Advancing the Art of Cybersecurity Cost Estimating

New Considerations for Structure and Assumptions

Austin MacDougall amacdougall@technomics.net

William Gellatly wgellatly@technomics.net

Jessica Kleinman jkleinman@technomics.net

International Cost Estimating and Analysis Association (ICEAA) Professional Development & Training Workshop

5/1/2024

Abstract

The growth in quantity and intensity of cybersecurity threats has led to new cyber best practices, such as Zero Trust and Secure by Design. These practices present challenges when developing cost estimates for the development and maintenance of information systems. This paper examines how these topics and other new cyber trends influence costs. It evaluates the cost implications in both the design (incorporating cyber requirements into new system development) and sustainment (cyber support for existing systems) phases. For each phase, this research also examines cyber frameworks used throughout industry and relates them into a cost element structure that can help estimators collect data and generate defensible estimating methodologies. Finally, this paper translates cyber cost estimating lessons learned into recommended content improvements to the technical baseline documentation upon which cost estimators rely. Standard treatment of cyber in technical baselines should facilitate much needed consistency in the composition of cyber cost estimates.

Keywords: *Cybersecurity, Zero-Trust Architecture, Secure by Design, Work Breakdown Structure, IT Cost Estimating*

Table of Contents

1. Introduction

1.1. Purpose

In recent years, cybersecurity has become a topic of increasing attention for Information Technology (IT) systems. Recent advancements in cybersecurity strategies include new approaches to core cyber functions such as authentication and access management. Additionally, federal and private-sector systems have begun to incorporate security enhancements earlier in the Software Development Life Cycle (SDLC). Cyber practitioners emphasize building cyber requirements into a system at development, rather than relegating cybersecurity to a patch and maintenance task.

These advances pose a challenge to developing credible cost estimates. Programs eagerly seek to understand the cost to modernize their cybersecurity posture but lack relevant data to estimate cost properly. Similarly, cost estimating practitioners themselves lack the 'right' historical data. This is due in part to the fact that general cost benchmarks and standard Work Breakdown Structures (WBS) provide little visibility into cybersecurity to facilitate understanding of cost. In the absence of useful guidance and standards, organizations left to their own devices have unwittingly compounded the problem by routinely accounting for cybersecurity-related costs in a variety of life cycle cost elements, including but not limited to hosting, software license procurement, and help desk support. The obvious outcome of this situation is lack of understanding of cybersecurity costs to facilitate informed decisions. As federal agencies and private-sector organizations target mandate-driven cybersecurity developments, improving the cost estimating capability of cybersecurity is critical to understanding how much these initiatives will cost.

This paper explores the subject of cybersecurity with a focus on improving IT cost estimates. This research illuminates recent cybersecurity advancements, including Zero Trust Architecture (ZTA) and Secure by Design, and assesses their implications on recurring and non-recurring cybersecurity costs. It also examines previous cost research on various cybersecurity topics and leverages this work to offer potential ideas for how to collect and categorize data for key development and sustainment cost drivers. Finally, the research applies these ideas using hypothetical case study programs, specifically the development and sustainment phases of an IT system.

Intended to promote a uniform approach to cyber cost estimating, this paper introduces a Secure Software WBS, recommends approaches to data collection, and promotes common definition of cybersecurity costs. The Secure Software WBS can serve either as a stand-alone structure or a supplemental resource to integrate within other standard structures used in government and industry. This structure builds on existing authoritative process frameworks, primarily the Secure Software Development Framework (SSDF) created by the National Institute of Standards and Technology (NIST).

1.1. Background

The increase in focus on cybersecurity costs coincides with a time when countless high-profile cyber incidents have ravaged federal and private sector spaces. These incidents compromise data, impair system functionality, and result in massive remediation costs. In Fiscal Year (FY) 2022, federal agencies of the United States Government reported 30,659 security incidents (5). The breach of the network management software SolarWinds in 2020, along with a high-profile breach of Microsoft Exchange the following year, caused costly damage across the federal and private sectors that is still being calculated years later (6). In the private sector, information systems face a significant number of attempted attacks on an ongoing basis, with JPMorgan Chase repelling 45 billion attempted intrusions per day (13).

To adapt to these increasing challenges, federal and private sector organizations have encouraged defensive steps to protect their information systems against future attacks (2). In the aftermath of SolarWinds, the US Government issued several executive orders and memoranda to encourage adoption of standard best practices and improved information sharing policies (11). These recommendations apply to many stages of the cybersecurity process. Of particular interest to cost estimators are enhancements that require updates to the system security architecture and those that emphasize incorporation of cybersecurity requirements in the development of a system. Zero Trust

Architecture and Secure by Design are two new cyber best practices that have captured the interest of the cost community for these reasons.

2. Cybersecurity Conceptual Overview

Zero Trust Architecture and Secure by Design are both new efforts with minimal cost history available. This section will define and summarize the scope and goals of each initiative. This context will help estimators understand the cost implications of each effort for new and existing programs.

2.1. Zero Trust Architecture

Traditional cybersecurity architecture utilizes a perimeter-based model. In this context, a "perimeter" is a physical or logical boundary of a system within which a particular security policy or architecture is applied (7). In a perimeter-based paradigm, all assets located within the enterprise network boundary receive "implicit trust", meaning that access is not re-verified once items are within the perimeter (2). A fundamental weakness of the perimeter model is that once a malicious actor breaches a system perimeter, there is no further protection within the network from additional damage (9).

To address this weakness and move away from relying exclusively on perimeter-based architecture, the U.S. Office of Management and Budget (OMB) has required Federal agencies to implement a Zero Trust strategy (11). Zero Trust emphasizes a "trust but verify" approach to user authentication. This approach assumes that all devices, even those within a network, have the potential to be compromised and thus require continued authentication (9). Instead of a perimeter approach, Zero-Trust emphasizes the securing of all communication regardless of network location. Continual monitoring of user authentication and authorization, along with comprehensive tracking of all system assets, is enforced (9). Figure 1 shows the areas impacted by Zero Trust, as defined in the Cybersecurity and Infrastructure Security Agency's (CISA) five "pillar" model of Zero Trust maturity [4].

