/A | 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. | Functional | Subset Of | Technology Development & Acquisition | TDA-01 | Mechanisms exist to facilitate the implementation of tailored development and acquisition strategies, contract tools and procurement methods to meet unique | | Example 1: Record lessons learned through root cause analysis in a wiki that developers can access and search.<br>Example 2: Plan and implement changes to the appropriate SDLC practices. |
| | | | Functional | Intersects With | Product Management | TDA-01.1 | Mechanisms exist to design and implement product management processes to update products, including systems, software and services, to improve functionality and correct | | |
| | | | Functional | Intersects With | Cybersecurity & Data Privacy Representatives For Product Changes | TDA-02.7 | Mechanisms exist to include appropriate cybersecurity & data privacy representatives in the product feature and/or functionality change control review process. | | |
| | | | Functional | Intersects With | Secure Coding | TDA-06 | Mechanisms exist to develop applications based on secure coding principles. | | |