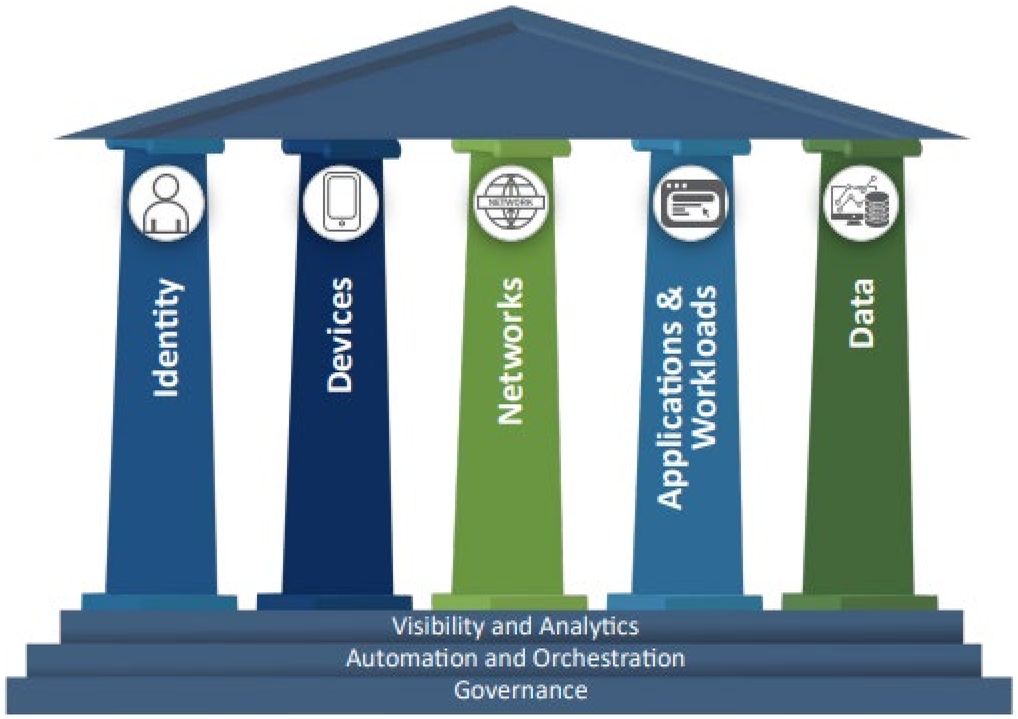


Figure 1: CISA Zero Trust Maturity Model.

In the context of an IT system, the term Zero Trust Architecture (ZTA) represents the application of these Zero Trust principles into an enterprise's cybersecurity plan. ZTA is not a single, unified architecture itself. Instead, it prescribes a set of guiding principles to help improve an organization's security posture [9]. Like many other IT paradigms, programs can apply ZTA tenets to a system in many ways, depending on an organization's specific needs. Converting a program to Zero Trust does not occur over a single cutover event, it requires gradual evolution to Zero Trust principles while strengthening the perimeter-based security architecture that remains. NIST anticipates that many systems will use hybrid approaches, using Zero Trust architecture in higher-risk processes but maintaining traditional perimeter approaches in other areas where ZTA is less practical [9].

2.2. Secure by Design

The Secure by Design approach to software development expands on similar cybersecurity architecture themes with broader implications beyond the authentication process. This approach considers correcting cybersecurity vulnerabilities as a built-in requirement that must be addressed during software development and thus prior to software release. Historically, software products have relegated many features seen as critical for optimal system security to optional add-in features, forcing resource-constrained customers to choose between cost savings and system security (3). Under a Secure by Design approach, software is developed and configured in such a way that the most secure configuration is a default baseline. CISA describes application hardening, supporting application security features, and secure default settings as vital characteristics of a Secure by Design system (3).

Multiple federal and private sector organizations developed elements of this approach concurrently, with support from CISA (3). In a 2023 memo, CISA outlined key concepts for implementing a Secure by Design standard. The agency defines Secure by Design as the building of information systems in a way that protects malicious actors from gaining access to data, devices, and software (3). This approach applies to both custom software development efforts and Commercial Off-The-Shelf (COTS) software purchases, each requiring distinct efforts.

A Secure by Design approach is compatible with both Waterfall and Agile software development paradigms. Presently, neither SDLC process adequately addresses cybersecurity requirements. The weaknesses that a Secure by Design solution would remediate are common to each of these development paradigms, along with hybrid development methods (3). As the name implies, Secure by Design is more commonly associated with new systems or systems undergoing a complete rebuild.

Regardless of development paradigm used, Secure by Design limits the type of programming language used. To be compliant with Secure by Design principles, a system must use memory-safe programming languages. Recently, both Microsoft and Google have identified that a lack of memory safe languages causes most software vulnerabilities (10). Several widely used programming languages, including C and C++, are not memory safe. This means malicious actors can exploit how these programs manage their memory, causing crashes or altering executable instructions [(3)]. Memory-safe languages, such as C#, Java, and Python, avoid these memory safety risks [(10)].

3. Estimating Implications and Challenges

Secure software development poses many of the same estimating challenges that cost estimators face with other emerging technologies. Most notable is lack of a historical precedent and therefore data that facilitates understanding of potential cost drivers and their respective influence on cost. In short, cost estimators don't have the cost, technical and programmatic data required to develop parametric cost estimating relationships (CERs) or even scaled analogy estimates.

Additionally, concepts like ZTA and Secure by Design represent guidelines, rather than absolute standards. This introduces ambiguity in generating requirements and schedule assumptions. It may take some time to gather enough supporting data to make generalizable assumptions on secure cyber development. However, careful approaches to data collection and organization may expedite that process.

3.1. Data Availability

Multiple obstacles limit the ability to quantify the effort needed to integrate cybersecurity into the SDLC. While most work would be performed by contractor labor, few mechanisms exist to require the effort to be quantified with enough detail to provide meaningful estimates. Given these data limitations, many estimators find themselves with only general program sizing metrics available to understand the level of effort required.

For programs in the software development phase, the best sources of available security data may be found in requirements documentation. Examples of potential sources include an Operational Requirements Document (ORD), a Concept of Operations (CONOPS) or system architecture documents. Artifacts produced during the security engineering phase, such as Plan of Action and Milestones (POAM) documentation, can help estimators quantify the amount of development effort needed to remediate security vulnerabilities. However, POAMs are usually not completed until the conclusion of security assessment. For systems in sustainment, there are more potential data sources available. Artifacts from asset management or security monitoring systems should help the estimator quantify the number of assets within a system.

3.2. Cost Estimating Methodology

Despite these obstacles to obtaining data, programs still need some way to estimate the cost of developing security requirements into software development. Some practitioners have already researched ways to improve cost estimating assumptions on this subject. Much of this research has focused on adapting function point estimates or Constructive Cost Model (COCOMO II) outputs calculated using Source Lines of Code (SLOC). Some of these approaches apply factor-based modifications to development efforts, with a focus on how security influences the total cost of software development. Other approaches consider cybersecurity vulnerabilities using terms like “negative use cases” or “abuser stories” to quantify the extent of secure software development requirements at the system architecture level.

A recent paper sought to quantify this effect by developing a rating scale and applying it to development programs using COCOMO Effort Adjustment Factors (EAFs) to quantify the cost impact of cybersecurity in the development process. This research developed a rating score and used it to apply effort multipliers for security development requirements, ranging from a 19% minimum increase to a 102% maximum (12). Future research is likely necessary to validate recommended multipliers within this range. At the present time, estimators cannot stop at only adjusting development costs for anticipated security requirements. Credible estimates will require a holistic understanding of how cybersecurity impacts many different parts of a development program, including systems engineering, software license procurement, and testing.

3.3. Cost Element Structures

Another challenge to quantifying cybersecurity costs is the lack of a structured approach to collecting and organizing cost data. Few existing WBS templates adequately cover detailed cybersecurity tasks and sub-tasks. Many structures group cybersecurity with other unrelated tasks in non-descriptive parent sections, such as Help Desk Support or COTS software procurement. These deficiencies can result in undercounting critical cybersecurity labor and software requirements and, consequently, underestimating the costs. As the cybersecurity landscape increases focus on development and architecture efforts, it is important that cyber costs throughout the SDLC are properly categorized in a way that can be generalized and then mapped to an existing WBS.

Other cybersecurity experts have developed frameworks that can inform WBS development. The Secure Software Development Framework (SSDF) created by NIST outlines a fundamental set of secure practices within the SDLC. These practices are subdivided into four categories, each with a detailed set of sub-practices (8):

1) Prepare the Organization
2) Protect the Software
3) Produce Well-Secured Software
4) Respond to Vulnerabilities

Although not every sub-section of the SSDF represents a discrete work product suitable for inclusion in a WBS, the framework provides a valuable overview of the steps a typical program must undertake to incorporate cybersecurity into the SDLC. Table 1 below compares the parent categories of the SSDF with generalized SDLC categories that are included in a typical IT system WBS.

Table 1: Crosswalk of parent SSDF categories to related WBS categories.

SSDF Category	SSDF Definition (8)	Generalized WBS Crosswalk
Prepare the Organization (PO)	Prepare people, processes, and technology to perform secure software development at organization level.	Systems Engineering, System Development, System Procurement, Training
Protect the Software (PS)	Protect all components of software from tampering and unauthorized access.	System Development, Data Center Support
Produce Well-Secured Software (PW)	Produce well-secured software with minimal security vulnerabilities in releases.	System Development, System Procurement, Testing
Respond to Vulnerabilities (RV)	Identify residual vulnerabilities in software releases and respond appropriately to address and prevent recurrence.	Sustainment-Phase Systems Engineering, SOC Support

This paper offers guidance for mapping cyber tasks specified in the SSDF into a generalized product oriented WBS more suitable for use in LCCEs, such as the IT WBS developed by the Department of Homeland Security (DHS) Cost Analysis Division (CAD).

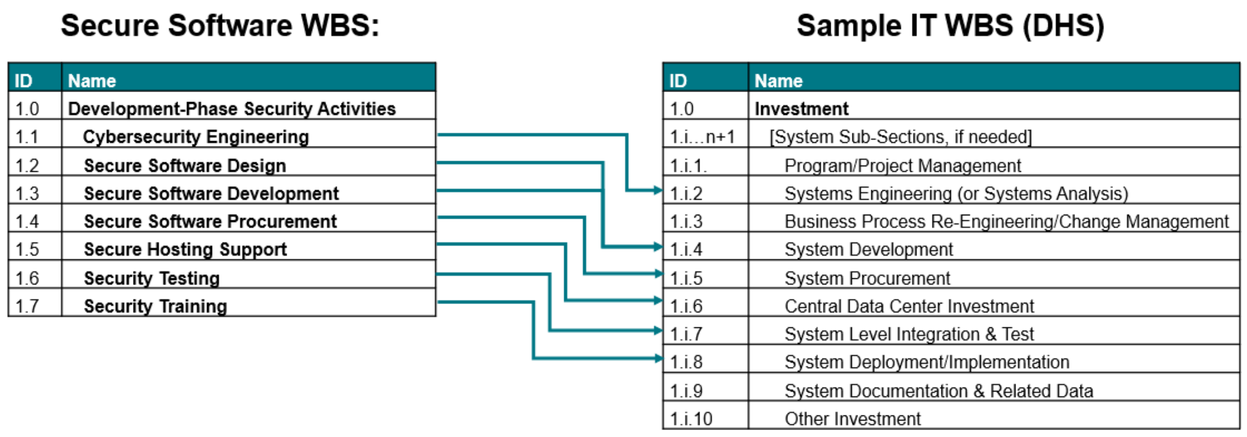
The objective of this structure, referred to as a Secure Software WBS in the following sections, is to link the objectives in the SSDF with tangible deliverables, objectives, and activities common to IT development and sustainment efforts.

4. WBS Categories – Development

The following sub-sections describe categories associated with a secure software development effort. These categories are designed to align with parent-level categories common in most WBS templates representing the SDLC:

1) Systems Engineering
2) System Design
3) System Development
4) Commercial License Procurement
5) Hosting
6) Testing
7) Training

Each effort within these development phase sub-categories is assumed to be non-recurring and only lasting the duration of the development period. However, some efforts should continue in a sustainment phase. These activities are listed in the subsequent section "WBS Categories – Sustainment." Figure 2 below shows a comparison of the Secure Software WBS with a standard WBS. Arrows indicate alignment between categories in the Secure WBS and analogous categories that describe the equivalent SDLC steps in the broader IT WBS. The categories in the Secure Software WBS would be listed in the parent categories at right at a lower level of indentation.



Secure Software WBS:

ID	Name
1.0	**Development-Phase Security Activities**
1.1	**Cybersecurity Engineering**
1.2	**Secure Software Design**
1.3	**Secure Software Development**
1.4	**Secure Software Procurement**
1.5	**Secure Hosting Support**
1.6	**Security Testing**
1.7	**Security Training**

Sample IT WBS (DHS)

ID	Name
1.0	**Investment**
1.i...n+1	[System Sub-Sections, if needed]
1.i.1.	Program/Project Management
1.i.2	Systems Engineering (or Systems Analysis)
1.i.3	Business Process Re-Engineering/Change Management
1.i.4	System Development
1.i.5	System Procurement
1.i.6	Central Data Center Investment
1.i.7	System Level Integration & Test
1.i.8	System Deployment/Implementation
1.i.9	System Documentation & Related Data
1.i.10	Other Investment

Figure 2: Crosswalk of Secure Software WBS with a standard IT WBS: Development Phase.

A full inventory of all lower-level sub-categories is shown in Appendix A. The categories in these sections represent a holistic view of cybersecurity-related activities in the development and sustainment phases of an IT program. Certain cost categories may incur higher costs in a program being developed to Secure by Design standards. Others are more specialized and may not be applicable for all programs. The Case Study sections to follow provide some information as to how Zero Trust Architecture or Secure by Design may impact these costs.

Some costs in this section align with historical cybersecurity efforts, such as obtaining and renewing Authority to Operate (ATO). In US federal government systems, ATO refers to the security authorization process. It represents the US government's acceptance of the risks inherent to a system and authorizes its use. For most federal organizations, an ATO must be renewed every three years or whenever significant changes to the system impact its risk level (1). Due to the need for continued re-authorization, ATO activities should encompass both development and sustainment phases.

4.1. Cybersecurity Engineering

This section includes early-stage efforts associated with assessing the necessary requirements for a secure software development effort. These are roughly analogous to Systems Engineering activities in the broader SDLC. Labor is required to complete these activities, so costs are likely to be measured by Full-Time Equivalent (FTE) or labor hour counts. When collecting this data, it is important to understand how frequently the organization plans to review security requirements. The SSDF prescribes review of security requirements at least once per year; however, more frequent reviews may increase the effort required. Specific sub-activities include defining requirements, determining roles, determining supporting toolchains, and setting criteria of security checks. Most of the SSDF's "Prepare the Organization" sub-tasks align to the Cybersecurity Engineering category. Table 2 below shows the four sub-categories listed under the Cybersecurity Engineering and Assessment category.

Table 2: Cybersecurity Engineering and Assessment WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.1.1	**Security Requirements Definition**	Labor to assess and document security requirements and communicate to software developers.	SSDF PO.1.1-PO.1.3
1.1.2	**Determine Roles and Responsibilities**	Labor to define and document assignment of security roles throughout the SDLC.	SSDF PO.2.1, PO.2.3
1.1.3	**Determine Supporting Toolchains**	Define toolchain categories, specify types of tools in each category.	SSDF PO.3.1
1.1.4	**Define Security Check Criteria**	Define performance and scoring criteria, ensure validity of verification. Includes non-configuration labor associated with obtaining Authority to Operate (ATO).	SSDF PO.4.1

4.2. Secure Software Design

This section includes efforts associated with designing software that is well-secured and addresses known potential security vulnerabilities. During this phase, the organization determines the impact of the risks that a software is likely to face, and how these risks should be mitigated. Following this initial risk assessment, an Independent Verification and Validation (IV&V) effort should review the design and expected risks for concurrence. efforts align to the design stage of the SDLC. If using a broader WBS that groups design and development together, these can be merged with the following category, "Secure Software Development." Like the previous Cybersecurity Engineering category, quantifying these categories is best done in terms of labor. There is little data on potential estimating relationships for these tasks, but the number of risks identified in this phase may impact downstream effort for Secure Software Development. The two sub-categories in this section are Risk Analysis and Independent Verification and Validation (IV&V). Table 3 shows the two sub-categories listed under the Secure Software Design category.

Table 3: Secure Software Design WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.2.1	**Risk Analysis and Mitigation**	Apply a risk-based approach to identify and mitigate cybersecurity risks, record responses and design decisions.	SSDF PW.1.1-1.3
1.2.2	**Independent Verification and Validation**	Review of design and processes with a qualified team of experts not involved in the design.	SSDF PW.2.1

4.3. Secure Software Development

This section includes other software development costs that involve modifying the code of the system. This includes developing security checks, securing source code, and other code protection activities. This section does not include COTS software purchases, and each sub-category in this section is primarily driven by labor costs. When quantifying the size of Secure Software Development, estimators should consider sizing factors of a general nature as well as those specifically related to security. If data is known on risks identified during the Software Design phase, or through negative use cases identified elsewhere in the requirements generation process, effort required to adjudicate these risks should be quantified. If only system sizing data such as Function Points or SLOC is available, applying a factor to estimate Secure Software Development is a valid, if less defensible, methodology. Table 4 below shows the six sub-categories listed under the Secure Software Development category.

Table 4: Secure Software Development WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.3.1	**Software Security Check Development**	Develop processes to create and automate system security checks, including those used by the system to obtain ATO.	SSDF PO.4.2
1.3.2	**Implement and Configure Toolchains**	Includes labor associated with configuring and integrating toolchains, or software tools used to perform complex software development tasks.	SSDF PO.3.2-3.3
1.3.3	**Secure Source Code**	Create source code in accordance with all applicable secure coding practices and review it for vulnerabilities.	SSDF PW.5.1, PW.7.1-7.2
1.3.4	**Software Code Protection**	Store source code in secure repositories with limited access.	SSDF PS.1.1-2.1
1.3.5	**Develop Secure Software**	In-house creation of secure software components	SSDF PW.4.2
1.3.6	**Configure Compile-Interpret-Build Processes**	Configuration of compile, interpret, and build processes to ensure secure-by-default.	SSDF PW.6.1-6.2

4.4. Secure Software Procurement

This section includes efforts associated with the purchase, configuration, and integration of pre-existing security software, including COTS software products. This section includes both commercial license purchase costs as well as the labor needed to set up and integrate these software products. Pricing structures can vary significantly from vendor to vendor; however, understanding the number of users in a system should provide insight into how to estimate through these complex structures. Table 5 below shows the three sub-categories listed under the Secure System Procurement category.

Table 5: Secure System Procurement WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.4.1	**Commercial Security Software Purchases**	Purchase software from commercial, open-source, and third-party vendors. Describes purchase costs from vendors. If desired, costs can be further sub-divided into software category (Authentication, Encryption, Monitoring, Intrusion Protection).	SSDF PW.4.1, PW.4.4
1.4.2	**Secure-by-Default Configuration**	Configure commercial software products to default to maximum security settings.	SSDF PW.9.1-9.2
1.4.3	**Security Hardware Appliances**	Purchase of security hardware, such as Intrusion Detection System hardware, where needed.	SSDF PW.4.1, PW.4.4

4.5. Secure Hosting Support

Whether a system is using an on-premise or a cloud hosting environment, securing the software's hosted environment will require additional labor and resources. The efforts listed in these subcategories do not represent the full scope of hosting expenses but instead represent security-related efforts that will drive labor and storage requirements for a system's hosting environment. Additional labor would be necessary to secure a hosting environment and harden its endpoints, as represented by the first sub-category. Release archiving costs would likely increase the required storage needed for a system. Table 6 below shows the two sub-categories listed under the Secure Hosting/Data Center category.

Table 6: Secure Hosting/Data Center WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.5.1	**Secure Environment and Endpoints**	Separate and protect each environment and harden development endpoints.	SSDF PO.5.1-5.2
1.5.2	**Release Archiving**	Securely archive release files, verification information, and provenance data.	SSDF PS.3.1, PS.3.2

4.6. Security Testing

This section includes efforts associated with integration and test activities conducted to validate that the system is compliant with security requirements. This includes efforts to design and perform security tests for the system. This section also includes testing needed to obtain a system's initial ATO. Table 7 shows the one sub-category listed under the Security Testing category.

Table 7: Security Testing WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.6.1	**System Security Testing**	Design and perform tests to verify compliance with security requirements.	SSDF PW.8.1-8.2

4.7. Security Training

This section includes efforts needed to develop and conduct role-based security training. Depending on the size of a system and the number of roles for developers and end users of the system, the scope of necessary training development can vary. Table 8 shows the one sub-category listed under the Security Training category.

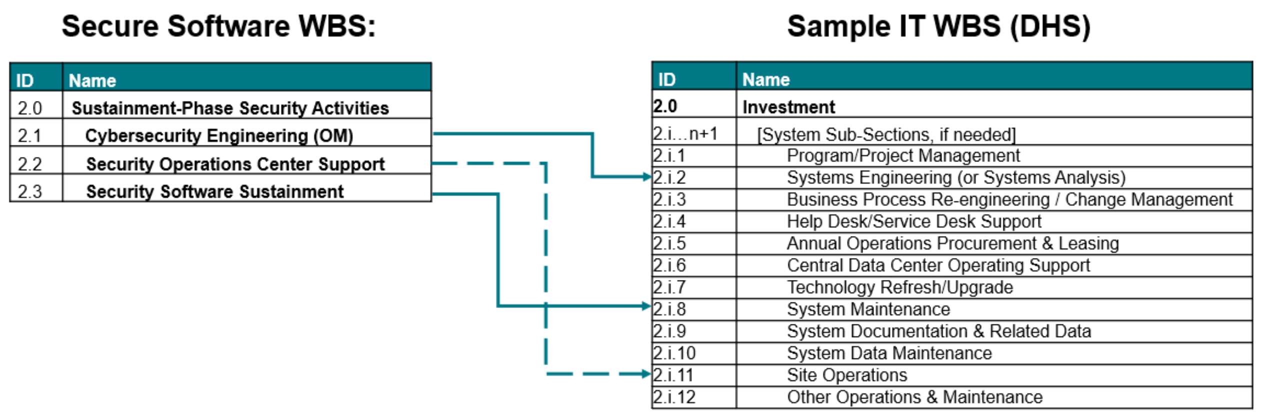
Table 8: Security Training WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
1.7.1	**Security Training**	Develop and conduct role-based security training to system developers and users.	SSDF PO2.2

5. WBS Categories – Sustainment

In addition to the development phase WBS elements discussed in the previous section, our Secure Software WBS also encompasses sustainment phase WBS elements, specifically recurring sustainment activities grouped into three parent categories.

Figure 3 below shows a comparison of the Secure Software WBS with a standard WBS for sustainment phase activities. Arrows indicate alignment between categories in the Secure WBS and analogous categories that describe the equivalent SDLC steps in the broader IT WBS. A full list of the sustainment categories is shown within the complex WBS in Appendix A.



Secure Software WBS:

ID	Name
2.0	Sustainment-Phase Security Activities
2.1	Cybersecurity Engineering (OM)
2.2	Security Operations Center Support
2.3	Security Software Sustainment

Sample IT WBS (DHS)

ID	Name
2.0	Investment
2.i…n+1	[System Sub-Sections, if needed]
2.i.1	Program/Project Management
2.i.2	Systems Engineering (or Systems Analysis)
2.i.3	Business Process Re-engineering / Change Management
2.i.4	Help Desk/Service Desk Support
2.i.5	Annual Operations Procurement & Leasing
2.i.6	Central Data Center Operating Support
2.i.7	Technology Refresh/Upgrade
2.i.8	System Maintenance
2.i.9	System Documentation & Related Data
2.i.10	System Data Maintenance
2.i.11	Site Operations
2.i.12	Other Operations & Maintenance

Figure 3: Crosswalk of Secure Software WBS with a standard IT WBS: Sustainment Phase

5.1. Cybersecurity Engineering (OM)

This section includes efforts associated with secure systems engineering. Once in sustainment, renewal of a system's Authority to Operate (ATO) becomes a significant driver of recurring effort. By default, ATO renewal is required once every three years at minimum. However, a more frequent requirement would impact effort needed for cyber engineering in the sustainment phase. Table 9 shows the one sub-category listed under the Cybersecurity Engineering (Sustainment) category.

Table 9: Security Engineering (Sustainment) WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
2.1.1	**Ongoing Security Assessments**	Efforts required to secure ATO renewal for a system.	For US federal systems [(1)], may be required in other contexts

5.2. Security Operations Center Support

This section includes labor associated with a Security Operations Center (SOC). A SOC is responsible for responding to security incidents, analyzing potential threats, and assessing vulnerabilities within a system, among other roles. These costs are often categorized with Help Desk/Service Desk labor, but separate categorization is important to assure requirements are properly tracked and measured.

Table 10 shows the three sub-categories listed under the Security Operations Center category.

Table 10: Security Operations Center WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
2.2.1	**Monitoring Support (Threat Hunting and Assessment)**	Includes resources for Threat Hunting, Vulnerability Assessment, and other proactive security management activities.	SSDF RV1-3
2.2.2	**Incident Support (Incident Response and Recovery)**	Includes resources associated with incident response and recovery following any security incidents	SSDF RV1-3
2.2.3	**Threat Information Data Acquisition and Subscription**	Purchase/subscription of any external data sources identifying emerging cybersecurity threats.	SSDF RV1-3

5.3. Security Software Maintenance

This section includes renewals of COTS license subscriptions, hardware renewal costs, and other recurring subscription costs associated with secure licenses. Costs associated with specific security license categories, such as authentication, encryption, monitoring, and intrusion protection, should be sub-divided in additional categories where applicable. Table 11 shows the two sub-categories listed under the Security Software Sustainment category.

Table 11: Secure Software Sustainment WBS Summary.

WBS #	Sub-Category	Definition/Description	Source
2.3.1	**Security Software Sustainment**	Renewal of any COTS software licenses purchased in Development phase.	Renewal of licenses cited in SSDF PW.4.1, PW.4.4
2.3.2	**Security Hardware Sustainment**	Renewal of any security hardware appliances purchased in Development phase.	Renewal of hardware cited in SSDF PW.4.1, PW.4.4

6. Application

To illustrate possibilities for data collection and applying the WBS artifacts above, this section showcases two case studies that represent hypothetical program examples of how security costs can be understood across the SDLC. These case study scenarios are generalizable enough to provide insights that can be applied to a wide variety of programs, public and private alike. For each case study, a set of assumptions are made paired with recommended solutions for estimating the hypothetical programs.

At the end of each case study section is a summary of applicable activities specific to the case, along with a proposed estimating methodology given the available program data. For the sake of brevity, the table shows these at the parent category level, showing 7 development categories and 3 sustainment categories. Many assumptions listed in the methodology column draw on accepted best practices for software estimates found in sources such as the Cost Estimating Body of Knowledge-Software (CEBoK-S) or the U.S. Government Accountability Office (GAO) Cost Estimating and Assessment Guide. These case studies do not prescribe universal estimating assumptions or methodologies to apply for each estimate. Rather, they illustrate how our Secure Software WBS can shape methodology development and identify remaining data gaps given the source data available in each case.

6.1. Case Study 1

A logistics management organization is developing a new system for tracking shipments. This system will provide updates to its users as a shipment proceeds from point of origin to its recipient, with updates communicated along the way. The user base of this system will be both internal and external users. The system will be required to comply with all relevant NIST standards and governmental regulations.

The organization has chosen to set up a new development environment for this effort and plans to use its existing authentication and encryption services. The authentication service meets the base NIST standard but will require additional optimization. This optimization is needed to meet both industry-specific standards and government regulations. Additionally, refinements will be needed to comply with Zero Trust architecture within key modules of the system. In addition to these changes, the authentication and encryption services will both require expansion to meet the user base increase expected in the new system.

To augment the existing authentication and encryption services, the system will require additional security services, specifically monitoring and intrusion protection. The organization will have to select, develop, test, and integrate solutions for the system to meet both needs. The entire system will require significant testing to ensure reliability and security.

The system in this case study example has requirements documentation, including a Concept of Operations (CONOPS) and a Functional Requirements Document (FRD) containing some high-level security requirements. The program's development estimate uses these documents to develop a function point count and a Simplified Function Point (SiFP) approach to develop effort and cost estimates.

The estimator already knows that the program will be hosted in a cloud environment and will rely on a stand-alone SOC for security operations support. Although this is a new system, the organization has some cost data on an analogous secure software development for more specialized uses within the organization. This system is approximately 50% the developed size and 25% of the user base of the current system, so any data for this analogous system would need to be normalized to account for size differences with the case study system.

As a new software development, this system would require effort in most of the WBS categories mentioned previously. As with all other forms of cost estimating, applying the right data sources and methodology for each WBS element would determine how thoroughly this structure could be defined. In this case study example, the system has requirements documentation, pricing schedules for currently held software licenses, and preliminary staffing estimates for the development process, including security tasks. Table 12 illustrates the anticipated activities for each WBS category defined. A proposed estimating methodology is listed for each WBS element, based on the assumed available data. As more information is available on the program, an estimator should collect the relevant data to develop estimating methodology and assumptions.

Table 12: Example WBS Application for Case Study 1.

WBS #	WBS Parent Category	Relevant Program Activities	Proposed Estimating Methodology
1.1	**Cybersecurity Engineering**	Pre-design assessments to understand scope of requirements needed to incorporate ZTA and Secure by Design approaches into development. Criteria may be indicated in a program Requirements document.	**Primary:** use scaled analogy data from prior program to estimate cyber engineering team size. **Secondary:** quantify security requirements from CONOPS/FRD, including frequency of ATO/security renewals.
1.2	**Secure Software Design**	Design of how security requirements identified in Cybersecurity Engineering	**Primary**: use results from Cyber Engineering assessment activities,

WBS #	WBS Parent Category	Relevant Program Activities	Proposed Estimating Methodology
		category can be integrated into broader design and architecture of the system.	including number of identified vulnerabilities. **Secondary**: use system size (FP) and number of interfaces for scaled analogy.
1.3	**Secure Software Development**	Develop security functionality as part of broader application development process.	**Primary**: output from Cyber Engineering assessment activities used for Secure Software Design, with any additional effort added from Risk Assessment/IV&V. **Secondary**: use system size (FP) and number of interfaces for scaled analogy.
1.4	**Secure Software Procurement**	Purchase of new COTS licenses for cybersecurity functions (Authentication, Encryption, Monitoring, Intrusion Protection). New development effort assumes 0 pre-existing licenses available.	**Primary**: determine license criteria for Cost-per-License estimate. Potential drivers include number of users, number of servers, number of endpoints/connections, dependent on vendor pricing structure. **Secondary**: number of system users
1.5	**Secure Hosting Support**	Additional labor during hosting stand-up to ensure that hosting environment and endpoints are set up and configured securely.	**Primary**: determine based on hosting size data, such as number of servers, processing cores, and/or virtual machines. **Secondary**: number of system users
1.6	**Security Testing**	All applicable security test events conducted, with issues adjudicated by development team.	**Primary**: using Test Plan or similar documentation, assume number of cybersecurity test cases drives security testing cost. **Secondary**: leverage analogous cyber testing data or use system sizing estimate and analogous system testing factor.
1.7	**Security Training**	Develop and conduct training for any new user roles created by the new system.	**Primary**: number of training products needed for security-related training curriculum. Determine if using in-person ILT, webinar, or video training. **Secondary**: number of system users
2.1	**Systems Security Engineering (OM)**	Sustainment-phase security engineering activities, including operational security assessments and renewal of system ATO.	**Primary**: measure frequency of ATO/security renewals. **Secondary:** use scaled analogy data from prior program to estimate cyber engineering team size.
2.2	**Security Operations Center Support**	Stand-up of program SOC support for proactive (monitoring and assessment of potential threats) and reactive (incident response and recovery) functions.	**Primary**: number of incidents and/or support tickets from analogous program. **Secondary**: number of system users
2.3	**Security Software Maintenance**	Annual maintenance of any COTS products purchased in “Secure Software Procurement” category.	**Primary**: leverage Cost-per-License metrics used in Secure Software Procurement, with applicable renewal costs in place of procurement. **Secondary**: number of system users

6.2. Case Study 2

A federal organization’s IT asset management system is transitioning from a perimeter-based security paradigm to a Zero Trust security paradigm. The system tracks and manages access to multiple types of devices, including laptops, cell phones, and network devices belonging to the organization. The system is currently in operation and has limited security solutions in place. As part of the updates for this system, the organization plans to implement Zero Trust authentication, encryption, and monitoring solutions. The system has a pre-existing development environment which will need to be assessed to ensure that there are no vulnerabilities in development tools and processes being used by the system.

The security solutions being implemented are required to meet all base and sector-specific standards, along with broader government regulations. Compliance with these standards may require some rework of the overall system to ensure that it continues to function as planned once the security improvements are implemented. Additionally, as part of this effort the government has determined that a thorough assessment of the security of the system should be conducted to allow the identification and remediation of any issues that are outside of the scope of already planned improvements.

Because the program is in sustainment, there is available data for actual costs. The program has also provided requirements data, including SLOC counts, number of users, and number of servers. The system uses an enterprise-wide SOC and SOC support costs are charged back to the organization. The organization has also implemented ZTA on multiple other systems of varying sizes and has summary-level ZTA development costs for each. Table 13 shows how the example WBS would capture the range of efforts required for this case.

Table 13: Example WBS example for Case Study 2.

WBS #	WBS Parent Category	Relevant Program Activities	Proposed Estimating Methodology
1.1	**Cybersecurity Engineering**	Primary effort involves engineering for ZTA optimization. Secondary efforts include proactive assessment of system for additional vulnerabilities. Any additional vulnerabilities are flagged for subsequent Design/Development sections.	**Primary**: adjust staff costs current security team to include expanded scope or ZTA development. **Secondary**: quantify security requirements from CONOPS/FRD, including frequency of ATO/security renewals.
1.2	**Secure Software Design**	Redesign of existing system components, determination of ZTA boundaries.	**Primary**: use results from Cyber Engineering assessment activities, including number of identified vulnerabilities. **Secondary**: use system size (FP) and number of interfaces for scaled analogy from prior programs.
1.3	**Secure Software Development**	Primary effort includes development and integration of new functionality for ZTA. Secondary effort includes execution of any vulnerability remediation flagged in Cybersecurity Engineering category.	**Primary**: use results from Cyber Engineering assessment activities used for Secure Software Design, with any additional effort added from Risk Assessment/IV&V. **Secondary**: use system size (FP) and number of interfaces for scaled analogy from prior programs.
1.4	**Secure Software Procurement**	Purchase of new COTS license types and increased quantity of existing licenses. For ZTA, heavy focus on authentication software such as SSO and MFA.	**Primary**: determine license criteria for Cost-per-License estimate for any new licenses. Potential drives include number of users, number of servers, number of endpoints/connections, dependent on vendor pricing structure. **Secondary**: use number of users for scaled analogy for prior programs that completed ZTA re-architecture.
1.5	**Secure Hosting Support**	Minor re-configuration of environment, assumes limited "new" effort needed.	Minimal direct cost assumed, but hosting footprint may need to expand to host new security software beyond expected hosting growth.
1.6	**Security Testing**	Testing of any changes made in Development process to ensure no inadvertent damage to functionality or system security.	**Primary**: number of test cases if Testing Plan is available. **Secondary**: extrapolate from past actuals, adjust to accommodate adds, removals, and changes to security testing made during ZTA development.
1.7	**Security Training**	Likely no need training events, other than process knowledge for end-user Help Desk (not in scope) and SOC (possibly in scope).	Minimal direct cost assumed but determine if training module updates exceed normal training update effort.

WBS #	WBS Parent Category	Relevant Program Activities	Proposed Estimating Methodology
2.1	**Systems Security Engineering (OM)**	Sustainment-phase security engineering activities, including operational security assessments and renewal of system ATO.	**Primary**: measure frequency of ATO/security renewals. **Secondary:** use scaled analogy data from prior program to estimate cyber engineering team size.
2.2	**Security Operations Center Support**	Likely few changes unless authentication management creates additional scope.	**Primary**: collect cost drivers used by enterprise SOC (i.e., users, servers, and/or T-shirt sizing) and adjust charge if needed. **Secondary**: scaled analogy from other systems that completed ZTA update to determine amount of labor needed.
2.3	**Security Software Maintenance**	Include any additional COTS license purchases in assumptions for annual sustainment and subscription fees.	**Primary**: leverage Cost-per-License metrics used in Secure Software Procurement, with applicable renewal costs in place of procurement. **Secondary**: number of system users

7. Limitations

The primary limitation of this research is the lack of historical cost, schedule, and requirements data on actual programs undergoing development to test the proposed structural framework. Regardless of the limitations of available data, the researchers' objective is that better definition of the work required for development and sustainment of current cybersecurity best practices may improve the future data collection prospects. Better data collection will help test existing hypotheses on potential cost drivers and longer-term trends in the relationship between development and sustainment costs. Under a Secure by Design structure, a greater emphasis on security during development could theoretically reduce downstream costs, including software sustainment (2). However, Secure by Design is a new paradigm, and there is not enough available cost data to support this assertion. As a result, assumptions of cost savings should be viewed with extreme caution until enough time has passed to produce actual costs.

Another limitation of this research is the fact what constitutes a useful structure for understanding current and emerging cybersecurity paradigms today may be obsolete in the future given the rapidly changing nature of cybersecurity. The evolution of cybersecurity threats as well as responses to them will likely continue to evolve over time and so too should the structure that enables collection, understanding and estimation of costs.

A final limitation of our research is scope. By design, this research focused on information systems without specific consideration for embedded hardware or other specialized uses. Our case study examples were chosen to represent very generalizable systems. Cybersecurity requirements will likely be different for a system that manages defense, intelligence, public works, or strictly private-sector applications. Additional research would be needed to determine the exact nature of customization needed to estimate cyber costs in each of these use cases.

8. Conclusion

This groundbreaking research presented in this paper should serve to de-mystify cybersecurity costs for IT programs. We developed a standard structure, the Secure Software WBS, that cost estimators can and should use today to collect, understand, and estimate cyber-related costs. We provided two cases studies that demonstrate how the structure can be used for two different programs, one that’s a new development program and the other that’s an existing program undergoing an update.

We are confident that the Secure Software WBS will facilitate understanding that leads to improved data collection and cost estimates that enables more informed decision-making.

References

1. Bennerson, Jo Anna. (2017, March 17). *Navigating the US Federal Government Agency ATO Process for IT Security Professionals*. International Systems Audit and Control Association.
2. Executive Order 14028. (2021, May 12). *Improving the Nation's Cybersecurity*.
3. United States Cybersecurity and Information Security Agency. (2023, October). *Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Secure by Design Software*.
4. United States Cybersecurity and Information Security Agency. (2023, April). *Zero Trust Maturity Model*. Version 2.0 https://www.cisa.gov/sites/default/files/2023-04/zero_trust_maturity_model_v2_508.pdf
5. United States Government Accountability Office. (2023, December). *Cybersecurity: Federal Agencies Made Progress, but Need to Fully Implement Incident Response Requirements* (GAO-24-105658). https://www.gao.gov/assets/d24105658.pdf
6. United States Government Accountability Office. (2022, January). *Cybersecurity: Federal Response to SolarWinds and Microsoft Exchange Incidents* (GAO-22-104746). https://www.gao.gov/assets/gao-22-104746.pdf
7. United States National Institute for Standards and Technology. Computer Security Resource Center. (Accessed 2024, January 26) https://csrc.nist.gov/glossary/term/security_perimeter
8. United States National Institute for Standards and Technology. (2022, February). *Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities* (NIST SP-800-218).
9. United States National Institute for Standards and Technology. (2020, August). *Zero Trust Architecture* (NIST SP 800-207). https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf
10. United States National Security Agency. (2022, November). *Software Memory Safety*.
11. United States Office of Management and Budget Memorandum M-22-09. *Moving the U.S. Government Toward Zero Trust Cybersecurity Principles.* https://www.whitehouse.gov/wp-content/uploads/2022/01/M-22-09.pdf
12. Venson, Elaine, Clark, Bradford, and Boehm, Barry. (2024, January). *The Effects of Required Security on Software Development Effort*. The Journal of Systems and Software.
13. Vigliarolo, Brandon. (2024, January 18). JPMorgan exec claims bank repels '45 billion' cyberattack attempts per day. The Register. https://www.theregister.com/2024/01/18/jpmorgan_exec_attacks

Appendix A

Secure Software Work Breakdown Structure (full)

ID	Name	Source Crosswalk
1.0	**Development-Phase Security Activities**	Roll-Up
1.1	**Cybersecurity Engineering**	Roll-Up
1.1.1	Security Requirements Definition	SSDF PO.1.1, PO.1.2, PO.1.3
1.1.2	Determine Roles and Responsibilities	SSDF PO.2.1, PO.2.3
1.1.3	Determine Supporting Toolchains	SSDF PO.3.1
1.1.4	Define Security Check Criteria	SSDF PO.4.1
1.2	**Secure Software Design**	Roll-Up
1.2.1	Risk Analysis and Mitigation	SSDF PW.1.1, PW1.2, PW.1.3
1.2.2	Independent Verification and Validation	SSDF PW.2.1
1.3	**Secure Software Development**	Roll-Up
1.3.1	Software Security Check Development	SSDF PO.4.2.
1.3.2	Implement and Configure Toolchains	SSDF PO.3.2, PO.3.3
1.3.3	Secure Source Code	SSDF PW.5.1, PW.7.1, PW.7.2
1.3.4	Software Code Protection	SSDF PS.1.1, PS.1.2
1.3.5	Develop Secure Software	SSDF PW.4.2
1.3.6	Configure Compile-Interpret-Build Tools	SSDF PW.6.1, PW.6.2
1.4	**Secure Software Procurement**	Roll-Up
1.4.1	Commercial Security Software Purchases	SSDF PW.4.1, PW.4.4
1.4.2	Secure-by-Default Configuration	SSDF PW.9.1, PW.9.2
1.4.3	Security Hardware Appliances	SSDF PW.4.1, PW.4.4
1.5	**Secure Hosting Support**	Roll-Up
1.5.1	Secure Environment and Endpoints	SSDF PO.5.1, PO.5.2
1.5.2	Release Archiving	SSDF PS.3.1, PS.3.2
1.6	**Security Testing**	Roll-Up
1.6.1	System Security Testing	SSDF PW.8.1. PW.8.2
1.7	**Security Training**	Roll-Up
1.7.1	Security Training	SSDF PO.2.2
2.0	**Sustainment-Phase Security Activities**	Roll-Up
2.1	**Cybersecurity Engineering (OM)**	Roll-Up
2.1.1	Ongoing Security Assessments	SSDF RV.1.1., RV.1.2
2.2	**Security Operations Center Support**	Roll-Up
2.2.1	Monitoring Support (Threat Hunting and Assessment)	SSDF RV1-3
2.2.2	Incident Support (Incident Response/Recovery)	SSDF RV1-3
2.2.3	Threat Information Data Acquisition/Subscription	SSDF RV1-3
2.3	**Security Software Maintenance**	Roll-Up
2.3.1	Security Software Sustainment	SSDF PW.4.1, PW.4.4
2.3.2	Security Hardware Sustainment	SSDF PW.4.1, PW.4.4